**Custom Physics Simulation – Fredrick Bancan**

**What does this simulation demonstrate?**

This custom physics simulation demonstrates the basic mechanics and theory behind real-time physics simulation in computer games. Shown are multiple basic geometric collision objects in different states (static and dynamic) using mathematical models of physics and physics equations to simulate realistic interactions such as changing velocity upon impact based on each-others mass and elasticity.

As a bonus, the simulation also shows basic implementation of a game loop, use of c++ class hierarchies and examples of efficient and clean real-time performant algorithms which could be referenced or used in future computer game or real time projects which involve physics to a relatively realistic degree.

**What improvements could be made to this simulation to support further features and accuracy?**

There is almost no limit of alterations to such a simulation to make it more performant, accurate or flexible for any given purpose. Generally, the most generally applicable improvement would be to implement some sort of spatial partitioning to allow for increased numbers of individual objects being simulated simultaneously. Apart from that, other performant (advanced) improvements which could be made include implementation of multi-core support, broad faze collision detection, and depending on application, server-side vs client-side collision resolution and filtering.

For accuracy, depending on the level of accuracy demanded by a specific application, there different improvements which could be made and implemented. For specifically physics-based applications/games, there could be a large demand for realism, accuracy, and most likely performance. In these applications, a larger variety of collider types would most likely need to be implemented including capsules, oriented bounding boxes, and convex hull colliders. Algorithms could also be created to automatically generate approximate collision hulls for any object added to the scene. The algorithms responsible for computing the collision response between these collider types must be accurate and performant enough to run up to hundreds of times per physics update. Further accuracy could be implemented, if necessary, for the colliders to interact realistically in intense situations involving very high velocities or forces. In these situations, additional checks would need to be made to ensure there are no objects which phase through each-other without collisions detected, and positions must be properly manipulated to account for overlapping geometry. Certain physics-based applications may also require support for angular momentum, friction, aero dynamics, fluid simulation, fluid dynamics, soft bodies, ragdolls, joints, constraints, torque, projectile simulations, and material simulation. Any of these would need to be implemented while accounting for all the previously mentioned demands and features considered based on the application demands.

Larger alterations to the simulation which could be made include conversion to a 3D scene (including all physics and graphics), sounds, visual representations of contact points and velocity, and graphical improvements such as lighting or textures.

**References and research material used to influence the creation of this Custom Physics Simulation**

* <http://www.chrishecker.com/Rigid_Body_Dynamics>
* *AIE course tutorials and exercises (Advanced diploma of Professional Game Development, 2nd year)*
* *AIE course tutorials and exercises (Diploma of game development specializing in programming, 1st year)*
* <https://developer.mozilla.org/en-US/docs/Games/Techniques/3D_collision_detection>

**Third-party libraries/code used to create and/or influenced this physics simulation**

* *AIE Bootstrap (*<https://github.com/AcademyOfInteractiveEntertainment/aieBootstrap>*)*
* *OpenGL Mathematics library* (<https://github.com/g-truc/glm>)
* *OpenGL (*GLFW <https://www.glfw.org/>)
* *AIE course tutorials and exercises (Advanced diploma of Professional Game Development, 2nd year)*
* *AIE course tutorials and exercises (Diploma of game development specializing in programming, 1st year)*