CS 174A
Winter 2019

Fish are Friends, not Food

Team Members:

No changes since the proposal.
004769092 - Kenny Chan
404814243 - Grand Huynh
004789797 - Jonathan Myong

High Level Description:

We created a fish tank simulation where you feed your pet fish and watch him grow. The primary goal of this simulation is to provide a relaxing pastime to help our peers unwind after a stressful week. We aim to provide a comforting companion to those restricted by the apartment pet rules in Westwood.

The fish and fish food are bound within a transparent glass tank. You can enter treat mode with the binding in the control panel. In this mode, when you left click, you randomly drop a fish food for your pet. The fish will swim towards the food and eat it. Upon eating the fish food, it will grow slightly. Once it has eaten enough food, it will become a **super fish** with bright colored scales. In addition, we have created a “follow fish” mode where you can more closely observe your fish and follow it as it wanders around the tank.

Implementation:

- Hierarchical structure of fish: The fish object consists of an spheroid for its main body, two conal cylinders for its middle body, a frustum for its lower body, and a custom object for its tail. Each piece farther away from the main body is dependent on the last. The fish also has two spheres for its eyes, that depend on the main body.

- Lookat matrix for fish: We used the Mat4.look_at() function in order to create a look-at matrix. With this, we applied it to our camera using graphics_state.camera_transform in order to rotate our camera to center the screen at the object of our choice. By repeating this every time in the display function, we were able to track the movement of our fish every frame in order to have a smooth appearance.

- Fish Behavior: Our fish has two general states - idle and movement. In its idle state, it gently stays afloat, slowly sinking down while slowly moving its fins. During movement, our fish has a destination, and moves towards its destination with a curved movement. Without the presence of a fish food object, the fish randomly chooses a goal destination, and goes into movement state every so often. Otherwise, it stays in its idle state. When it detects that a fish food object has been placed, it then changes its destination to be where the fish food object is.

- Parametric curve: We used a parabolic bezier curve to model the motions of the fish. The three points used were the fish's current position, the fish's goal position, and the position with the same y coordinates as its goal and x-y coordinates as its current. Combined with a half period of a sin function representing t, this allowed us to model a relatively dumb fish accelerating to its goal in a realistic manner.
- Custom polygonal object w/ custom positions, normals, and UV: We created a tailfin polygon to make our fish look slightly more realistic. To do this, we created a polygon in shape.js, similar to the cube object. We added in the points needed to generate the polygon, the appropriate normal vectors for each point, and mapped the texture using UV mapping. We broke up the object into triangle polygons and generated the appropriate indices to have our shape resemble a V shaped fin.
- Custom texture based on UV coords: For our texture, we used a stock image found online of a fish scale pattern. After resizing this image, we mapped it out as a 1x1 square and UV mapped the appropriate coordinates to the appropriate sides of our tailfin object. The two flat-faced sides of our tailfin were textured with a V shaped cut out of the fish scale pattern image.
- Unit Collision: There is a collision detector function that factors in the center and size of both fish and food in order to determine if these objects are overlapping. It determines the distance between the two centers by using the distance formula for points. Then, it compares whether this distance is larger than the size of the two objects. If they are overlapping, then there is a collision, and we will allow the fish to eat the food. Everytime the function that draws all foods is called, it will check if each food has collided with a fish in its new position before drawing it. If so, it will remove the food object.
- Gravity: The next height for a falling food is subtracted by the time alive times fall speed. The fall speed was experimentally determined in order to create a more realistic falling animation. When each food object is initialized, its spawn time is stored in an array of food_spawn_t[i]. The time alive is calculated from the difference between the current time and its spawn time.
- Table: The table is constructed by a series of squares used scaled and translated using matrix math in a similar fashion to the previous project. It was textured by using a wood skin that came with the default starter code.
- Glass: For both scenes, we added a boolean scene_comp.trans that specifies if the scene object should be drawn transparent. We created another scene_component called Transparent_Glass that draws only the transparent glass. Then in the render function we check the bool scene_comp.trans to determine whether to draw the scene component as transparent. In order to make the objects look transparent, we enable blending and remove depth testing.

Division of Labor:

Jonathan: Fish Object, Fish Movement, Fish Behavior Logic, Collision Detection
Grand: Table, Transparent Glass, Fish Food Gravity, Collision Detection
Kenny: Custom Tailfin, Mouse clicking, Fish food placement, Camera Tracking

Adherence to plan:

Overall, we were able to reach our high level goals. Our top priorities were to implement camera tracking, game logic, and realistic fish behavior. We successfully implemented camera tracking and game logic. Game logic included feeding the fish by left clicking to drop treats and the collision detection needed to eat the food. We actually added a few extra features, including the ability for the fish to grow and eventually grow bright scales. However, although we achieved many of the project's high level goals, we needed to pivot on the direction on some of our features.

One of the difficulties faced in the project was the creation of the transparent glass. We chose to add a separate seen with a boolean in order for the render function to be able to distinguish which shapes should have been drawn transparent. Although it was a simple fix, we needed a lot of time in order to read through the starter code to get a strong understanding of the calling structure of the provided code.

Another big difficulty we encountered was implementing mouse clicking while allowing for the camera to rotate. We struggled with several possible implementations, with the most promising being factoring in the camera location with the screen location of a click to match that with a location in the tank. However, we decided to just have a key binding for following fish, and to have food appear in random locations.

A third difficulty we encountered was implementing fish behavior. Getting the fish to actually point towards the direction it was heading proved to be a challenge. Our first implementation kept track of it's immediately next position and aligned the fish with the vector difference between it and the fish's current position. However, that resulted in the fish rapidly changing directions in an unnatural manner, probably due to small changes in t disproportionately affecting the inverse tan. We thus implement the correct direction in the xz plane, but "faked" it in the y direction with a sin function.

In retrospect, although we correctly predicted that fish behavior would be difficult, we failed to see that mouse clicking and transparent shading would be some of the hardest parts of our project. We managed to achieve our big-picture goals, and though we weren't able to implement some of our more ambitious goals, we were satisfied with our end product.

References:

eye, glass, wood(original starter code) texture sources!
The glass texture was found at this link:
http://karaelvars.com/wp-content/uploads/2018/10/transparent-glass-png-15-texture-for-free-download-on-mbtskoudsalg-cozy-popular-512x512.png
Fish tailfin texture was found at this link:
https://i2.wp.com/www.knockoutsportswear.com/wp-content/uploads/2016/04/swimskinz-fish-scales-pattern.jpg?fit=800%2C800&ssl=1
Fish food texture was found at this link:
https://www.theaquaponicsource.com/wp-content/uploads/2014/07/p-2047-AFFAB004-2.jpg
The textures were resized using this website
https://resizeimage.net/.