Costal Ecosystem Simulation.

By Mark Emmanuel k21009628 and Vaibhavkumar patel k21076223.

Contents

[Simulation Summary 2](#_Toc97134581)

[Food web 2](#_Toc97134582)

[Breeding. 2](#_Toc97134583)

[Time – Day and night. 2](#_Toc97134584)

[Plant Simulation 3](#_Toc97134585)

[Weather Simulation 3](#_Toc97134586)

[Disease Simulation 3](#_Toc97134587)

[Sources 4](#_Toc97134588)

# Simulation Summary

The Simulation is based around a costal ecosystem where we have two animals in the ocean and four animals on the land. The animals that are simulated are fishes, pigs which are the prey in the simulation. The other four which are predators are sharks, humans, pythons, and eagles.

# Food web

Foodweb Diagram

Description automatically generatedBelow is the food web diagram of the interaction made between the species. There also special cases where some animals can eat other animals if they are weak. Weak is a special attribute that is related to how old an animal is and if they have disease or not.

The colours indicate the colours of the species in the simulation for clarity. The animals are now categorized as carnivores and herbivores. This is because they both have different types of find food method and in future, they have characteristics that are shared by their respective classes.

# Breeding.

All the animals only propagate if they find a partner who is of opposite gender. This is done by checking if the animal adjacent to this animal is not of the same gender when giving birth.

# Time – Day and night.

The day and night affect both animals and plants. The animals on land all sleep during night-time and are only able to move during daytime. The sea creatures can move at night-time as well. The plants such as grass and seaweed only grow during daytime. The current length of day and night is equal and is of 5 steps.

Challenge tasks

Island generation

The simulation is about a costal ecosystem, so we wanted to implement islands as well. To do so we have used Perlin noise algorithm(check source) which outputs us a 2d array of floats. We then use this array which is random enough but still to a human eye looks natural and coherent that it is suitable to use for island generation. This algorithm takes in few parameters’ octaves, persistence, and the seed number. The octaves and persistence are set manually to create the effect of island that we were trying to aim for. The seed number is selected randomly out of 100 every time we are using this Perlin noise object. So currently the algorithm will output 100 unique different maps for the Perlin noise.

Then there is a map class which uses the noise map created by the Perlin noise and maps the value them to an integer these integers are used as an ID for our land classes which are ground(0), shallow water(1) and water(2). The Noise Map through two for loops which then check if the float value is 0.5 then in the map 2d array 1 is inputted. If the value is between 0.5 and 0.66 then 2 is inputted and lastly 0.66 to 1 float value will be mapped to 2. By doing so we ensure that ground and water will be separated by shallow water most of the times like real life. This is because the likelihood of two location next to each other have similar values to one and another.

# Plant Simulation

To implement plants, we created plant objects. These objects are on every piece of land. They can be of different types (e.g. grass and seaweed). They have an instance field called isBig which is true if the plant is big enough to be eaten. Furthermore, when this field is true for certain plants they can be shown in the GUI. For example, Grass will be shown since it is big enough however seaweed won’t be shown since it is in the ocean. At every step the plants will increase in size and grow until they are big enough at which point, they will stop growing. When they are eaten the isBig field is set to false and its size is reduced back to 0. While a plant is growing it won’t show the plant but instead the land underneath it. For example, once the grass has been eaten, it will show the ground below it. Additionally, the weather can impact the growth rate of the plants. For example, if the plant is in sunny weather, then it will grow much faster than in cloudy weather. And if it is not daytime then the plants won’t grow since there is no sunlight.

# Weather Simulation

The weather for a land will be stored in that land object. Weather, like land, uses the Perlin noise algorithm to form a map of how the weather will be. The ID for each location corresponds to a type of weather (0 – Sunny, 1 - Cloudy, 2 – Raining). The weather not only affects the plants but also the animals. The visibility is reduced the worser the weather is. For example, when it is raining visibility is very low. So, for animals with poor eyesight they are unlikely to find food and so would not move.

When weather is being shown this, it will be painted over the animals and lands. However, it isn’t opaque and therefore you will still be able to see anything underneath it. Additionally, every few days the weather map will be changed to simulate how weather changes as time progresses.

# Disease Simulation

The disease is an object that is stored within an animal. An animal can have multiple diseases at the same time. When that animal is adjacent to another animal of the same type then each disease has a chance of transmitting to the other animal. When an animal is infected with a disease then its food level drops much faster for each step.

At the start of every simulation an animal has a chance to spawn with the disease. We have a disease affecting the pigs. However, new sub classes of disease can be introduced to create new diseases that affect the other animals. Furthermore, disease can be easily updated to affects plants as well which will create a more realistic simulation.

# Sources

Algorithm Examples. 2022. *Perlin Noise Algorithm*. [online] Available at: <https://java.algorithmexamples.com/web/Others/PerlinNoise.html> [Accessed 24 February 2022].