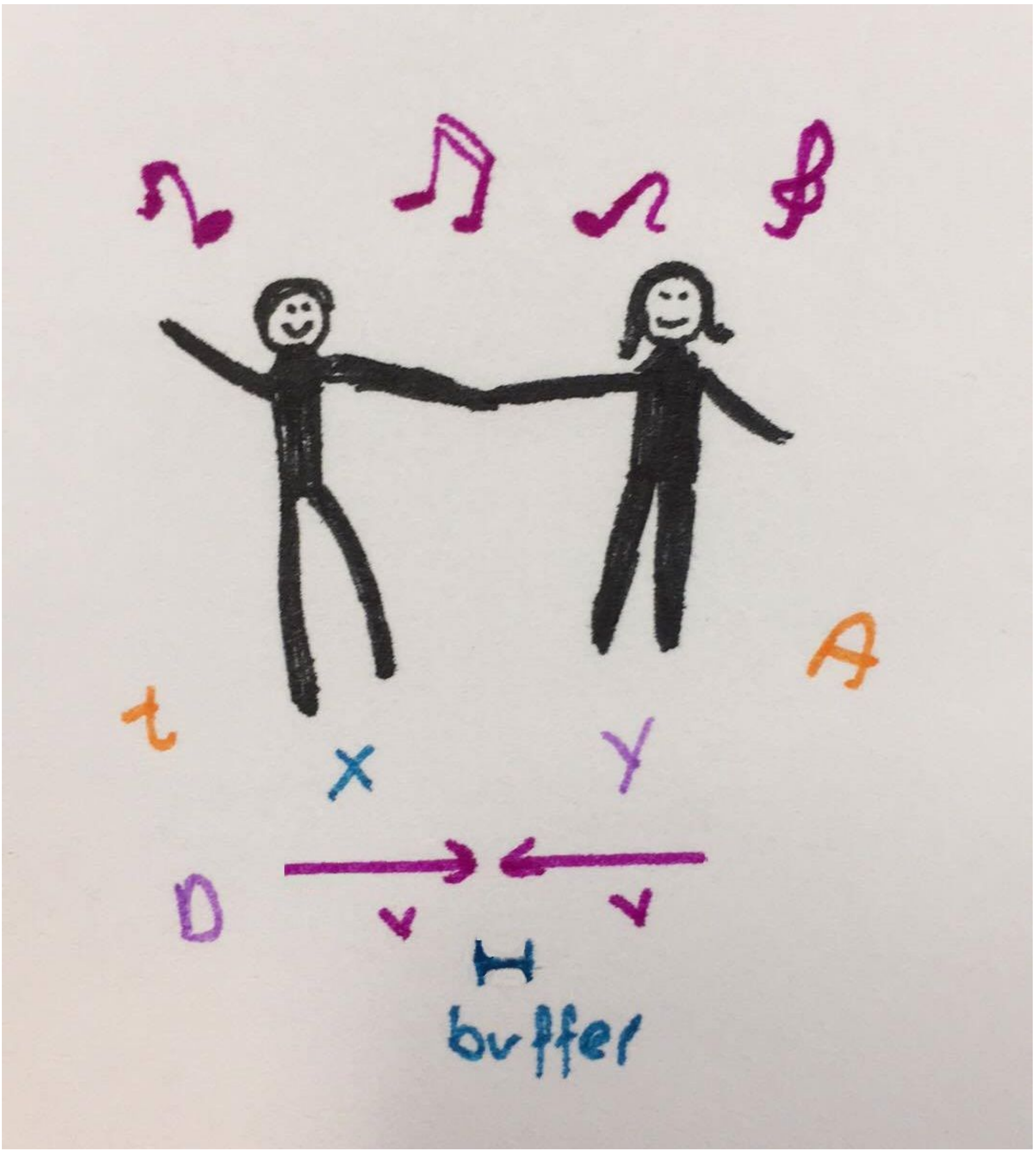
Nazlı Uzgur (nuzgur@andrew.cmu.edu)
Tia Jiang (tianyinj@andrew.cmu.edu)

