Custom Physics Documentation

PHYSIC SIM NAME

Benjamin\_Luke\_Scott

2022

Contents

[Visualised Game Using Your Custom Physics Simulation 2](#_Toc95731610)

[Custom Physics Simulation Interactions 3](#_Toc95731611)

[Custom Physics Simulation Class Diagram 4](#_Toc95731612)

[Custom Physics Simulation Potential Improvements 5](#_Toc95731613)

[Improvement #1 5](#_Toc95731614)

[Improvement #2 5](#_Toc95731615)

[Improvement #3 5](#_Toc95731616)

[Third Party Libraries 6](#_Toc95731617)

[References 6](#_Toc95731618)

# Visualised Game Using Your Custom Physics Simulation

The game I chose to visualise my physics simulation is Pachinko. The game works by having kinematic circles fall because gravitational force is being added each frame as well as a collision check. When the circle touches any Physic Object the collision check will apply contact forces causing the circle to bounce. You gain point depending on what coloured box trigger the circles enter at the bottom (see game imagine).

I Created the Custom physics engine using bootstrap and C++. It works by updating the simulation each fixed time step set by the player. (The higher the timestep the less accurate/slower the simulation). Meaning each frame, it will apply a gravitational force and do a collision check on every Physic Object inside the simulation.

**Menu Screen**

Graphical user interface

Description automatically generated

**Help Screen**

Text

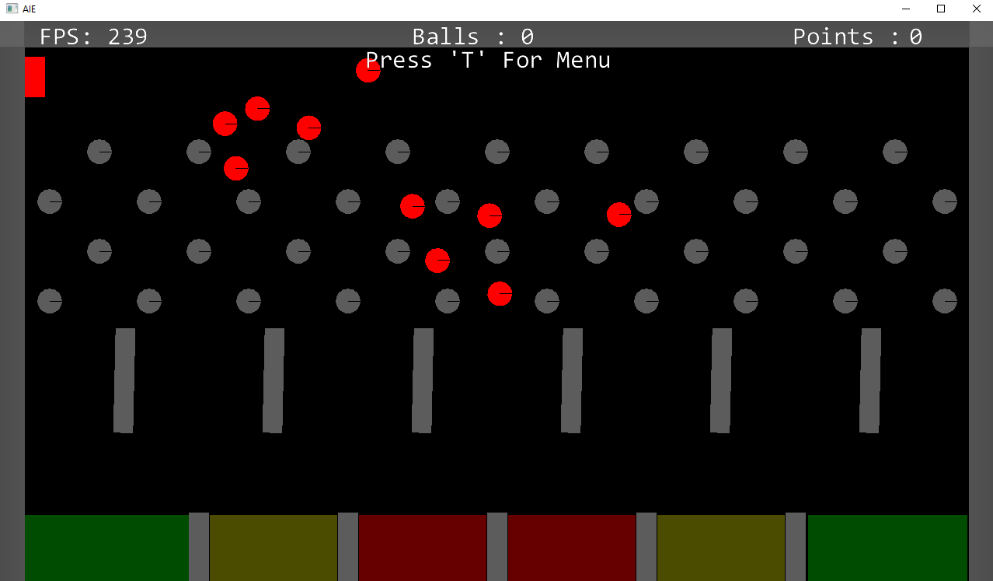
Description automatically generated

**Game Screen**

A picture containing background pattern

Description automatically generated

**Game Screen being played.**



# Custom Physics Simulation Interactions

The custom physics simulation is demonstrating how objects interact when they collide with each other like in the real world for example, if a big ball with lots of mass collides with a small ball with little mass, you would expect the small ball to be launched away.

Each Physic Objects have a Rigid Body attached to allow the physics objects to react to real time physics. This includes reactions to forces, mass, gravity, elasticity, drag and momentum.

If a physic object is a Trigger, it will be none kinematic meaning it doesn’t react to physics instead it will have a bool value toggled/function called on when a physics object collides with it and when it exits the collision.

There are 9 collision checks in this simulation each one taking in 2 Physic Objects and resolving the collision by applying contact forces. The simulation gets both objects mass, collision norm and penetration values to works out where to set its new position each frame.

Plane2plane, Plane2circle, Plane2box, Circle2plane, circle2circle, circle2box, Box2plane, box2circle, box2box.

# Custom Physics Simulation Class Diagram

[Delete This] Illustrate your custom physics systems using a [UML 2.0 Class diagrams](https://aie.instructure.com/courses/623/files/341384/download?wrap=1).

This includes Physics System’s classes, their properties, relationships and how they interact together.  
[/Delete This]

# Custom Physics Simulation Potential Improvements

[Delete This] Examine what improvements could be made to the simulation to support further features and more accurate simulations. (This refers to custom physics simulation library you are creating, not directly the game you have created. These are not mutually exclusive however.) [/Delete This]

## Improvement #1

Using Octree for better collision detection – now we are checking every frame for every object to collide with another. When too many objects get spawned in it will cause our simulation to lag, using octree would improve the collision detection. When an object moves we can check that the line segment between it and the new objects position doesn’t collide with anything else.

## Improvement #2

Collision Layers – I could add collision layers into my simulation like how unity does. This would mean I can have objects only colliding with others on the same layer.

## Improvement #3

More Shapes – I Can add more Shapes into my simulation rather than just having three.

Examples: Triangle, Hexagon, Pentagon and more unique convex shapes.

Eventually I would add the ability to use and detect collision with concave shapes. Using Sat or GjK detection.

# Third Party Libraries

Bootstrap – open-source toolkit / framework.

# References

AIE’s Tutorials - https://aie.instructure.com/courses/813/pages/physics-for-games-creating-a-physics-engine

Octree information - https://castle-engine.io/vrml\_engine\_doc/output/xsl/html/chapter.octree.html#:~:text=Generally%20speaking%2C%20octree%20is%20useful,player%20and%20the%20world%20geometry.