

TERRA FORMATION CONTROL (OR HOW TO MOVE MOUNTAINS)

J R Rankin
La Trobe University

Abstract

The new Uplift Model of terrain generation is generalized here and provides new possibilities for terra formation control unlike previous fractal terrain generation methods. With the Uplift Model fine-grained editing is possible allowing the designer to move mountains and small hills to more suitable locations creating gaps or valleys or deep bays rather than only being able to accept the positions dictated by the algorithm itself. Coupled with this is a compressed file storage format considerably smaller in size than the traditional height field or height map storage requirements.

Keywords

Terrain, height map, height field, 3D rendering, fractal.

Introduction

Terrains in 3D mobile games have been overly simplistic and unappealing. With the new Uplift Model terrain generation algorithm this is set to change as it allows much more complex and interesting terrain variations with no great impact on memory or hard disk space. A major problem with previous terrain generators for game designers is the lack of control in the previous terrain generation algorithms [2]. All terrain generation algorithms involve randomness to attain a realistic look, however designers are needing a terrain generation process that has controllable randomness in positioning the mountains in the terrain and their shape and appearance of ruggedness. Recent research on editing large complex terrains has been undertaken for this purpose [3, 4]. Another though lesser problem with previous terrain generators was the storage space required to hold them which is particularly acute for large scale terrains when storage is at premium on mobile devices.

The Uplift Model of terrain generation was introduced recently [1]. It is not based on fractals but could be texture mapped with a texture derived from fractal algorithms. It generates a relatively small number of peaks forming conical hat surfaces and then blends these surfaces together to form the terrain. This paper provides a timely generalization of the model. The generalization is to allow each peak to have its own ruggedness parameter R , rather than there being only one ruggedness value for the whole terrain. This is equivalent to allowing each cone surface to have its own base radius r_p as well as its own cone height y_p .

Uplift Model Terrain Storage

A fractal terrain may be as large and as detailed as the designer needs in a game but there is a storage cost. The terrain must be stored because every time the fractal algorithm is run again a new and different terrain is generated. A medium sized 1000x1000 mountainous game world

would be stored as 4 MB or more. By contrast with the Uplift Model we only need to store the peak data – the number of peaks and their central grid indices. Typically the number of peaks needed for a realistic terrain is around 100 for a 1000x1000 game space which results in less than 2 KB for file storage. Apart from the peak data however a few parameters for the Uplift Model should also be stored in this file. The format used for storing Uplift Model terrains is called the UPD format. It is a space separated text file where the first line is two positive integers: N_x , the number of grid lines along the x-axis and N_z , the number of gridlines along the z-axis. The second line consists of two floating point numbers: Δ_x , the distance between grid line divisions along the x-axis and Δ_z , the distance between grid line divisions along the z-axis. The third line contains only the integer N_p the number of peaks generated. Following this are N_p lines each containing the data i_p, j_p, y_p and r_p for a peak which is centred at the grid position (i_p, j_p) , the height of the peak is y_p and its base radius is r_p . Each cone in this generalized Uplift Model can contribute a different ruggedness parameter = y_p/r_p – the smaller r_p is and the larger y_p is the greater the ruggedness of that peak. There is no need to store a parameter R , the model's ruggedness parameter as $R = y_p/r_p$ will in general be different for each peak instead of being forced to be the same value across all peaks of the terrain. (This generalization does not preclude a terrain from having a single ruggedness value R for all peaks of the terrain however.) When this data is read in from file to the program, the Uplift Model algorithm converts it into a predictable but complex and realistic looking terrain such as seen in Figure 8.