How to dance without embarrassing yourself: A cooperative hybrid system



Abstract

We have worked on modeling some ballroom dances and moves in hybrid systems. We considered the dancers not stepping on each other's feet as a safety condition and worked to prove that if the dancers dance as they should, the collisions are sure to be avoided. We used the basic movements from the dance Hustle and box steps from Waltz and American Rumba. Since there are two people and they are not against each other as in hybrid games, we have added a cooperative part to the hybrid systems. We worked on and proved 4 models total: Hustle Baseline (Hustle steps, perfect knowledge by the dancers of each others speed), Hustle Guess (Hustle steps, imperfect knowledge), Box steps Baseline (Box steps, perfect knowledge) and Box steps Guess (Box steps, imperfect knowledge). We have proved the safety of the system even when the dancers don't know the exact speed of the other dancer.

Introduction

This paper is on ballroom dancing, specifically Hustle basics and Waltz and American Rumba box steps. We will model them as hybrid systems with the safety condition that the dancers don't crash into each other. We have also worked on modeling these basics in two different ways. In one of the models we're assuming that the dancers know each other's velocity and in the other model, they don't know the speed of the other dancer but they can guess and act accordingly.

These models are interesting for many reasons. First of all, dancing has never been modeled by cyber physical system so the main nature of this paper is something never seen before. We were able to prove the safety of the dancers with these models. Another new approach that we took on was that we proved two models for one dance move where in one of them, the dancers know each other's velocity and in the other one, they don't know but they know knowledge enough for them to make a reasonable guess as it is with dancing. The structure in these are very similar to hybrid games since there are two dancers but the two people are cooperative in this case.

This is also a challenge because it's never been done before. Cyber physical systems and hybrid models exist for proving the safety of electrical devices such as robots or planes but we are using real people in this case. Also, guessing, which is the main point of one of our models, is usually not done in hybrid systems. Another challenge is that cooperative hybrid systems are not as known as the opposite, game hybrid systems, so we don't have a lot of information on this topic either.

These models are pushing the state of the art because ballroom is a topic that has never been done with hybrid systems before and also our models have a very different goal, to have fun, than the systems modeled before. We are also having a guessing model and we are implementing a cooperative hybrid system so nearly everything about this project is state of the art.

In nearly all dances including Hustle, Waltz and American Rumba, there is a lead dancer, called lead and the other dancer is called follow. It is the lead's job to decide on the next move and signal to the follow and it is the follow's job to understand the signal and move accordingly. To make it more clear which dancer we're talking about at any given time, we decided to use different pronouns for the dancers. We'll use male pronouns for the lead and female pronouns for the follow.

We have three models that we've worked on:

Hustle Baseline Model:

In the Hustle Baseline model, we assume perfect knowledge: At any given time, the dancers both know not only their own positions, velocities and the other dancer's position, they also know the velocity of the other dancer. The lead dancer chooses the speed and since the follow knows the exact speed, she also moves with the same velocity.

Here is how hustle works when X's are dancers:

X _ _ _ hands _ _ _ X
_ X _ _ hands _ _ X _
_ _ X _ hands _ X _ _
_ _ _ X hands X _ _ _
_ _ X _ hands _ X _ _
_ X _ _ hands _ _ X _
X _ _ _ hands _ _ _ X

We have built a hybrid model depicting this dance and proved the safety condition of it where the safety condition is that the dancers don't crash into each other and the goal is they have fun! We have modeled this safety condition by saying that the dancers will always have some distance buffer between them. Realistically, the buffer may be the max of the length of the dancers' forearms and how far the dancers move their feet away from their body. Efficiency isn't really a key point in this project because dancing is not done with a goal to get somewhere, there is no end goal. We can consider the system efficient as long as no dancer is just standing still. By nature of dancing, the dancers will have fun so the goal is satisfied by default. We put in x and y, the positions of the dancers and only one v, velocity since the dancers will always have the same speed. In Hustle, the dancers always move with the same speed in opposite directions and in this model, the follow knows exactly what the leads velocity is. The dancers won't start running during dancing, which we will show by a maximum velocity limit of A and they have to hold hands the whole time, which we will show with a maximum distance of D. T and t will be denoting time.

