The Coralize Tool for Creating Underwater Environments

Ryan Abela
Institute of Digital Games
University of Malta
ryan.abela.01@um.edu.mt

Antonios Liapis
Institute of Digital Games
University of Malta
antonios.liapis@um.edu.mt

Georgios N. Yannakakis
Institute of Digital Games
University of Malta
georgios.yannakakis@um.edu.mt

ABSTRACT

This paper describes Coralize, a level design tool targeting the creation of underwater environments. The user interface allows a scene designer to customize the generative algorithms of different types of corals and sponges. Moreover, the designer can draw the generated corals onto the scene, following patterns found in real-world reefs.

1. INTRODUCTION

The procedural generation of vegetation has received considerable attention, and has been often used within the game industry. Tools such as *SpeedTree* (IDV 2002) allow game companies to create realistic 3D models of trees with infinite variation in a matter of seconds, reducing the required time and developer effort of creating elaborate forest scenes.

Coralize is a tool which helps users generate corals and reef systems, and is written as a plugin for the Unity3D game engine. The main aim of this tool is to procedurally generate different 3D meshes of corals and sponges, rather than using predesigned 3D assets. Additionally, the tool provides a functionality for the designer to build a reef system out of the generated corals, following patterns of real-world reefs.

A tutorial video of the Coralize tool demonstrating its use can be found in `https://youtu.be/nG7ZoIVCO7M`.

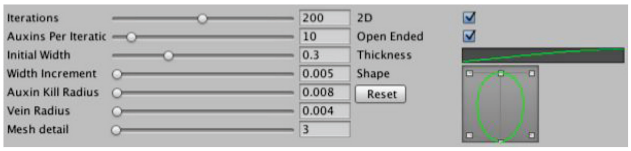
2. CREATING MARINE ORGANISMS

The user interface of Coralize allows a scene designer to customize the algorithms used to generate different types of marine organisms found in coral reefs and seabeds: in the current version of Coralize, stony corals, soft corals and sponges can be generated. By tweaking and customizing parameters of the generative processes, a scene designer has control over the visual appearance of the corals and sponges without creating results which can not be recognized as such.

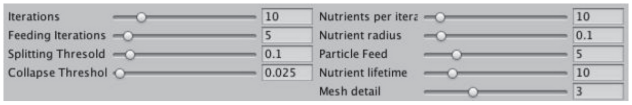
Stony corals are generated via L-system algorithms [4], which have been used extensively to replicate natural structures with some sort of self-repeating (fractal) pattern. L-systems are defined by a grammar of rewrite rules, producing

Proceedings of the 10th International Conference on the Foundations of Digital Games (FDG 2015), June 22-25, 2015, Pacific Grove, CA, USA.
ISBN 978-0-9913982-4-9.

(a) UI for customizing stony coral generation.



(b) UI for customizing soft coral generation.



(c) UI for customizing sponge generation.

Figure 1: User Interface for creating marine organisms.

a structure by applying these rules over a number of iterations. Through the graphic user interface, a scene designer can define the grammar used to generate the corals, the number of iterations (which affect the size of the coral), the thickness of the coral structure, the thickness variation and the number of edges of the generated mesh. Three grammars are currently included for stony corals in Coralize, two of which are inspired by [2]; the third one creates sea fans.

Soft corals photosynthesize in clear tropical waters, and are therefore visually more similar to plant life than the stony corals found in deeper waters. Due to this similarity, soft corals in Coralize are created via an algorithm originally used for generating the veins on leaves [3]. This algorithm works on the concept that a vein is attracted and develops towards a hormone (auxin) which is embedded in the leaf blade. Coralize allows a scene designer to customize the generative parameters, resulting in different soft coral structures. The customizable parameters are 11 in total, and include the number of iterations (which do not affect the size of the coral but do affect the curvature of its mesh), the initial width of the leaf blade and its increment per iteration, the vein radius and the shape of the structure.

Sponges have a very different structure and growth process than corals. Sponges are among the simplest multicellular organisms and obtain food, oxygen and dispose of waste through their pores. Sponge generation in Coralize is inspired by the accretive growth model of [1], although it does not recreate a fluid dynamics model due to computational constraints. The algorithm starts with an initial hemisphere

mesh which grows via nutrient particles randomly placed in the area above it; nutrients colliding with the sponge cause it to grow around the collision point, causing the porous structures common in sponges to emerge. Through the user interface, a scene designer can customize 9 parameters of the sponge generator, including the number of iterations (i.e. the number of nutrient particles and growth cycles used), the feeding iterations (i.e. the number of neighboring vertices which grow from a single nutrient), the nutrient radius and lifetime, the number of nutrients per iteration and the mesh detail of the initial hemisphere.