Uplift Model Editor

An intuitive and easy way to edit an Uplift Model terrain is based on a 2D plan view of the terrain area. The terrain base grid is viewed and each peak centre is represented by a green marker placed on the grid at its centre point (i_p, j_p) . As the peak is represented by a cone of height y_p standing on the ground, in the plan view the editor displays a circle of radius r_p for the peak. The circle has centre (i_p, j_p) and radius y_p/R and is the circle at the base of the vertical cone. The editor allows the designer to scale up the plan view as well as scroll it inside the editor window left and right or up and down so that an adequate detailed view is available to the designer for working with. A sample plan terrain view in this editor is shown in Figure 1 below. To edit the terrain the designer only needs to click the mouse near to a circle of influence to select that circle which will then appear highlighted in the colour red as shown in Figure 1 below. Once a circle is selected the designer can move its centre from grid point to grid point over the terrain base grid by pressing the 4 arrow keys thereby moving the peak (and its highlighted circle) to a new position. In this way the designer can pick circles and move them to rearrange the whole terrain. The designer can also move peaks to negative grid positions or anywhere off the base grid always moving by increments of $\pm\Delta_x$ and $\pm\Delta_z$ in the x and z directions. In the editor the z direction is from top to bottom of the window to correspond to the 3D plan view down the y-axis towards the origin in the negative y direction. Also note that the origin $x = 0, y = 0, z = 0$ is in the centre of the editor window. It does not matter if peak centres are off the terrain base grid as their influence can still effect base grid sites so long as its circle intersects the terrain base grid and thus the terrain heights at other base grid sites.

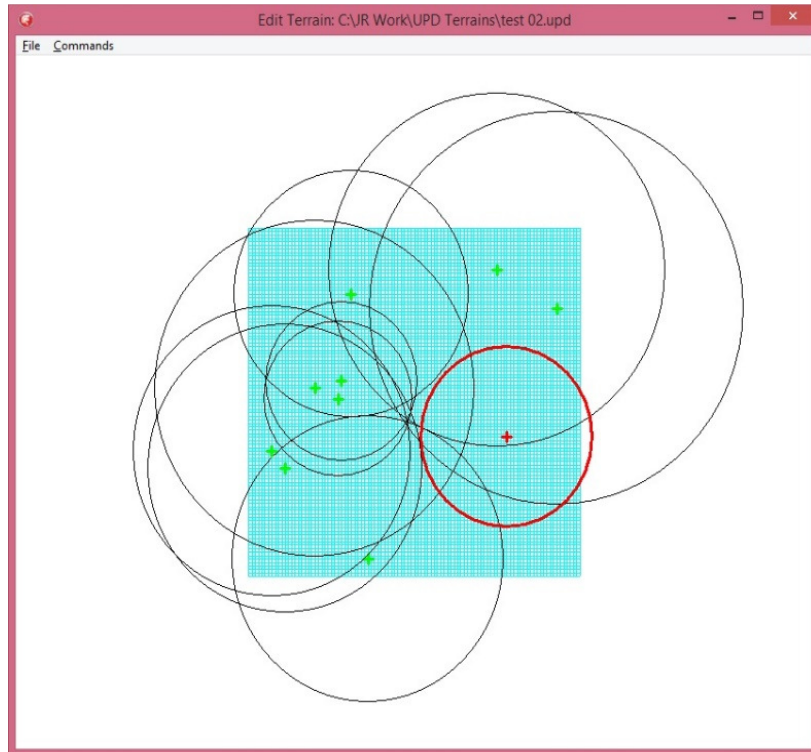


Figure 1. The plan view of a terrain in the Terrain Editor with only 10 peaks one of which is selected.

