**International Journal of Natural Sciences Research**
*2015 Vol. 3, No. 2, pp. 21-29*
*ISSN(e): 2311-4746*
*ISSN(p): 2311-7435*
*DOI: 10.18488/journal.63/2015.3.2/63.2.21.29*
*© 2015 Conscientia Beam. All Rights Reserved.*

# THREE-DIMENSIONAL MODELING OF THE VOLCANO BANDAI USING SHAPE TEXTURE

**Sergey I. Vyatkin**[1]

[1]*Institute of Automation and Electrometry, SB RAS, Novosibirsk, Russian Federation*

## ABSTRACT

*This paper reports a three-dimensional modeling of the volcano Bandai near The Aizuwakamatsu City. Aizuwakamatsu is located in the western part of Fukushima Prefecture, in the southeast part of Aizu basin. A volcano model is coded as multilevel height map. To visualize the terrain, tessellation is not required. During the recursive voxel subdivision on each level, voxels are projected onto the base surface (plane). The altitude corresponding to this address and a level of details are calculated, and use it to modify coefficients of the plane equation. As a result was obtained a terrain surface modulated with the values from the altitude map.*

**Keywords:** Shape texture, Scalar perturbation functions, Height maps, Voxel-based terrain, GPU.

**Received:** 4 December 2014/ **Revised:** 13 April 2015/ **Accepted:** 21 April 2015/ **Published:** 27 April 2015

## Contribution/ Originality

This study uses a new technique for rendering terrain. The computation time in terrain generation is practically independent of the height map resolution. The computed coordinates, as well as in the case of ordinary RGB texture map, will define address in the so called "altitude map" or "shape texture".

## 1. INTRODUCTION

Terrain visualization is a difficult problem for applications requiring accurate images of large datasets at high frame rates. Many systems use regular terrain elevation grid with square cells. It leads to algorithmic simplicity of computations, database uniformity, and strict definition of relationship between adjacent levels of details (LODs), which results in database generation simplification. On the other hand regular grid obviously involves information excess when considering the number of grid posts. Most novel real-time visual systems have terrain skinning processors (TSP), which guarantee continuous LODs change with respect to surface roughness and viewpoint distance. One of the main reasons to incorporate TSP in the visual system is its ability to generate terrain skin with low depth complexity (near 1) and hence reduce the image generator (IG) load when compared to traditional including terrain polygons in an environment database and LOD switching. TSP could be either an installable hardware device, or software process executed by geometry processor (GP). If TSP is an application specific device having local processor then it might be able to generate in real-time over 1000.000 terrain triangles when using regular grid and which is a considerable part of the IG system performance. This simulates search for a terrain skinning methods of IG unloading at the cost of more heavy TSP load, and particularly non-girded terrain. Regular grid could be square, or hexagonal, or triangular and so on. For Cartesian coordinate system most natural is square grid with axis collinear cell sides and grid aligned origin. Let us call this grid regular and all other – irregular. "Most" an irregular is grid with randomly spaced nodes. Others could have other kind of irregularity: square grid rotated, or shifted relatively to coordinate system, or with independently shifted nodes. Some properties of these grids are "regular", and we can use them for irregular grids evaluation. Generally, to achieve higher compression it takes more processing resources (time, memory), including decompression. In our case compression factor is a number of IG input triangles under irregular grid model with respect to that of regular grid terrain model.

Regular grid has fixed sampling rate for each LOD. In this case LODi is a set of triangles which approximates terrain so as the maximum error is not higher than appropriate constant Ei:

$$Emax < Ei \qquad (1)$$

Irregular grid has fixed bandwidth, and LOD with the same maximum error could have fewer nodes in this case. Here LODi is represented by set of triangles, organized in clusters so as for each of them maximum approximation error is between current LOD maximum and next LOD maximum:

$$Ei+1<Emax<Ei \qquad (2)$$

Irregular grid bandwidth wideness has two consequences. Each LOD can have "built-in" (implicit) surface roughness [1] and therefore database volume could be reduced.

For regular grid the maximum amount of memory also could be specified, because of limited mountain's height. If this limit is 300m for 10m space frequency then regular and irregular grids will introduce the same error at regular cell size 8 times less than that of irregular. The later gives us a difference in triangles number about 64 times. This is in accordance with Scarlatos and Pavlidis [2].

