Gerik Zatorski

Final Project Report

My original proposal was to model a free throw by basketball players. The grand hope was to model the shooting form of a couple players. I could do this with some readily available data on their limb lengths and some videotape. I was then going to either introduce joint velocities or (maybe even force the joints) to shoot the ball towards a hoop. If my model was any good, it would come near the hoop. I would then make some very minor tweaks to each player's velocities to find a shot that was go in. With a working model and desired joint positions, I could then apply a little noise and generate a sample of shots for each player and compare. I think this was pretty ambitious, and I turned out to be right.

What I ended up with was a collection of models that needed some crucial connection. The most crucial connection of these connections was a numerical solution to lagrangian dynamics as the shooter generated roll on the ball after release (listed as trelease in the code).

Before we get there, let's start with the shot dynamics. I started with a single model and was unable to spend more time on stretch goals like forced torques and the like. In the "Kinematics of Shot" section, I set up transformations for all the joints and a ball that is fixed to the middle of the hand. (My drawing of these transformations can be seen in the Frames.pdf file) I took some reasonable guesses as joint positions and linearly interpolated them to generate movement of the body. I then calculated the end position of the ball using said transformation matrices. An animation can be seen in Shot_Kinematics_Calculate_Conditions_At_Release.gif file. Note that the ball never releases in this animation because it was my intent to use the same motion after release to apply external forces to the ball. More on that…

The second cell in the code (Release and Roll -> Launch Initial Conditions) is where I attempted to begin using more of the lessons in 314. My first (and final) attempt at modeling this was to constrain the ball to a no-slip condition until it rolled up to the fingertips. At that point I would drop the constraint and allow the ball to fly free. Unfortunately, I was unable to get NDSolve to produce an output for this (I let it run once for 30 min with no luck). It is possible that this would not of produced a good model. Had I noticed any unnatural behavior, it was my plan to drop the constraint a little prior to reaching the fingertips and instead apply a force through the point of contact between the hand and ball. Maybe even adding one more link to the 7R representation to represent the fingertip pads. This link would be tiny and ideally driven with some spring like forces. In the spirit of including visual helpers, you can look at Dropping_The_Ball_Figuratively_Speaking.gif which isn't very interesting, but shows that I can easily start incorporating Lagrangian dynamics calculations at the shot release point.

The part that never connected with the shot model can be seen in the third cell. This was supposed to test whether or not the shot went into a basket. I had constructed a basketball hoop according to NBA regulations. I did some basic testing in it to ensure that it was a possibility and it seems to do pretty well. I then started on the shooting model and never had time to return.

There is more notable things to be done for this part of the simulation beyond obvious things like the mismatched quadrant with the player. Among them are incorporating some slip-grip model of bouncing balls. I had an experimental paper[1] in mind that specifically tested basketballs. The spin of a basketball is very important to the dynamics like many sports and it would have been worth it to incorporate some of those features which accept the shot outcome. The next biggest consideration was a spring system on the front to absorb some energy during impacts. It would not be necessary for the back rim which is firmly attached to the backboard. Regardless, the current state of this model can be seen in the the Court_View.gif and Basket_View.gif files. The second file was included to show the interactions with the rim. As for constraints in the court model, they are fairly simple. Collisions with surfaces are just based on the radius of the ball and it's moving center of mass wrt static obstacles. For the collision with rims, I calculated the planar distance between the center of the ball and the rims and then compare that value with the sum of the radii of the rim (thickness/2) and the ball. All collisions also include a coefficient of restitution equal to 0.8 for the velocities after impact.

[1] http://scitation.aip.org.turing.library.northwestern.edu/content/aapt/journal/ajp/70/11/10.1119/1.1507792