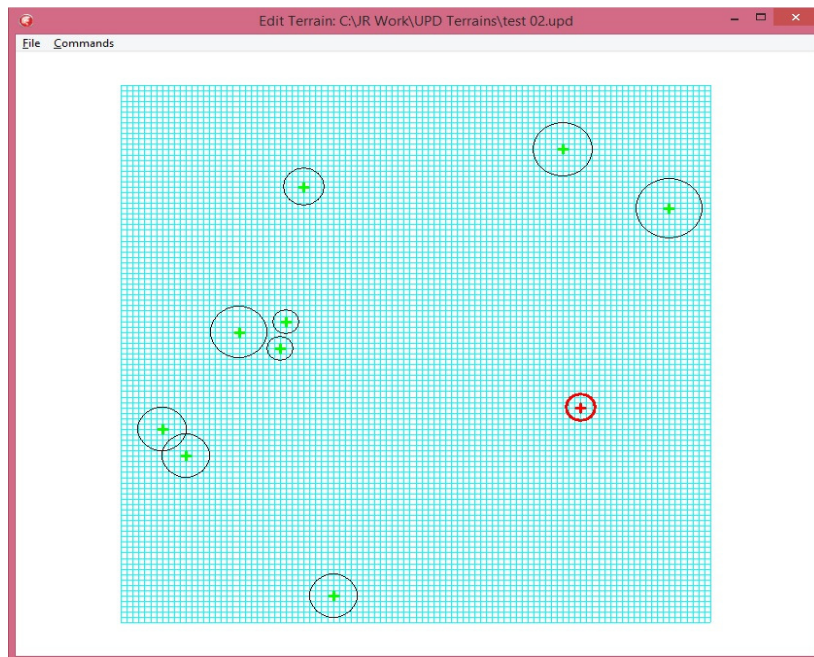


Figure 2. The same terrain with scale factor 0.1 for the peaks and the terrain base zoomed in.

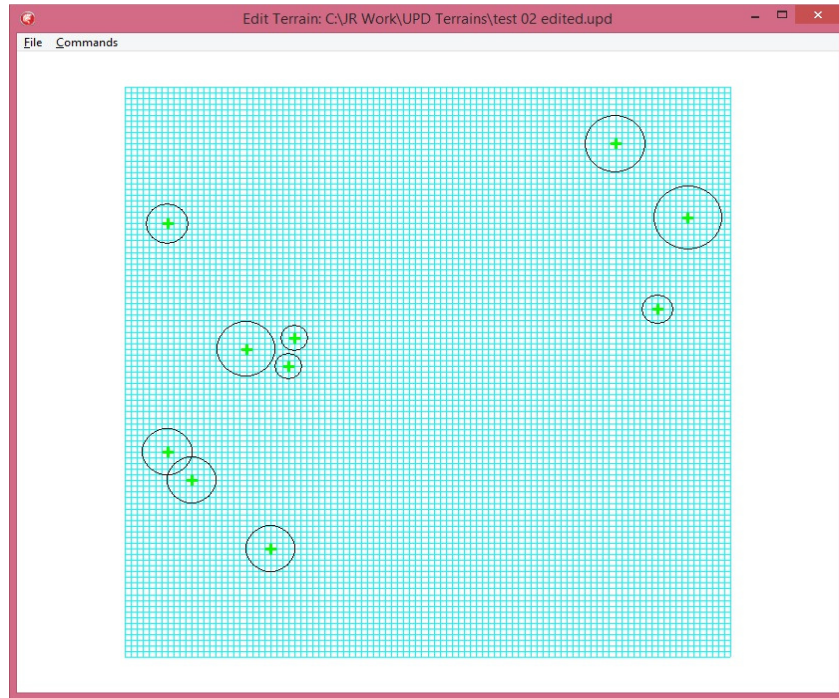


Figure 3. The selected circle has been moved to result in a major change in the terrain design.

