Nick Huebner, Jason Jones
CSS 450: Final Project Proposal
11/16/2010

Functional Specification

We are proposing a simple game where an object is falling through a sort of tube-like structure (the level) with a varying surface on the inside. The goal is to minimize the amount of contact that the object has with the inside surface of the tube (level) by manipulating the object.

- The object to be dropped consists of multiple pieces that can be controlled separately.

- The tube is a long, downward pointing structure with a random surface on the inside. The object of the 'game' is to control the shape of the object by rotating and moving it so that it avoids the sides of the protrusions in the tube.

- The object to be dropped is randomly generated. It will have a basic shape such as a circle/ sphere, and several attachments to it that are simple, but multi-jointed. For example, we will attach another object to the base object to represent some sort of appendage or arm.

- The appendages of the falling object are controllable from different views.

- The entire tube is too large to fit onto the screen at one time. For this reason, there will be one area showing the current view, and another mini-map panel showing the entire 'level' but in less detail that the normal view. You can click on a location in the mini-map to make the normal view display that area (instead of following the object that is falling)

- If a part of the object collides with the geography of the level, a sound indicating a collision is played. When the object reaches the end of the current level, a victory sound is played to indicate that the user has successfully passed the level.

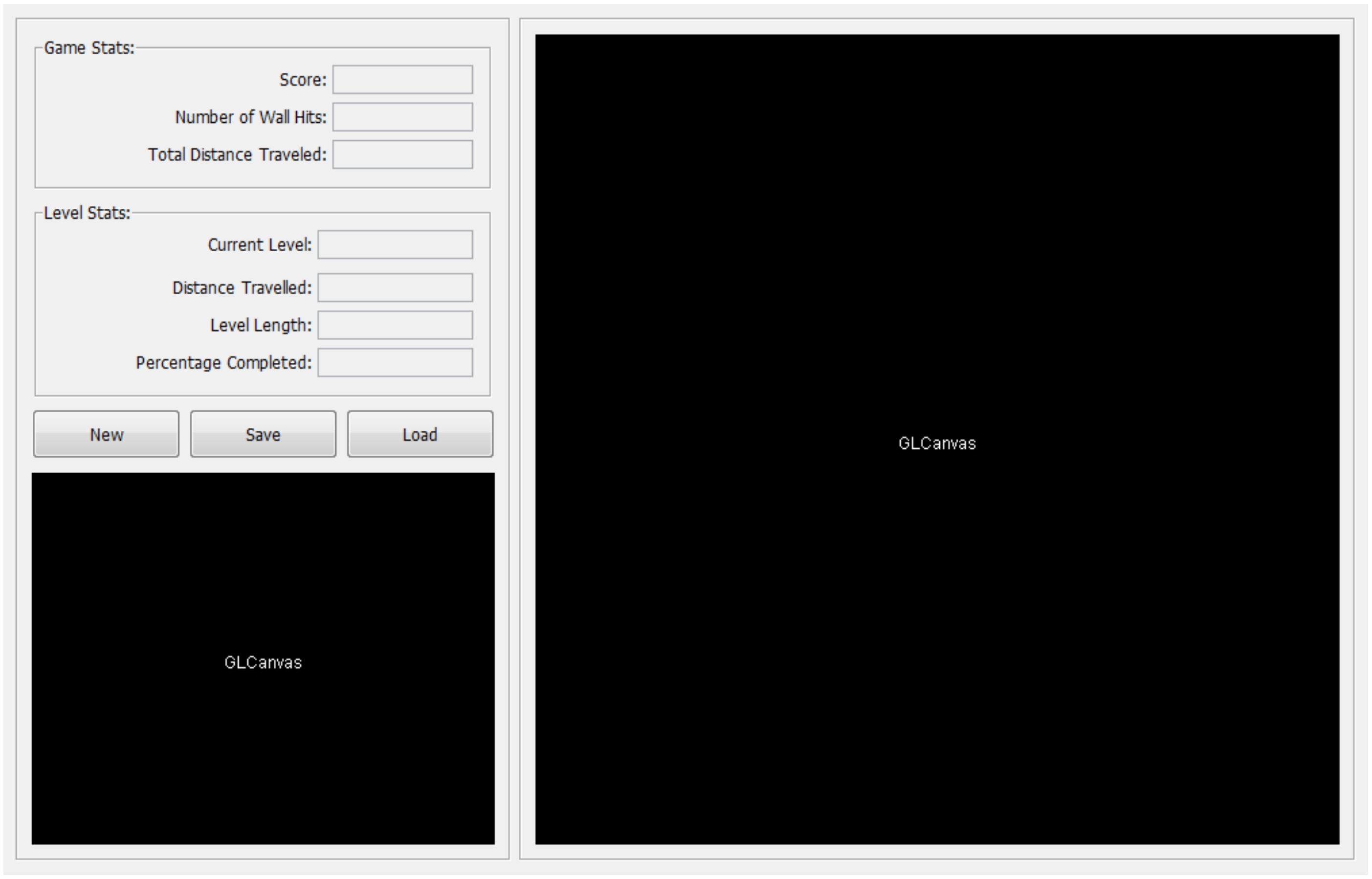
- After a level is passed, the user progresses to a new level that is also randomly generated, but with an increased complexity and the object is falling slightly faster.

- At any time, the user can choose to restart the current level, restart the game (such that a new object to be dropped is generated and a new level terrain is generated). The user can also choose to save the state of the game such that the object, level, and other aspects of the current state (points, collisions, etc) are saved and can be restored later.

- There is an easy mode and a hard mode. In the easy mode, the object is falling at a certain rate that is relatively slow. This allows the user more time to rotate/manipulate the object to avoid collisions. In the hard mode, the object is falling faster, and the variations in the walls of the level are greater so that it is harder to avoid a collision.

GUI Layout

Figure 1: Layout of the GUI



The GLCanvas on the right will be the “main” view/controller pair and will show a top-down view of the falling object. The smaller canvas on the left will be a view/controller pair that shows a side view, which can be panned and zoomed. Another part of the GUI will output various statistics about the game and current level.

Object Interactions

The object has variable pieces to it and it falling through a level that has other objects in it that represent the surface of the inside walls of the tube-like structure. If an element of the falling object hits something in the level, then it is pushed back in such a way as to alter the configuration of the object. For example, if the falling object was humanoid in shape with a ‘neutral’ form (standing straight), and one of the arms of the object hit a protrusion in the all of the level, then the arm might rotate to some extent (based on the velocity or normal of impact with regard to the object it hit).

When the object reaches the bottom of the level, the user proceeds to the next level. The orientation of the falling object is reset (and/or a new object is generated for the next level)

View/Controller Pairs

Each graphical representation of the level is a view/controller pair: the main view showing the object's current position in the currently viewable area of the world, and a mini-map showing the whole level (but in less detail).

- **Main View**
  Mouse actions on this view affect the object. The user can use the right/left mouse buttons to manipulate aspects of the falling object

- **Mini-map View**
  There is a small rectangle on this view showing the area that is being rendered in the main view. The user can click anywhere on the mini-map to move the camera to that location (similar to movement in Mp4)