Ocean Simulator: Inception

https://github.com/i12345/ocean-simulator/tree/f0d4c7b4b86193571ed0fc8349a64b3a0f2f9c85

Vision:

The ocean simulator will be a video game that lets players virtually “swim” in an ocean with marine life using motion-control.

Target demographics:

Who would enjoy the ocean simulator? Children and adolescents would enjoy this game. For example, the children’s video *Baby Shark* is very popular. Adults who appreciate nature and marine life and who are willing to experience nature through simulations will also enjoy this product. In order to better understand our target demographic, we could ask students from a local high school to test and review the simulation.

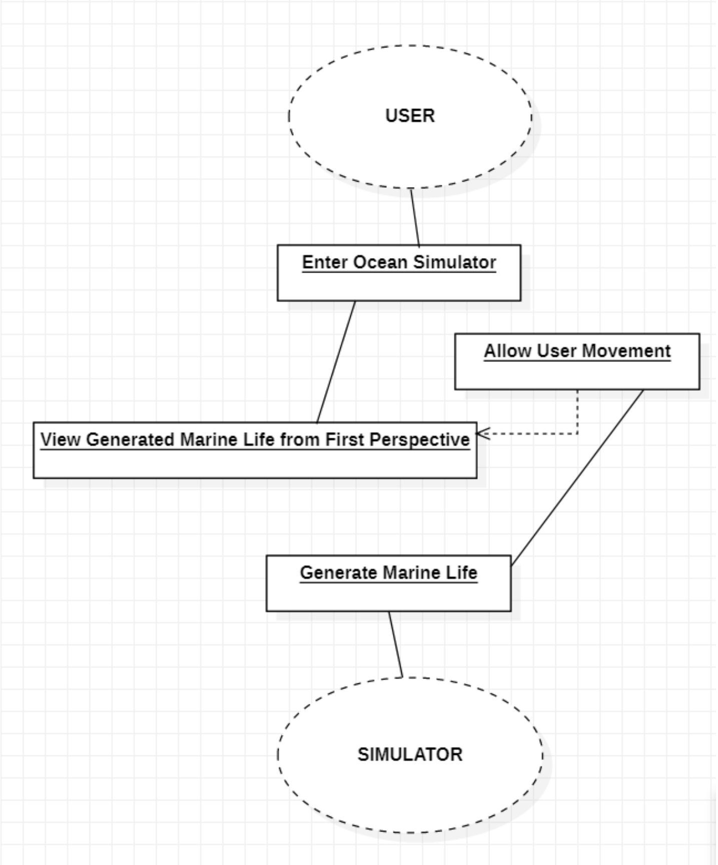
Competitors:

Similar video games and interactive projects featuring marine life already exist, such as Ultimate Ocean Simulator, Ocean Mammals: Blue Whale Mari, Fish Abyss: Aquarium Simulator, and David Attenborough’s Great Barrier Reef. They have many kinds of animals, and Ultimate Ocean Simulator lets the player “play” as one of the marine animals. David Attenborough’s Great Barrier Reef has many educational videos and articles as well. Besides these, there are many non-interactive underwater experiences available on the Internet, such as regular and 3D videos. However, these games lack motion-controlled input.

Move! lets a person play a video game with motion-control, but it doesn’t feature the ocean.

Use-case model:

- Play swimming in the ocean from your home
- Learn about marine life while playing the game
- Go diving with friends without the risks of real diving



Supplemental information:

This game will at least include these animals and plants:

- Two different kinds of fish
- Whales
- Sharks
- Corals
- Algae
- Kelp

It will also contain at least these features:

- Motion-controlled input for the player (a diver)
- One ocean environment to swim around in

If there is extra time available, then other animals and plants could be added, such as:

- More types of fish
- More types of whales
- More types of sharks
- Manta rays and stingrays

- Starfish
- Crabs
- Jellyfish
- Sea anemones
- Lily pads
- Turtles

Also, if extra time is available, these features could be added:

- Multiplayer
- Dynamically generated animals and plants
- Physically based fluid simulation (i.e., more realistic underwater physics)
- Ability to play as one of the animals

Core Features:

- Navigation between main menu, "set up camera", and playing in the simulation
- 3D model / rendering for animals and diver
- Motion-control for the diver
- AI controls the animals' behavior