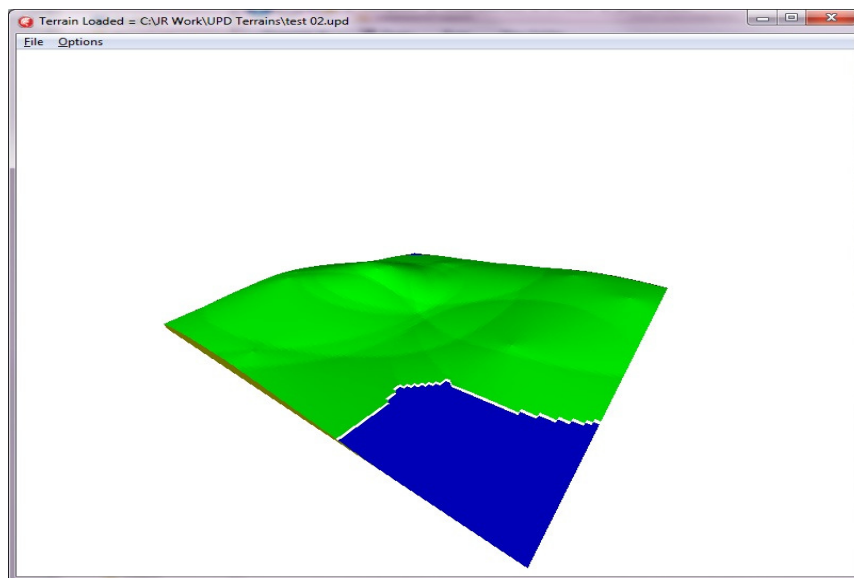


Figure 4. A 3D view of the terrain before editing.

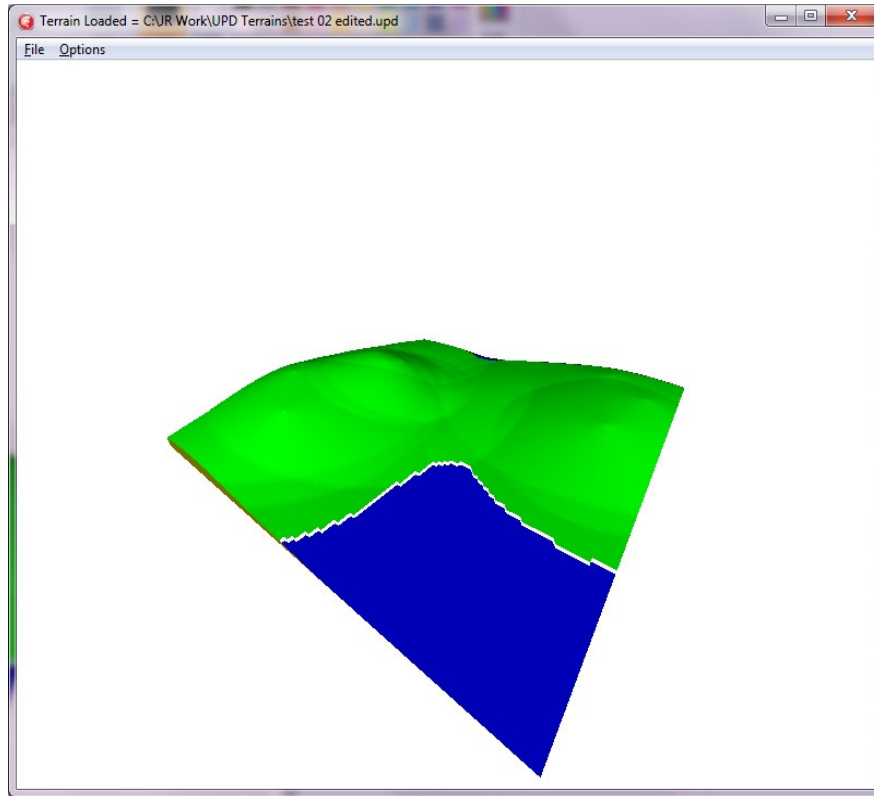


Figure 5. A 3D view of the terrain after editing showing the increased size of the bay.

With larger terrains, larger numbers of peaks and higher peaks it was found that there is a confusion of circles confronting the designer and many interesting terrains had circles of influence larger than the base grid rectangle as seen in Figure 1 above. The method used to make the designer's job easier here was to allow the designer to enter a scale factor which reduces the radii of all peak base circles. When the circles were mostly non-overlapping it became much easier to select and move the peaks and thus rearrange the mountain ranges. Another approach is to sort the peaks into height order and allow the designer to select k the number of largest peaks to display. In this approach only large peaks will be displayed (the k largest) and smaller peaks are not displayed as they would clutter the editing screen. After editing a UPD file the file is written back to disk (with a different file name).