Our model works like this: The lead chooses an appropriate velocity and the follow copies him. An ODE runs for T time. Then the lead chooses another speed the other way and the follow copies him again. An ODE runs for T time.

Here is the model:

```
Functions.
  R T.
  R D. /* Max distance between the dancers so that we can hold hands! */
  R A. /* This is the maximum speed for the dancers. Before we start, me and my partner
agree that nobody runs :) */
  R buffer. /* Comfortable distance between 2 people */

End.

ProgramVariables.
  R x. /* Position of the left person */
  R y. /* Position of the right person */
  R v.
  R t.

End.

Problem.
(x+buffer<=y & T>0 & A>0 & D>buffer & y-x<=D & buffer>0 )
->
[
{

  v := *; ?(v>0 & 2*v <= (y-x-buffer)/T & v<A);
  t:=0;
  {x'=v, y'=-v, t'=1 & t<T}

  v := *; ?(v>0 & 2*v <= (D-(y-x))/T & v<A);
  t:=0;
  {x'=-v, y'=v, t'=1 & t<T}
```

```
  Wait
}*@invariant(x+buffer <= y & y-x <= D)
](x+buffer<=y & y-x <= D)/* Safety condition. */
End.
```

Explanation of proof:
There are two ODE that take action within each loop iteration. In order to prove that the safety condition holds after each iteration, we make a differential cut of the safety condition after the first ode. With the cut, we were able to prove the model by explicitly solving the ode.

Hustle Guess Model:

In this model, we assume imperfect knowledge. Each dancer knows their own position and velocity. They also know the other dancers velocity since they can see them. In this model, they don't know the other dancer's velocity. The lead's job is to just lead, so he just chooses a speed for himself. The follow needs to guess the speed in range [v-veps, v+veps]. This will work because you can see your partner and also have a physical connection while dancing. So as long as eps is a reasonable value, this is a realistic guess.

This is still the same dance, Hustle so the moves are the same. So we are using most of the same variables here with the same intention: x and y for positions, D for maximum distance between the dancers, A for maximum speed of the dancers, buffer for the minimum safe distance between the dancers, t and T for time. The difference from the previous model is the velocities. In this guess model, we have v1 and v2, the velocities of the dancers, vguess for the follow's guess as to what the lead's velocity is and veps as the imperfection of the guess.

As soon as the lead changes his speed, v1, the follow dancer has a chance to adjust her speed v2. But she can only approximate v1 with a margin of error of veps. Furthermore, one important assumption that the model makes is that the veps should be at most 50% of v1. In other words, one essential condition for our dancing system to hold safe is that the follower's guess should be in the range from 0.5*v1 to 1.5*v1.

The model works like this: The lead chooses an appropriate velocity (correct direction and less than A) and the follow guesses his speed with some uncertainty. An

ODE runs as the dancers move for T time. Then the lead chooses another speed for the next step and the follow dancer guesses an appropriate speed for herself again. An ODE runs for T time as both dancers move.

Here is the model:

```
Functions.
  R T.
  R D. /* Let's hold hands and don't tear each other's arm apart */
  R A. /* At the beginning, me and my partner agree that nobody runs :) */
  R buffer. /* Comfortable distance b/w 2 person */
  R veps. /*Fixed imperfection*/

End.

ProgramVariables.
  R x. /* Position of the left person */
  R y. /* Position of the right person */
  R v1.
  R v2.
  R vguess.
  R t.

End.

Problem.
(x+buffer<=y & T>0 & A>0 & D>buffer & y-x<=D & buffer>0 & veps>0)
->
[
{
  v1 := *; ?(v1>0 & 2*v1 <= (y-x-buffer)/T & v1<A);
  vguess := *; ?(((vguess-v1)^2<veps^2) & vguess > veps );
  v2 := vguess-veps;
  t:=0;
  {x'=v1, y'=-v2, t'=1 & t<T}

  v1 := *; ?(v1>0 & 2*v1 <= (D-(y-x))/T & v1<A);
  vguess := *; ?(((vguess-v1)^2<veps^2) & vguess > veps );
  v2 := vguess-veps;
```

```
  t:=0;
  {x'=-v1, y'=v2, t'=1 & t<T}

}*@invariant(x+buffer <= y & y-x <= D)
](x+buffer<=y & y-x <= D)/* Safety condition. */
End.
```

