Submarine Simulator Project

By Tao-Sen Chang and Klaudia Przybylska

I. Sun - is located above the skybox. It is the main light source. For rendering, it uses a function drawObjectTexture() and shader_sun files.

II. Submarine:
    1. It is a secondary light source that gives less light than the sun and its position is dynamic - it is controlled by the user. For rendering, it uses a function drawSubmarine() and shader_submarine files.
    2. It is controlled by the user - using key ‘w’, ‘s’, ‘a’, ‘d’, ‘z’, ’x’, ’q’, ’e’ to move to any position or to rotate.
    3. If the distance between shark and submarine is less than a certain number, the shark will attack the submarine. The HP number on the left-top corner will decrease with each attack. The submarine may escape from the shark if the distance between them is greater than a certain number.
    4. It is one of the physics actors with sphere geometry, and it is also a kinematic actor so that it is not affected by gravity. When the shark attacks the submarine, the collision happens.
    5. It has bump mapping located in shader_submarine.

III. Turtle - it moves using Catmull-Rom splines. For rendering, it uses a function drawObjectTexture() and shader_tex files.

IV. Single fish - it moves using Catmull-Rom splines and it has bump mapping. For rendering, it uses a function drawObjectBump() and shader_fish_bump files.

V. Sharks:
    1. It has three stages. The first one is “Normal mode”, where the shark swims around using Catmull-Rom splines, and waits for the submarine closing it. The second one, “Attack mode” happens when the submarine moves too close to a shark. It then proceeds to follow it and attack it. The third mode is called “Go back mode” and is triggered by the submarine moving a certain distance away from the shark. The shark goes back to his original trajectory and switches to “Normal mode”. (It happens in renderScene() function in main.cpp)
    2. It is one of the physics actors with sphere geometry. When the shark attacks the submarine, the collision happens.
    3. They are not affected by gravity (since they are underwater).
    4. For rendering, it uses a function drawObject() and shader_without_tex files.

VI. Boids fish:
    1. They are created using particle effect and use Boids flocking algorithm which is located in separate file Boids.cpp, Boids.h.
    2. For rendering, it uses a function drawBoids() and shader_boids files.

VII. Fog - Sharks, Single fish, Boids fish and Turtle use fog based on their distance to the submarine. If they get too far they “disappear”. (It happens in the fragment shader)

VIII. Vertex shader animations - they are used by Sharks and Single Fish.

IX. Bubbles:
1. They are created using particle effect (Particle.cpp, Particle.h files).
2. They have static environment mapping, are transparent and have blended color of fog to look better. (It happens in the fragment shader)
3. They appear when the submarine is moving, they move up and disappear in 1.5s.
4. For rendering, it uses a function drawParticles() and shader_particle files.

X. Skybox - it is a cube of size 1800x1800x1800. It has fog (in the fragment shader) that is not based on the distance to the submarine. For rendering, it uses a function drawSkybox() and shader_skybox files.

XI. Physics
1. We use the Physx library, and set the gravity to 9.8.
2. There are two physics actors - the submarine and the sharks.
3. The collision happens when the sharks attack the submarine.
4. initPhysicsScene() - it is for initializing the submarine physics actor.
5. createSharkActor() - it is for initializing the sharks physics actor when it changes to “Attack mode”.

XII. Catmull-Rom splines:
1. It is used by Shark, Turtle and Single Fish.
2. It is located in the files: CatmullRom.cpp and CatmullRom.h.