An example of editing a small terrain is shown in Figures 2 and 3 where only one peak was moved. The corresponding 3D terrain views are shown in Figures 4 and 5. It can be seen that the effect of the designer's editing is to increase the size of the bay in the foreground. An example where a mountain is moved to generate a new valley is shown in Figures 6 to 9 below. When a peak is selected as in Figure 6, the designer can click on the main menu under Properties to view the peak's properties i_p , j_p , y_p and r_p and the peak index. The designer can change these property values in the pop up dialog box thus giving the designer fine-grained editing control over the terrain. This editing was done on two peaks in Figure 6 resulting in the terrain plan of Figure 7. When the two UPD files were displayed as 3D terrains we see that the tallest mountain (indicated) has moved leftwards opening up a valley down to an enlarged bay – see Figures 8 and 9.

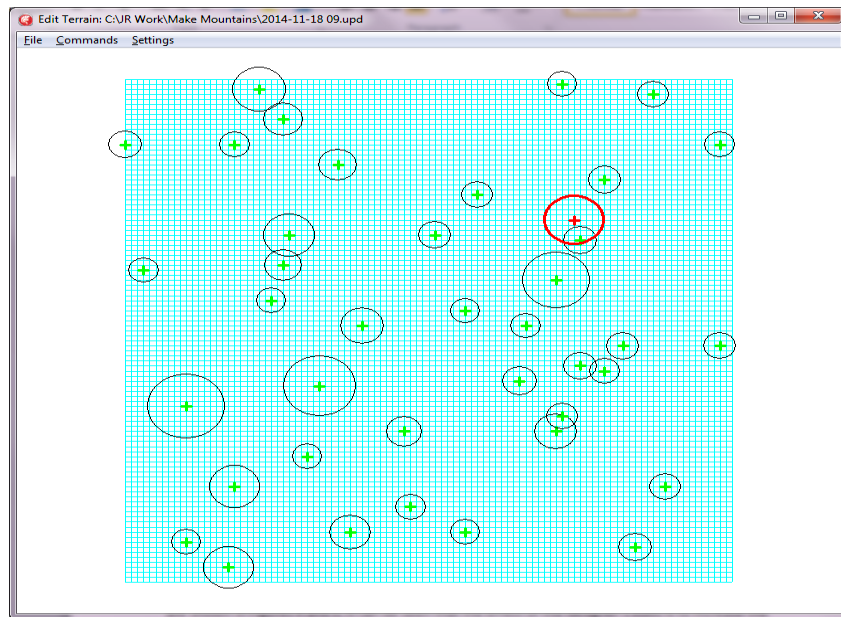


Figure 6. The selected cone (number 18) has $y_p = 586.6$ and $r_p = 29.3$.