Explanation of proof:
With the assumption that the follower's guess is not more than 50% off the true velocity of the lead, we were able to use the same proof as for our baseline model.

Box Steps Baseline Model:

The box model is a basic step in Waltz and American Rumba. In this move, the dancers are always facing each other and they are always moving towards the same direction while keeping the distance between them the same. This is fundamentally different than the Hustle basics. In hustle, we focused on changing the distance between the dancers every second where in the box step, we want to keep the distance exactly the same. The dancers are facing each other but in this case, they will move like one.

Consider X and Y as dancers:

Step 0:			Step 1:			Step 2:		
_	_	X	_	_	_	_	_	_
_	_	Y	_	_	_	_	_	_
_	_	_	_	_	X	X	_	_
_	_	_	_	_	Y	Y	_	_

There are two kinds of box step. One of them is as above, down and left. The other one is up and right but except the direction, they are exactly the same move so proving the safety for one should prove the other one as well.

We have used some of the same variables again such as t and T for time, A for maximum speed, v for the speed, buffer for minimum and D for maximum allowed distance between the dancers. The safety condition is that the dancers don't get closer than buffer. Unlike hustle, this is a two-dimensional model so the position of a dancer is (x1,y1) and the other one's is (x2,y2).

The model works like this: In this model, we assume follow has perfect knowledge of the lead's move. So at each step, both dancers choose the same speed with the correct direction for each of them and do the move.

Here is the model:

```
Problem.
(y1+buffer <= y2 & (x2 - x1)^2 <= D^2 & T>0 & A>0 & D>0 & buffer>0 )
->
[
{

  v := *; ?(v>0 & v<A);
  t:=0;
  {y1'=-v, y2'=-v, t'=1 & t<T}

  v := *; ?(v>0 & v<A);
  t:=0;
  {x1'=-v, x2'=-v, t'=1 & t<T}

  v := *; ?(v>0 & v<A);
  t:=0;
  {y1'=v, y2'=v, t'=1 & t<T}

  v := *; ?(v>0 & v<A);
  t:=0;
  {x1'=v, x2'=v, t'=1 & t<T}

}*@invariant(y1+buffer <= y2 & (x2 - x1)^2 <= D^2)
](y1+buffer <= y2 & (x2 - x1)^2 <= D^2)/* Safety condition. */
End.
```

Explanation of proof:
The proof for this model is trivial. Both the horizontal and vertical distance between the two dancers holds exactly the same as in the initial condition. So we were able to prove it by making a differential cut of the safety condition after each ODE and prove it by differential invariant or just explicitly solve the ODE.

Box Steps Guess Model:

This model can be seen as a combination as the baseline model for box steps and the guess model for Hustle. The move and the steps are still the box step in this model but this time, we assume imperfect knowledge so the follow needs to guess the speed.

We've done this by making sure that the follow moves just a little slower than the lead, so she is able to follow the lead's moves. We will explain the variables before describing the tactic in more detail.

We have used some of the same variables again such as t and T for time, A for maximum speed, v1 and v2 for the velocities of the lead and the follow, buffer for minimum and D for maximum allowed distance between the dancers, (x1, y1) and (x2, y2) for the positions of the dancers. This time, we have variables oldx1 and oldx2. These are variables that will be used in the dancers adjusting to each others velocity between the two steps of the box steps as will be explained later.

