Physics Documentation

Class Diagrams are on UML.pdf, so the image can scale nicely for looking at each class.

References and research material for the simulation (bibliography) etc

https://docs.unity3d.com/Manual/RigidbodiesOverview.html

https://glm.g-truc.net/0.9.9/

5/3/2021 ^

Third Party libraries:

Only one third party library was bootstrap; it was used in the creation of the physics simulation. The bootstrap library includes functions for drawing lines, primitives and textures of our choosing. It also includes the glm library which handles the vector math needed for some of the collision detection and resolution implementations.

What is the physics simulation simulating?

The physics simulation is representing a basic game of pool with striped colour balls and full colour balls, as I had problems with the texture class of bootstrap, I made half the balls red and half the balls blue, however game logic wise it is the same as the normal game. There is one white ball and one black ball, the player shoots the white ball into the other coloured balls to try and hit them into one of 6 sockets around the pool table. If the player sinks the white the other player will have the option to place the white wherever they want in the leftmost third of the table. The player will win when they sink all 7 of their balls in then the black without sinking the white on the same turn. If the other player sinks the black early or sinks the white with the black for the win, the player whose turn it wasn’t wins in that case.

How do all the physical bodies interreacting with each other?

Collisions

Sphere:

SphereToPlane

SphereToBox

SphereToSphere

Box:

BoxToPlane

BoxToBox

BoxToSphere: This takes in the colliding sphere and then calls the SphereToBox function swapping the arguments to fit.

Plane:

PlaneToPlane: As the plane implementation in this physics engine is only kinematic there is no collision occurring.

PlaneToSphere: This takes in the colliding sphere and calls the SphereToPlane function swapping the arguments to fit.

PlaneToBox: Calls the BoxToPlane function swapping the arguments to fit.

Collision Resolution:

-

Improvements that can be made (3 improvements)

Awake and asleep

In the current engine all physics objects are checked to interact with each other in a dual for loop where all objects check all other objects. If an awake and asleep functionality is implemented, it would be possible to reduce the amount of object checks per frame and increase performance. This would be implemented by performing a check on each object while doing the multi for loop, checking for the current velocity of the object, if it is lower then a certain threshold it will be set to asleep, only waking up when another object collides, or another force increases its velocity above the threshold. All current awake objects will be in their own vector/list, by doing this instead of the check being for each object then another foreach object being used, each check for collision will look like for each object awake, then for each object in scene. While this may not be the most efficient implementation, it will increase the speed of the simulation by over 10%, especially in a pool simulation where after the “break” of the initial setup many balls will be left unmoving for an extended period.

Object pooling

When remaking the simulation, each ball is remade and all current balls are deleted if not already, if object pooling is included, when a ball is sunk it will be added to another vector for storage or placed offscreen, then once the game is finished when it is reloaded the objects position will be changed. If implemented object pooling will reduce the chance of problems with memory occurring as all memory needed is loaded at the start of the simulation. It will also be faster then creating each object again on restart as when the object is removed from the simulation its velocity will be set to 0 so on each reset of the game instead of remaking a sphere object only the vector 2 position will need to be changed.

Remaking the maths library

The maths library that is being used by the bootstrap library is the glm maths library, while it is a decent maths library, it was made for opengl programming, not for physics simulations. Because of this some of the functions may not be useful for physics simulations. By basing a maths library off glm and improving / making new functions for physics simulations, improvements can be made. By making a maths library it will also teach me all the functions and improvements that are unknown to me right now and will also reinforced the understandings I currently have.