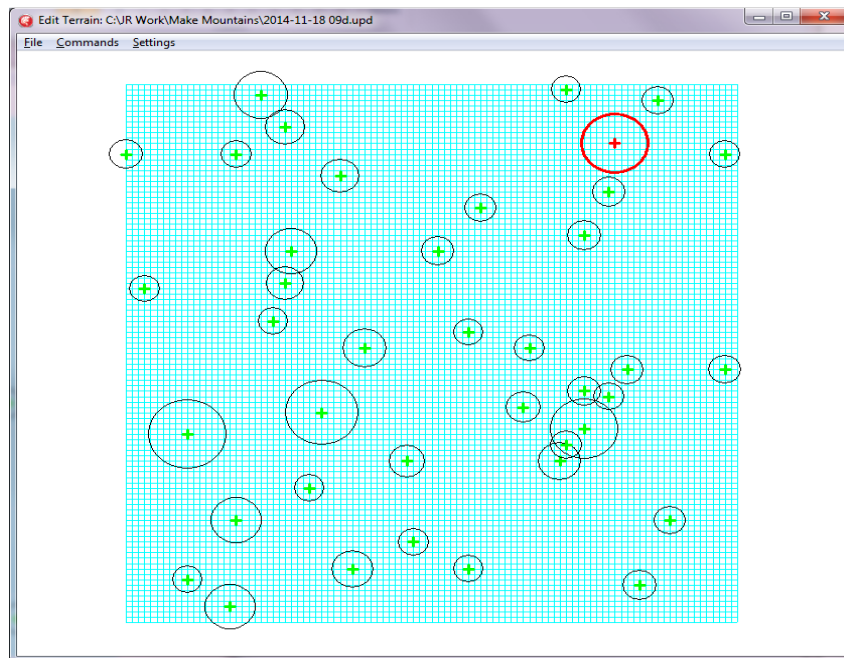


Figure 7. Cone number 18 is moved and now has $y_p = 674$ and $r_p = 33$.

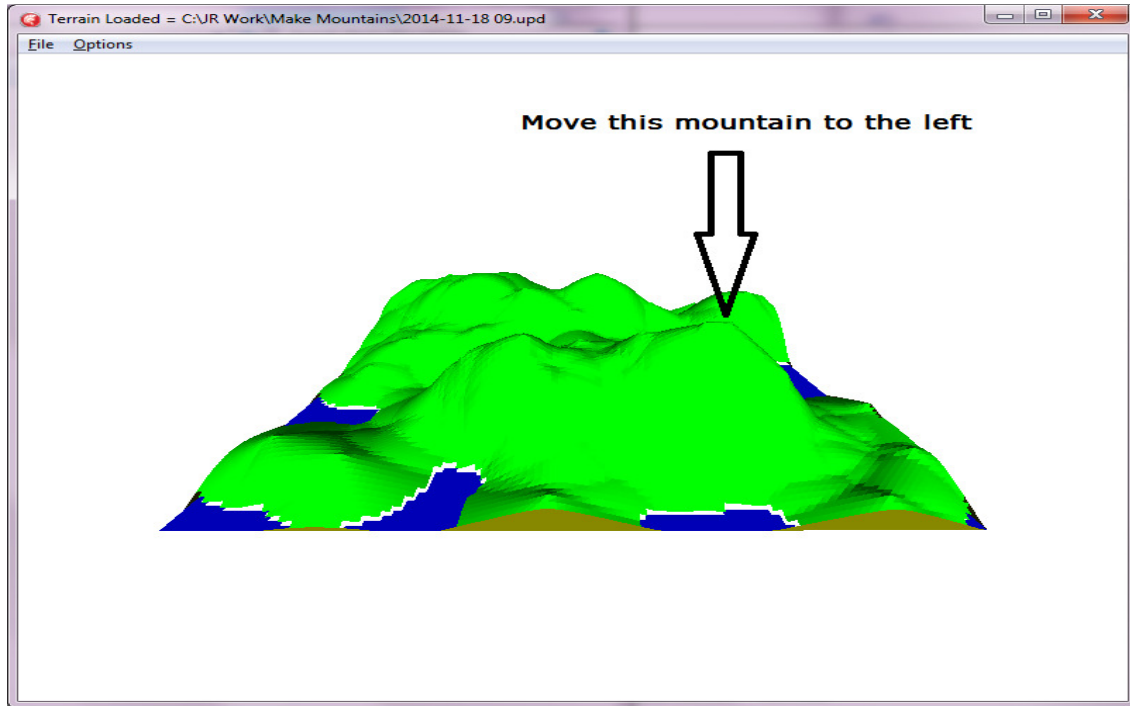


Figure 8. The randomly generated scene before editing.