Business case:

- The free version lets you swim in only a few places.
- If the user wants to unlock all features (like visiting the Great Barrier Reef or dolphins around Hawaii) they must pay for the full version.
- We gather anonymous data on how people respond to ocean life for research purposes (not for ad-targeting methods), and we could sell this data to researchers.
- We can sell the game to universities or schools to use in science class

Software Development Plan:

For the first iteration of the project, we plan to design the overall structure of the simulator as well as design or find models for marine life. We also plan on dividing simple implementation tasks which are shared and reviewed through the GitHub repository. Basic UI implementation and further research of software tools needed to design the simulator will also be done.

Tasks:

- ❑ Set up starting repository with single web page, source code, and compiler configuration
- ❑ Add screens: simulator and "Set up camera" screen
- ❑ Add player (diver) 3D model & texture

- ❑ Make the player able to control the diver with keyboard / mouse
- ❑ Make the player able to control the diver with motion
- ❑ Add generic fish: 3D model & texture
- ❑ Add AI for the fish
- ❑ Work at making the physics realistic for being underwater
- ❑ Add whales: 3D model & texture
- ❑ Add AI for whales
- ❑ Add sharks: 3D model & texture
- ❑ Add AI for sharks
- ❑ Improve the materials & textures used for the fish, whales, and shark to be more realistic

1st iteration:

- Set up starting repository with single web page, source code, and compiler configuration
- Add screens: simulator and "Set up camera" screen
- Add player (diver) 3D model & texture
- Make the player able to control the diver with keyboard / mouse
- Add generic fish: 3D model & texture

Tools:

- PlayCanvas (just the PlayCanvas game engine; not the PlayCanvas editor)
- GitHub repository
- NPM
- TypeScript compiler

Research:

- Marine habitat
  - Where should it be?
  - What animals live there?
- Marine animals
  - How do they behave?
    - What kind of AI system is used?
    - How can we simulate the AI for them?
  - What do they look like?
- How to use PlayCanvas
  - Loading 3D models
  - Rendering realistic materials
  - Moving from the main menu to the "set up camera" screen
- How to use ammo.js physics engine (with PlayCanvas)

    - Collision detection
    - Soft-body physics
- Motion-controlled input
    - How to do
    - How can it be used to control the diver’s movement?

Other Resources:

- Assets: 3D models and images for marine life

UI Prototype:

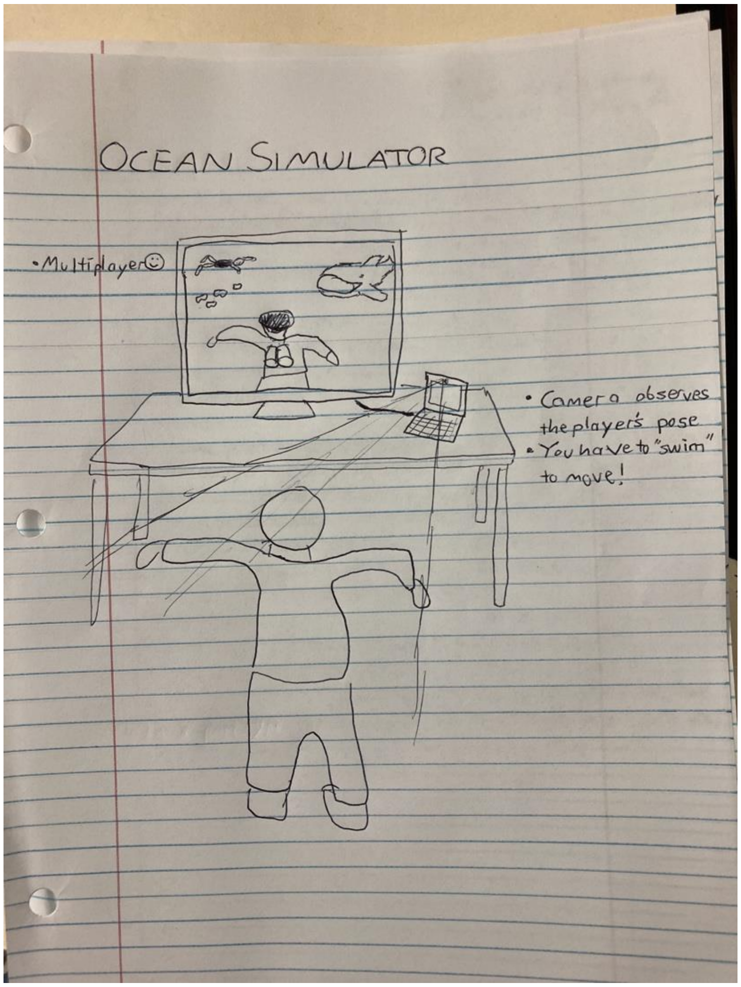
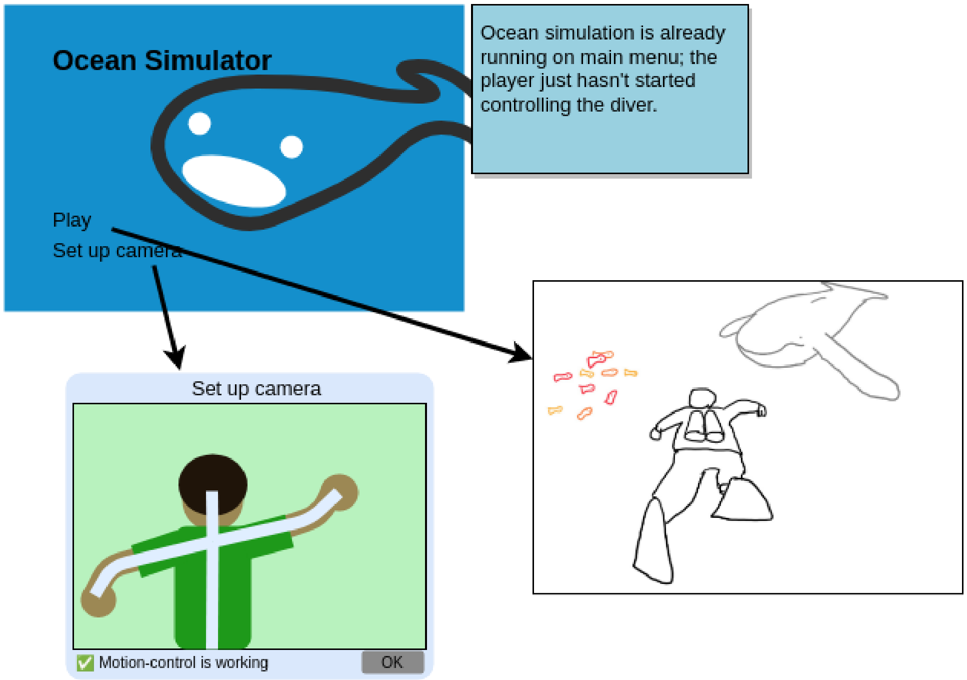

OCEAN SIMULATOR
•Multiplayer☺
• Camera observes the player's pose
• You have to "swim" to move!

Figure 1: This is a visual representation of what it looks like for a person to play the game.



Ocean simulation is already running on main menu; the player just hasn't started controlling the diver.

Figure 2: These are initial drawings of the screens that the user navigates.

Risks:

- Unrealistic schedule: The features of this project may not be developed in the short time frame
  RE: 60% x Total Project Framework Complete = 60% Project Framework
  Can be dealt with by requirements scrubbing and project estimations after each incremental delivery
- Wrong functionality: Since the actual customers cannot be fully defined, there is little domain knowledge for the project
  RE: 40% x Total Project Domain = 40% Domain Unknown
  Can be dealt with by defining actual users for the project and incorporating their feedback for each iteration into the project.
- Requirements Volatility: Changes to the project can occur based on the abilities to complete each feature successfully
  RE: 20% x Total Project = 20% Project is Volatile
  Can be dealt with by scheduling incremental deliveries for each iteration, and having well-defined features that must be implemented for the project (also make sure most features do not have minor dependencies)
- Wrong functionality: Simulation sickness and virtual reality sickness could affect players, meaning that this game could be unhealthy to play for long periods of time, and thus its core vision could be a harmful idea.
  RE: 10% x Total Project Harmfulness = 10% Project Harmfulness
  Can be dealt with by further researching simulation sickness and virtual reality sickness and testing a technical prototype of the simulator for several hours.