Milestone 1 -Sphere Controller

Summer 2014

Due: June 4, 2014 at 11:55 PM

Assignment Type: *Work individually.* Discussing algorithms, design approaches, bugs, etc., is perfectly fine. Sharing code is NOT acceptable. Please document sources of help in your README, especially if you felt the help was significant. The source of help could be a student, newsgroup, web page, etc.

Late Policy:

Starting with Milestone 1, we will use a 2^(*n*+1) late policy point penalty, where *n* is the number of days late. For example, 1 day late is a reduction of 2^(1+1) or minus 4 points. Assignments are considered late if submitted any amount of time beyond the due date/time. Each 24 hours late beyond the due date determines the number of days late. More examples: 2 days late is minus 8, 3 days late is minus 16, 4 days late is minus 32, 5 days late is minus 64 (no assignments accepted any later). Note that this late policy does not apply to assignments that involve in-class presentation, and is subject to change for any specific assignment.

Description:

For this assignment, you will be creating a sphere character controller and 3D model as well as a level for your sphere to explore. The sphere controller will respond to user input by causing a sphere object to interact with the world. The sphere will be able to traverse ground surfaces, jump, change speed, animate via a programmatic rolling animation, and make sounds triggered by various events such as collisions or rolling.

Game Play

- The player will use the keyboard to cause the sphere to roll forwards/backwards and left/right.
- The player can press a jump button to cause the sphere to jump if it is on the ground
- The player can press a speed modifier key to cause the sphere to go faster
- The sphere will not pass through solid objects in the game world
- A 3rd person chase camera will follow the sphere
- You must **NOT** use the physics engine. Use the CharacterController for this assignment.

Additionally, you will configure two different avatar behaviors that give two different feels.

- Heavy, powerful
- Light, quick

Game Feel Constraints:

- You should use textures, motion of the sphere, graphical effects, and sounds (background sounds and/or sound effects), as you see fit to create a compelling experience for the player that expresses the desired game feels.
- The sphere must roll convincingly with the correct rotation relative to translation. There should not be any jerkiness to the motion.
- A keypress of "1" will select the *Heavy* game feel and a keypress of "2" will select the *Light* game feel.

Scoring:

1.) Create a textured sphere model that is controlled by the player as well as a level to explore. (10 points)
2.) Sphere is viewed in 3rd person by a chase camera (10 points)
3.) The sphere can move around the world with the arrow keys and implemented with the Unity CharacterController (10 points)
4.) The speed of the sphere goes faster when the SHIFT key is held down (10 points)
5.) When on the ground, the sphere jumps ONCE then falls back to the ground before it can jump again. (10 points)
6.) The sphere collides with world/level geometry and is stopped from penetrating. Note that you **cannot** use the physics rigid-body simulation built into Unity. Use the Unity CharacterController (10 points)
7.) The sphere animates (rolls) in the appropriate direction when moving and is based on the size and speed of the sphere, as well as the ground surface it is rolling on. Again, you may **not** use the physics rigid-body simulation built into Unity. Instead calculate the rotation using knowledge of geometry. To earn the full points your sphere must roll without spinning faster or slower than it should for the speed it is translating. Also changes in direction should not result in any snapping or jerkiness. (20 points)
8.) Implement the *heavy* and *light* game feels triggered with a keypress or "1" and "2". The two modes should be convincing and easily discernable. (20 points)
9.) Proper submission. See submission instructions below. (up to 20 points deducted for incorrect submission)

Submission:

- WEBPAGE DEMO: Publish the prototype to a webpage as interactive web content (via Unity plug-in). Document your prototype on the web page you publish the game to. Include the details of your prototype's functionality, what requirements did you complete, not complete and which are buggy. For each game feel mode, document what you did to

achieve the desired game feel. **Note that you must deploy your Unity game as a web browser plugin, not a zip file, or other means of distribution.**

- SOURCE CODE: Create a README file that includes any special install instructions the grader will need to be aware of for building and running your code (For example: which scene file is the main file that should be opened first in Unity). Zip up the entire Unity project including this README file. Please keep your assets small if possible without affecting experience (if you use textures and sounds), and remove any unused assets, intermediate and final build files, etc before you zip up the project, so that the upload is as small as possible. Submit your source on t-square uploading the zip file and in the text section place the same documentation from webpage, as well as the link to the web page.

Extra Credit:

Add a particle system that creates a shower of sparks or debris at the point of a collision (up to +5 points)

Hints:

It may be helpful to make your sphere character a hierarchy of an invisible root object with the CharacterController. Attach a child object of the sphere geometry to the root object. Doing so will allow you to leverage the power of the scene graph to rotate the sphere for animation purposes independently of the CharacterController.