Real-time Procedural Terrain Generation Through Swarm Behaviours

[Extended Abstract]

Angel Fernandez Cabezas
School of Computing & Mathematics
University of Derby
Derby, England
afercab@correo.ugr.es

Tommy Thompson*
School of Computing & Mathematics
University of Derby
Derby, UK
t.thompson@derby.ac.uk

ABSTRACT
This paper explores research in progress on Swarm Intelligence related to Procedural Terrain Generation. Our aim is to develop means to provide useful and real-time terrain deformation using Swarm algorithms. We feel this could prove interesting for enhancing the experience of gameplay in some game genres such as Real Time Strategy (RTS), simulation or even platform games.

Categories and Subject Descriptors
I.2.1 [**Artificial Intelligence**]: Applications & Expert Systems—*Games*

General Terms
Theory

Keywords
Swarm Intelligence, Procedural Terrain Generation

1. INTRODUCTION
This work is focussed on the development of Swarm Intelligence. It is to be deployed in a 3D map created in C# using the *XNA* framework that can be modified in real time according to a series of parameters (i.e. height of the neigbour cells, distance to the point where the user has clicked or the amount of change applied to the terrain), with the intent to create useful and practical results. This is employed with the intent that the user can modify these parameters to impact the percentage of change applied to the terrain.

Up until now, Procedural Terrain Generation algorithms have been static. This means that they generate a candidate map according to some data, but the generation is seldom left in the hands of the user. The aim of this research is to provide a different experience by allowing a player or developer to modify the map on which they are playing in real time by using a GUI and the aforementioned swarm algorithms.

We feel this research could lead to interesting developments that improve vastly the experience of playing or developing content for videogames. Other possible uses for this research could be developing models for tectonic simulations: using proper physics, this research might help us understand us better the behaviour of some terrains and the impact of their changes on structures.

2. RELATED WORK
Work by Togelius et al. in [6] successfully used PTG for RTS games such as Starcraft. This method used a Multi-Objective Evolutionary Algorithm (MOEA), creating terrain as a sub-element of the main goal, which is generating a complete and playable game map that also includes positioning for gaming elements and objectives.

Their genotype modelled the standard deviations of a gaussian distribution, the coordinates x and y of the mountain peak and the height of the mountain. For genetic operators, they use probability based on mutations and simulated binary crosses, which suggests that the genetic strings of the descendants will look like either one of its parents or the other. As stated before, this gives very good results with RTS games, but because of the genotype that it uses, the style of the features of the terrains that it generates include soft and rounded looking peaks.

Work by Frade et al. in [2, 1] uses genetic algorithms to create height functions. A height function is an equation applied to the value of each vertex in a height map that already exists, producing a new height map by doing so. In this case the genotype is represented by a tree of operators, and evolves by adding, removing or substituting operators through genetic programming.

Even though they started using an interactive evolution approach, they ended up using two fitness functions: accessibility measurement (good for those terrains with lots of flat areas) and obstacle edge length measurement, which makes sure that every player finds obstacles in the map, so the game is challenging. By using three types of mutation for

*Tommy Thompson is a member of the Distributed and Intelligent Systems Research Group (DISYS).

the genotype, we end up getting a solution space that has good exploration. Another benefit from that is that the height function is prevented from being too long. The problem with these algorithms is that it outputs terrains that are too flat and have predictable patterns, so we can not use them for many types of game.

The study by Raffe et al. [4] focusses on terrain samples that are decomposed into smaller patches which are then recombined to obtain a new terrain, and for the genotype it uses a two dimensional array of patch identification numbers. For the crossover, the descendants duplicate the genetic structure of one of their parents, and then assign to every patch the probability of being switched for the corresponding patch on the other parent. For mutation, each patch is assigned the possibility of being swapped with another one chosen randomly.

It is relatively easy for a new user with little experience to generate game maps by running this algorithm a couple of times, and you can obtain a huge variety of terrains just by mixing the samples. However, these patches limit the exploration of the solution space, as it depends on those that the user has input beforehand. And it can have trouble generating terrains with shapes that have not been provided, but there is a way to solve this just by using smaller patches.

