Assignment 2

Due date: Wednesday, October 18, 2017, by 6:00pm

Note: Late assignments will only be accepted with prior written permission of the instructor. Please make sure your code is in a professional style: well-commented, properly structured, and appropriate symbol names. Marks will be very generously deducted if not!

Description

Note that this assignment requires you build a scenario with 2D game mechanics. You can continue to use Unity, but will need to enforce the 2D-ness yourself. (You may also do this assignment in Java, Python or HTML5/JavaScript if you prefer, but you must then include a text file describing how your code is structured and how to run it.)

Also note that whatever your graphical context, you must do your own collision detection/resolution and manage your own physics (although you may use basic primitives provided by your environment, such as point, line, and simple object distance and intersection).

1. First, you need to produce a game terrain. This will be a 2D profile of a fish bowl, partly filled with water, with some space above it, and a little room on either side of it too.

   The bowl should generally follow a curved shape as per the diagram, including a lip at the top, but does not need to follow a specific curve or have much detail, and a piece-wise linear approximation is fine, as long as it looks reasonably bowl-shaped.

   The water level should be at about the 1/4–1/3 level of the bowl, and defined by a randomized line, with random detail at multiple scales. Each execution of the game environment should produce a different, randomized water line. You may use either midpoint-bisection or 1D Perlin noise to produce the surface line. There is no water motion.

   Figure 1: Overall view of the environment.
2. Within the water is a player-controlled *archerfish*. Archerfish are known for knocking down prey (insects) above the water by shooting droplets of water from their mouths.

Your fish lives in the bowl, just below the waterline. The fish should be generally pointing upward, ±60° from vertical. Its horizontal position is controlled by the player with the 'a' and 'd' keys, and its vertical orientation (angle) controlled by the 'w' and 's' keys. Your fish does not need to swim, or look particularly realistic, but should have a clearly defined position, mouth, and vertical axis. Movement should be constrained so it always and entirely stays in the bowl, but it does not have to have any interaction/physics with the water or other parts of the simulation.

3. Once every few seconds (with some randomization) a bug is generated that moves across the screen, a short distance above the bowl, unaffected by gravity. Bug speed across the screen matches wind speed, which is a wind force, moving left to right. The magnitude of the wind speed changes every 1–2s to a random value, within a range that (if constant) would result in a bug taking at least 2s to cross, and at most 10s.

As with the fish, bugs do not need a realistic representation, but should be generally small. They should, however, have at least two wings and a body. There does not need to be any animation or relative movement of the bug parts.

4. By pressing the spacebar, the player causes the archerfish to shoot water droplets. These should be 3 small circles of different diameter, initially appearing as a non-overlapping group, but otherwise moving independently. The droplets are propelled outward at the same angle as the fish, and subject to gravity. Give them an initial speed such that it takes 0.5s or more to clear the bowl, and generally scale the motion so the effect of gravity is obvious, but they can still be usually shot out of the bowl, from most fish positions. It should also be possible to hit a bug with a droplet.

Droplet motion must also be affected by wind, once above the top of the bowl. (Wind resistance does not need to be considered.)

5. Droplets do not interact with each other (or the fish), but can collide with the bugs and the bowl. Contact with a bug causes a droplet to disappear and the bug to become subject to gravity and to start to fall. (It should be possible for this to result in a bug ending up falling into the water, not just outside the bowl.)

Droplets that encounter the bowl (interior or exterior somehow) should result in collision response, bouncing off the bowl surface as if they were solid objects with very little loss of energy. (This is not realistic of course, but gives opportunities for using reflections from the bowl as part of targeting bugs, although whether that is effective will depend on the bowl shape.)

Droplets that encounter the water, or go offscreen (left or right) disappear. You can discard droplets that go too high as well, with your own choice of height threshold.

6. Falling bugs do not interact with other falling bugs (or the fish), and disappear once they encounter the water. Encounters with bowl surface (exterior or interior somehow), however, should result in a collision response (bounce).

Use a noticeably lower coefficient of restitution than the droplets.

Note that you do not need to model any rotational effects or friction (for the bugs, nor for the droplets).

**What to hand in**

Assignments must be submitted on the due date **before 6pm**. Submit your assignment to *MyCourses*. Note that clock accuracy varies, and late assignments will not be accepted without a medical note: **do not wait until the last minute**.

For the Unity questions, hand in an exported project containing all files needed in order to reconstruct and run your simulations.

For non-Unity questions, submit either an ASCII text document or a .pdf file with all fonts embedded. Do not submit .doc or .docx files. Images (plots or scans) are acceptable in all common graphic file formats.

This assignment is worth 15% of your final grade.