Using the interfaces described above, the scene designer can generate 3D meshes of corals or sponges. Once generated, the designer must specify a number of materials to the meshes, affecting their coloration. Several appropriate materials are included in Coralize, although users can also introduce their own via Unity's material editor. A scene designer can allocate more than one materials to one type of organism; when 'drawing' corals on the reef (see Section 3), one of these materials is selected at random and given to each instance of the coral — leading to variations in colors.

3. CREATING A REEF

Through the interface described above, a scene designer can create a *coral pool* which is populated with all the corals and sponges that are going to make up the reef system (along with the materials used for their colorations). This coral pool is used to place corals into a Unity scene. Apart from simply placing the corals and sponges, Coralize also offers ways of generating and distributing marine organisms in accordance to some occurrences in real-life reefs.

A scene designer must assign a game object with collision enabled to act as the *seabed*. Once assigned, the designer can drag a brush across it, and marine organisms from the pool will start being instantiated as game objects and placed on the seabed. The brush size and density of corals are customizable via the interface (see Fig. 2). While using this brush can (if so chosen) rotate and scale the marine organisms at random, the 3D meshes inserted into the scene are the same as the ones in the coral pool and have therefore no variation. Using the same meshes cuts down on design time (as no new meshes are re-generated) but also on rendering time, and provides more creative control to the designer.

In order to provide variation, an *automatic brush* allows the designer to create newly generated, unique coral and sponge meshes. When the automatic brush is dragged along the seabed, a placeholder cube is placed on the seabed while the coral is being generated in the background: when generation is completed, the cube is replaced by the coral. Moreover, the scene designer can add information about the environment which affects the growth of reefs. The current version of Coralize allows the placement of water currents, which the scene designer places as arrows on the scene indicating water flow. Since real-life sponges and corals rely on plankton and nutrients to be delivered through water streams, marine organisms placed along the current will be larger and healthier than those sheltered from it. When placing the current, the designer can specify its strength (i.e. how far away the water current influences growth) and its radius. Corals along the path of the current and within its influence have their thickness (which informs the generative process) increased. Future versions of Coralize can explore the adjustment of more parameters such as the number of iterations or the mesh detail, in order to create larger, more impressive corals along the current's path.

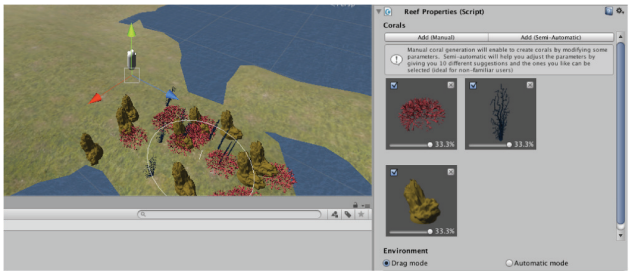


Figure 2: Placing corals and sponges on the scene from the coral pool (right) using the Coralize brush.

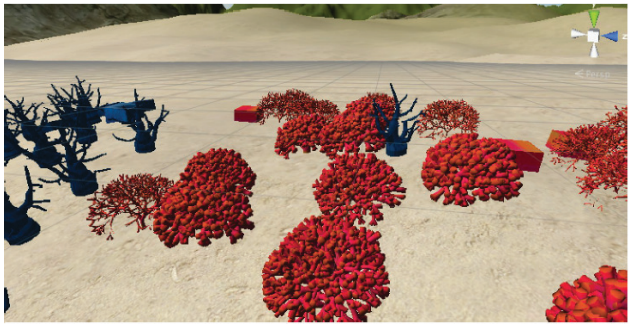


Figure 3: Corals along the path of a water current and within its influence radius have thicker 3D meshes. The scene includes some placeholder cubes from the automatic brush, which are replaced by meshes when generation completes.

4. CONCLUSION

Coralize is a Unity3D plugin which allows scene designers to build a reef system in minutes. Since the corals and sponges are generated algorithmically, diversity in appearance is ensured without the need of buying or handcrafting 3D assets and reusing them. The two different brushes enable scene designers to construct a seabed according to their needs, and added features like the water currents give a more realistic appearance to the reef.

Acknowledgements

The research was supported, in part, by the FP7 ICT project C2Learn (project no: 318480) and by the FP7 Marie Curie CIG project AutoGameDesign (project no: 630665).

5. REFERENCES

[1] J. A. Kaandorp and J. E. Kübler. *The algorithmic beauty of seaweeds, sponges and corals*. Springer, 2001.

[2] M. Meister. *Interactive Visualization in Interdisciplinary Applications*. PhD thesis, University of Erlangen-Nuremberg, 2008.

[3] A. Runions, M. Fuhrer, B. Lane, P. Federl, A.-G. Rolland-Lagan, and P. Prusinkiewicz. Modeling and visualization of leaf venation patterns. In *ACM Transactions on Graphics*, volume 24, pages 702–711. ACM, 2005.

[4] J. Togelius, N. Shaker, and J. Dormans. Grammars and L-systems with applications to vegetation and levels. In *Procedural Content Generation in Games: A Textbook and an Overview of Current Research*. 2015.