With our work, we wish to expand the experience by having dynamic surroundings. This could make gameplay more appealing, so creating a dynamic courtesy of Swarm Intelligence we hoped would be a good way to approach this problem.

3. OVERVIEW OF WORK COMPLETED

In our current work we have developed a tool that can change the shape of the terrain in real time. Only a mouse is needed to allow the user to interact with this application, so it is very intuitive and can be a good tool if applied to games with a touch screen. You can use the mouse in three diferent ways: dragging, clicking or click and drag. Each of these methods employs a different means of deforming terrain, with point and the algorithm spreads automatically depending on the height of the neighbour cells or clicking in one point and dragging to the other, which would result on the construction of a path to the other point. Utilising all of these against an intially flat terrain can yield results shown in Figure 1.

Clicking the terrain using the coordinate of the mouse as the center of a 5x5 matrix, applying changes to height to the neighbours according to their altitude towards the point the user has clicked. If the mouse button is held, the area of influence of the algorithm will increase. This allows for more customisation, we see the gradual changes to the terrain.

In our second approach, the clicking interaction, we modify the terrain using a swarm algorithm that takes a random walk through the terrain, modifying the height of all vertices that the swarm comes into close contact with. The height changes made are with respect to their neighbours. This produces a more realistic and detailed result. Finally, the third behaviour directs our swarm by focussing on the line drawn from the initial click to the point where the mouse

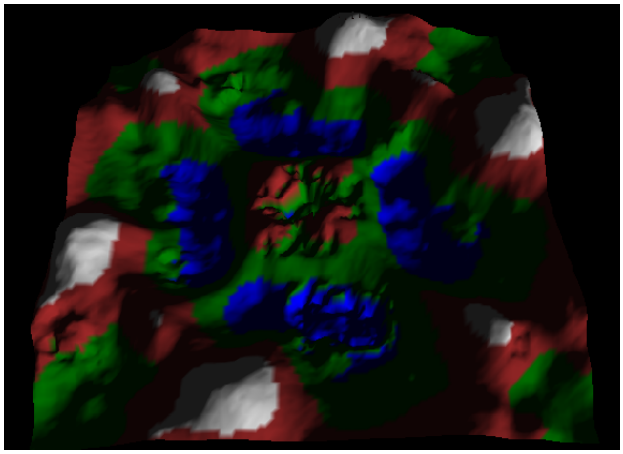

Figure 1: Final result after applying all three terrain deformation methods

button is released. This adopts the same process as before, only it now directs the changes across a linear path.

3.1 Current and Future Work

Having tested these methods work successfully, we are interested in modifying the 'dragging' methods to use geodesic distance rather than euclidean distances. We believe that employing geodesic distances will enable to modify the terrain more realistically and by extension achieve better results. In addition, we will conduct tests with users to gain feedback on the effectiveness of the current approach.

4. REFERENCES

[1] M. Frade, F. de Vega, and C. Cotta. Evolution of artificial terrains for video games based on accessibility. *Applications of Evolutionary Computation*, pages 90–99, 2010.

[2] M. Frade, F. Fernandéz de Vega, and C. Cotta. Breeding terrains with genetic terrain programming: the evolution of terrain generators. *International Journal of Computer Games Technology*, 2009, 2009.

[3] T. J. Ong, R. Saunders, J. Keyser, and J. J. Leggett. Terrain generation using genetic algorithms. In *Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 1463–1470. ACM, 2005.

[4] W. L. Raffe, F. Zambetta, and X. Li. Evolving patch-based terrains for use in video games. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 363–370. ACM, 2011.

[5] W. L. Raffe, F. Zambetta, and X. Li. A survey of procedural terrain generation techniques using evolutionary algorithms. In *Evolutionary Computation (CEC), 2012 IEEE Congress on*, pages 1–8. IEEE, 2012.

[6] J. Togelius, M. Preuss, and G. N. Yannakakis. Towards multiobjective procedural map generation. In *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, page 3. ACM, 2010.