CM3045 - 3D Graphics & Animation

Peer-Graded Assignment 2: Interactive Physics Scene

Project Report by Hristo Stantchev

Abstract:

This document covers all work and the reasoning behind it in achieving the goals of this peer-graded assignment. It includes the necessary tasks, extension tasks, extra work, references & self-evaluation.

My project is a 3D physics-based bowling game with models I personally made called “Absolutely Realistic Physics Bowling”. It required working with rigid body physics and using forces as well as research into both the rules and the physical aspects (materials, dimensions, weight, etc.) of bowling,

Introduction:

The interactive physics scene presented in this report is a physics-based bowling game, where players need to decide their position, direction, and magnitude of force applied to a bowling ball to hit pins. This report explores its development, and more importantly – how it completes the required tasks given in this peer-graded assignment.

Task 1: 3D Scene with at least 1 object acting under physics

Techniques Used:

1. Transforms – they work using matrix transformations - translation, rotation, and scaling. What is important to note is that order matters. Rotating before translating and vice versa have different results even when using the same parameters.
2. Colliders – They are much simpler than meshes and help establish collisions and triggers in a 3D space by using physics properties, mainly motion & dynamics (more on that in the next sub-point).
3. Rigid Body Physics – Introduces forces to the collision detection. Rigid bodies can have velocities and angular velocities. They hold mass, linear and angular drag (air/fluid resistance). They can have discrete or (several types of) continuous collision detection, interpolation settings for how fluid their motion appears to be [1] and can toggle whether they can use gravity. They can be static (no forces apply), Kinematic (No external forces apply), or dynamic (both external and internal i.e. from script component forces apply).
4. Physics Materials – they give an object with existing colliders parameters on their static and kinetic friction (force required to not move/ to move) as well as their bounciness (elastic collision & Coefficient of restitution) levels.
5. Materials – Concerned with the appearance of an object. The Albedo gives the material a colour (or image) to apply across its faces, whilst bump maps (3D detail without raising poly count) and displacement map (3D detail that raises poly count) give it its roughness values. Values such as metallic and smooth control how an object is shaded (how it reacts to light).

The scene consists of 4 planes enclosing the play space as well as some cubes to represent a bowling lane, starting zone, and pin area. Each element of the bowling lane that is used has its own colliders. The

bowling ball & pins have rigid body components (dynamic) as well as colliders and the interaction between them is key in how this scene works. The scene features 2 point lights to illuminate the room and one spotlight to highlight the pins.

Extra work: I committed to having a custom pin area, as well as modelling the bowling ball & pins. It was my introduction to modelling in 3D (and to using Blender). I also took texture assets from the website PolyHaven, all under the permissible CC0 License (free for personal & commercial use without need to credit). These assets included the texture as well as normal & displacement maps. I worked with them in making my custom materials for the objects.

Task 2: Control Scheme Using Forces

Techniques Used:

1. Forces – Can be applied to rigid bodies. Work with direction, mass, velocity, and delta time. Unity holds 4 force modes - force (continuous w/ mass), acceleration (continuous w/o mass), Impulse (instant w/mass), Velocity Change (instant w/o mass). [2] I used a normalized 3D vector for the direction, then multiplied it by a serialized float for convenient debugging, then selected the force mode for my use case.

Players take shots with the ball by selecting its position (applying force to it between 2 points at the start of the lane, perpendicular to it and changing direction using velocity change). When players press Space, the linear and angular velocities are set to 0 and rotation is frozen for an immediate stop. Then, by rotating the rigid body on its Y axis between -90 and 90 degrees (then converted to Euler) its rotation is selected upon another Space press. Then, a slider appears (0-255) and players need to press space when their desired power is reached. Finally, the direction is converted from Euler to a 3D vector and the ball rigid body is launched using impulse force. The ball rolls towards the pins or misses them based on user input.

Extension Task: Simple Physics Game

I created a whole game using a Finite State Machine model. I made a Game Manager singleton to control that. By accessing each state, simple UI scripts inform the player on their possible actions such as game start, select ball weight (8-12kg, unique model for each), forfeiting their shot, or anything mentioned above. The ball controller is responsible for user input applied to whichever ball is selected as described in Task 2. The Pin Manager holds details on pin instantiation quantity and positions, and interpreting the results of each ball roll through the rules of bowling (Hit, Gutter, Miss, Strike, Spare). This is done by checking each pin's Pin script which decides whether it is standing by checking its rigid body rotation transform.

Conclusion & Self-Evaluation:

I am content with my work but desire to make it more visually appealing.

References & Credits:

[1] On Interpolation – Unity Scripting API
[2] On Forces – Unity Scripting API
[3] Textures – PolyHaven - planks, parquet, wood, leather, tiles, concrete, dirt