Annika Peterson, apeterso
Charlie Swanson, cwswanso

Lab 6: Define Dancing

Description of the Hybrid System

We will be modeling a scene from the movie Wall•E. The scene we are modeling is when Wall•E and Eve celebrate their love by dancing through space. During this dance, they twirl around each other in a spiral, keeping up with one another, but never crashing into each other. We intend to model their behavior in this spiral. We intend to model this system as two robots (which are points with radius to represent their body) in a 3-D spiral whose x-y plane is a circle of radius cute_r. This circle will then move forward in the z direction. We are simplifying the scene to when they are within this spiral of constant radius. Wall•E and Eve should never leave this circle and to ensure that they never crash into each other we will require that cute_r is greater than the sum of Wall•E's and Eve's radii.

The properties that are at the core of correct functioning of the system are both for safety and for liveness. The first is the safety property that Wall•E and Eve never crash into each other. The second property is an efficiency property and a love property which is that Wall•E and Eve can never be too far apart from each other. This ensures that Wall•E's control cannot just be to stay put as Eve moves forward. Wall•E must keep up with Eve. To model this, we will stay that Wall•E must be on the opposite side of the circle than Eve at all times and that their z direction is the same (therefore, they are on the same plane).

Proving in KeYmaera

We have two models that will be used to prove these two properties. The first, simple-test.key, models the system without a controller. Simple-test.key models the system where Wall•E and Eve both have constant velocity that is the same and they are moving upward. In simple-test.key, we will prove that if they are on opposite sides of the circle to begin with, they stay on opposite sides of the circle (therefore they never crash and the distance between them is 2*cute_r).

The second model which is in followEve-test.key is a controller for Wall•E that aims at matching Eve's velocity (so accelerating if Eve is moving faster or breaking if Eve is moving slower than Wall•E) while also ensuring that Wall•E stays on his side of the circle (so never accelerates past the halfway mark on the circle. We will prove both that they never crash in this scenario and that Wall•E stays close within his bounds while accelerating. In this version, we will limit Wall•E to only accelerating or breaking.

All of the models assume that Wall•E and Eve's circle is centered at the origin to make the model simpler. Some loop invariants and differential invariants that we will use to prove the above claims are the direction for both the x and y direction are direction vectors and that Eve and Wall•E are always on the circle and never leave it.

More Advanced Models

A third model we have considered that extends the second model is a controller for Wall•E that aims at matching Eve's velocity (so accelerating if Eve is moving faster or breaking if Eve is moving slower than Wall•E) while also ensuring that Wall•E stays on his side of the circle (so never accelerates past the halfway mark on the circle. We will prove both that they never crash in this scenario and that Wall•E stays close within his bounds while accelerating. The difference between this and the second model is that Wall•E is now able to also coast for some time before choosing to accelerate.

We have also considered expanding the model to include accelerating in the z direction as well as keeping up in the z direction, if we have more time. This would be more complicated as we would be able to have Eve and Wall•E on different z planes and would want to ensure that they are never too far apart in the z direction.

Another expansion we have considered is to center the x-y circle not on the origin. This would complicate the math a bit more for the model, but would be more general.