In this model, the follow can approximate the lead's speed with a margin of error of veps, in both x and y directions. From step0 to step1 in the diagram above, the vertical move, to ensure that their vertical distance is greater than buffer(), we make sure that the follow always moves slower than the lead by making her velocity veps less than her guess, vguess. So she moves a little slower than what she expects the lead to do. Since we know that vguess is within the range [v1-veps, v1+veps] where v1 is the velocity of the lead, we are sure that the lead is faster than the follow. Then we can directly prove that buffer() increases monotonically.

Before the dancers start to move horizontally (Step1 to Step2), the lead, on observing that the positions of the dancers aren't perfectly across as they should be,, will make a quick adjustment with his max ability, and make sure that they are facing each other perfectly before the horizontal dance move starts. And we let the follow move at the vguess horizontally.

Here is the model:

```
Problem.
(y1+buffer <= y2 & (x2 - x1)^2 <= D^2 & T>0 & A>0 & D>0 & buffer>0 & veps>0)
->
[
{
```

```
  v1 := *; ?(v1>0 & v1<A);
  vguess := *; ?((vguess-v1)^2<veps^2 & vguess>veps);
  v2 := vguess-veps;
  t:=0;
  {y1'=-v1, y2'=-v2, t'=1 & t<T}

  t:=0;
  oldx1 := x1;
  oldx2 := x2;
  {?x1<=x2; {{x1'=A, t'=1 & t<=(oldx2-oldx1)/A}; ?(t=(oldx2-oldx1)/A);} ++ ?x1>=x2;
{{x1'=-A, t'=1 & t<=(oldx1-oldx2)/A};
  ?(t=(oldx1-oldx2)/A);}}
  v1 := *; ?(v1>0 & v1<A);
  vguess := *; ?(((vguess-v1)^2<veps^2) & veps*T<=D);
  v2:=vguess;
  t:=0;
  {x1'=-v1, x2'=-v2, t'=1 & t<T}

  v1 := *; ?(v1>0 & v1<A);
  vguess := *; ?((vguess-v1)^2<veps^2);
  v2 := vguess+veps;
  t:=0;
  {y1'=v1, y2'=v2, t'=1 & t<T}


  t:=0;
  oldx1 := x1;
  oldx2 := x2;
  {?x1<=x2; {{x1'=A, t'=1 & t<=(oldx2-oldx1)/A}; ?(t=(oldx2-oldx1)/A);} ++ ?x1>=x2;
{{x1'=-A, t'=1 & t<=(oldx1-oldx2)/A};
  ?(t=(oldx1-oldx2)/A);}}
  v1 := *; ?(v1>0 & v1<A);
  vguess := *; ?(((vguess-v1)^2<veps^2) & veps*T<=D);
  v2 := vguess;
  t:=0;
  {x1'=v1, x2'=v2, t'=1 & t<T}


}*@invariant(y1+buffer <= y2 & (x2 - x1)^2 <= D^2)
```

```
](y1+buffer <= y2 & (x2 - x1)^2 <= D^2)/* Safety condition. */
End.
```

Explanation of proof: The vertical dance moves are more or less similar to the ones we did in hustle. The horizontal dance moves needed slightly more techniques: we used the Monotonicity proof rule to ensure that lead can always make the adjustment before both dancers start to move horizontally.

Conclusion:

Our goal was to create 4 models featuring two different dance steps and one baseline model and one guess model for each and prove the safety of it where the safety is that there is a comfortable distance, buffer, between the dancers. We were able to prove this for all four models.

We had some initial challenges how to incorporate the dancing as a model. We thought first about having each foot be a different variable and then calculating the distance between them. However, that turned out to be more complicated than necessary and having a buffer instead would both be easier and solve the problem we had.

Our deliverables are the models and the proofs for the 4 models as .kyx and .kya files.

Appendix:

Liberzon, D. (2013). Limited-information control of hybrid systems via reachable set propagation. *Proceedings of the 16th international conference on Hybrid systems: computation and control - HSCC '13*. doi:10.1145/2461328.2461331