

3-Dimensional Motion of a Point Mass

by Austin Bearden

March 18th, 2018


The goal of this exercise was to explore the difference between modeling the motion of a point mass in 3D space in two different modeling environments. Approach 1 was completed using a pre-created JavaScript physics engine ([CANNON.js](#)). Approach 2 was created using plane JavaScript. A Math library was used in both Approach 1 and Approach 2 in order to provide sine, cosine, and PI. After creating both models, I entered the same starting values in each. I got noticeably different results in the z-component result. I am not completely sure why. One consideration is that the physics engine I used included a mass input. I pre-set this to 0.005 kg. However, my model does not include a mass input, since items fall just as fast regardless of how much mass they have in an environment with no air-resistance. Now, if I was considering friction of a ball, then I would need to know the normal force on the ball, and thus the ball's mass. So, one consideration is that, the physics engine includes several more factors than my point mass model uses. I tried to limit the factors in the physics engine as much as possible. There was a radius input. I preset this to zero. I also preset the mass to a very small value (0.005 kg). I hoped this would help the values match as close as possible.

Approach 1 produced a very large negative z-value while Approach 2 produced a fairly large positive z-value. Again, this may have to do with the fact that the physics engine is incorporating a more complicated model and the fact that there was no radius and yet a mass value. This would cause the point mass model to have very little air resistance, and thus move faster and farther. These are some questions that should be explored further as I decide whether the physics engine I found is a good option for my foosball game. I have given screen shot examples of my two models in use. I have also provided the links to the actual pages on my school server and my code on GitHub.

Approach 1:

Motion of Ball [Approach 1]

Using CANNON.js Physics Engine

Clone Repo at:  [GitHub](#)

Starting X-Coordinate	<input type="text" value="5"/>	meters
Starting Y-Coordinate	<input type="text" value="5"/>	meters
Starting Z-Coordinate	<input type="text" value="5"/>	meters
Speed	<input type="text" value="25"/>	meters per second
X-Y Direction	<input type="text" value="45"/>	degrees
X-Z Direction	<input type="text" value="45"/>	degrees
Time	<input type="text" value="15"/>	seconds
<input type="button" value="Submit"/>		

Output:

X-Position: 166.26094149524152 meters

Y-Position: 166.26094149524155 meters

Z-Position: -1974.1925894563758 meters


Link to Model: https://cs.iupui.edu/~aibearde/PhysicsFoosball/Approach_1/

Link to Code: https://github.com/AustinBearden/PhysicsFoosball/tree/master/Approach_1

Approach 2:

Motion of Ball [Approach 2]

Using Simple JavaScript

Clone Repo at:  [GitHub](#)

Starting X-Coordinate	<input type="text" value="5"/>	meters
Starting Y-Coordinate	<input type="text" value="5"/>	meters
Starting Z-Coordinate	<input type="text" value="5"/>	meters
Speed	<input type="text" value="25"/>	meters per second
X-Y Direction	<input type="text" value="45"/>	degrees
X-Z Direction	<input type="text" value="45"/>	degrees
Time	<input type="text" value="15"/>	seconds
<input type="button" value="Submit"/>		

Output:

X-Position: 192.49999999999994 meters

Y-Position: 192.50000000000003 meters

Z-Position: 128.735 meters

Link to Model: https://cs.iupui.edu/~aibearde/PhysicsFoosball/Approach_2/

Link to Code: https://github.com/AustinBearden/PhysicsFoosball/tree/master/Approach_2