Conclusions

The generalized Uplift Model makes terrain generation controllable by means of visually editing the associated UPD file in plan view. The process is intuitive and allows game designers and digital artists to more accurately control the appearance of the random game terrain according to their game needs. The generalized model allows the one terrain to contain a mixture of terrain types or zones depending on the distribution of peak ruggedness R . For example a generalized terrain can contain a zone of rolling hills from peaks with ruggedness $R = 1$ and a zone of high mountains from peaks with ruggedness $R = 20$.

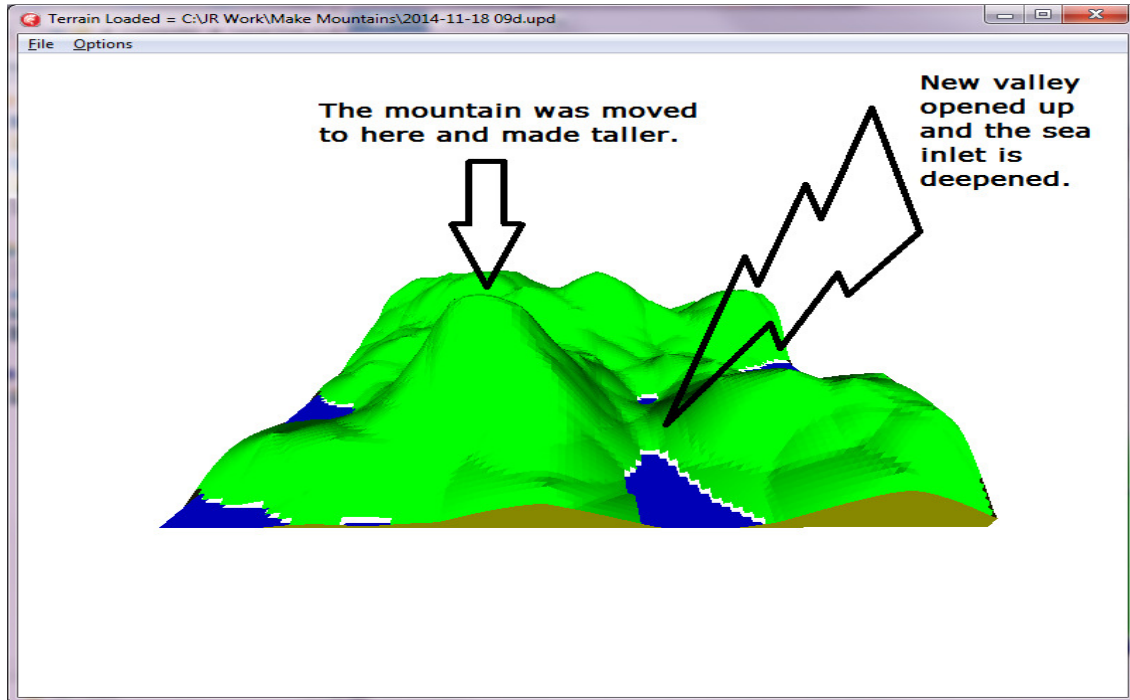


Figure 9. The same scene after editing to move a mountain left and open up a valley.

References

- [1] Rankin J. "The Uplift Model Terrain Generator", International Journal of Computer Graphics and Animation, Nov 2014.
- [2] Raffe W.L., Zambetta F. and Xiaodong L. "A Survey of Procedural Terrain Generation using Evolutionary Algorithms", IEEE World Congress on Computational Intelligence, WCC! June 2012, Brisbane, Australia
- [3] Li D, Hu Y, Meng D, Luo X, "The Research And Implementation of Interactive Terrain Editing and Crack Elimination", Computational Intelligence and Software Engineering (CiSE) 2009 pp1-4.
- [4] Vanek J, Benes B, Herout A, Stava P, "Large-Scale Physics-Based Terrain Editing Using Adaptive Tiles On The GPU", Computer Graphics and Applications IEEE vol 31, issue 6, pp 35-44, Nov 2011.