Rendering photo-realistic, complex terrain features at interactive rates requires innovative techniques. A polygonal model and geometric pipeline can be used but this introduces massive storage requirements and, ideally, a parallel implementation of the algorithm. However, features with high spatial frequency context (ridge lines and canyons) require large numbers of polygons to meet a specified level of terrain accuracy. Using traditional polygonal representation for the example complex surface give rise to a range of problems such as visible surface determination, depth complexity handling, controlling levels of details, clipping polygons by viewing frustum, geometry transformations of large number of polygons [2-9].

Numerous methods for rendering height-based terrain surfaces have been developed [10]. Databases for terrain use DEM (digital elevation model) models. This standard is designed by U.S. Geological Survey and, on essences, is a table of heights terrain with counting out through

© 2015 Conscientia Beam. All Rights Reserved.

7.5 or 15 minutes. DEM model consists of two files, binary file of data in which recorded heights in the manner of 16- bit fixed numbers, and head file which describes a format of record of numbers used in the file of data (BigEndian or SmallEndian ), but in the same way area on terrestrial surface which describe heights in the file of data. The continuous level of detail algorithm takes a two-part approach in which terrain is first divided into blocks for which a detail level can be selected at a coarse granularity [11]. The real-time optimally adapting meshes algorithm builds upon the algorithm [11] by organizing terrain meshes into a triangle bintree structure [12]. Geomorphing to the continuous level of detail algorithms described in Rottger, et al. [13]. The progressive mesh technique was extended to height-based terrain, and it enables smooth view-dependent terrain rendering with geomorphs [14].

Known non-polygonal methods of photorealistic relief visualization are quite slow. Attempts to increase speed by different types of acceleration methods (hierarchical [15], parametric [16], a massively parallel computer [17] or special parallel ray-casting hardware [18], hybrid ray-casting and projection technique [19] improve the situation, but still do not achieve real-time speed for high performance terrain visualization.

In order to render voxel-based terrain, proposed method must be able to convert a 3D scalar field representing the terrain into a set of vertices and triangles that can be rendered by the graphics hardware.

A method for constructing a triangle mesh whose vertices coincide with the zero-valued isosurface is the Marching Cubes algorithm [20]. Although it provides many greater capabilities, the use of voxel-based terrain in real-time virtual simulations also introduces several new difficulties. The algorithms used to extract the terrain surface from a voxel map produce far greater numbers of vertices and triangles when compared to conventional 2D terrain. The development of a seamless LOD algorithm for voxel-based terrain is vastly more complex than the analogous problem for height-based terrain. Texturing and shading of voxel-based terrain is more difficult than it is for height-based terrain. In the cases that triangle meshes are generated for multiple resolutions, arises the cracking problem. A method for patching cracks on the boundary plane between cells triangulated at different voxel resolutions was described in Shu, et al. [21].

Using a voxel-based model [22], however, can achieve the same results at a much lower hardware requirement. As a software solution, the method is portable so it can be integrated into any flight simulation system regardless of hardware architecture.

This paper describes results of some investigations concerned with modeling of a volcano Bandai in which it is proposed to use voxel-based terrain without triangulation. The possibility of high performance terrain visualization is investigated. Terrain is represented for the base of scalar perturbation functions [23]. The geometric model is based on non-polygonal representation, however it does support traditional objects, and i.e. polygonal models could also be reconstructed and visualized. Volume oriented rasterization algorithm and uniformity of object processing result in an efficient hidden surface removal and detection of spatial collisions. Chosen representation of terrain data is based on regular multi-level elevation map complemented with

© 2015 Conscientia Beam. All Rights Reserved.

levels of detail. This approach has several advantages (rapid generation and modification, efficient data storing and retrieving) over polygonized terrain models.

## 2. NON-POLYGONAL TERRAIN REPRESENTATION

The open simply connected set of points on the plane a domain of the plane was introduced in Vyatkin [23]. Let *D* be the plane domain and $\overline{\overline{D}}$ its closure. Let's enter the coordinate system (u, v) on the plane. Let x, y, z be the rectangular Cartesian coordinates of the points in the 3D Euclidean space $E^3$. Prescribe three continuous functions on the set $\overline{D}$:

$$x = \varphi(u,v),\ y = \psi(u,v),\ z = \chi(u,v), \tag{3}$$

Further assume that functions (3) have the following property. If $(u_1, v_1)$ and $(u_2, v_2)$ are different points of the set $\overline{\overline{D}}$, then $M_1(x_1, y_1, z_1)$ and $M_2(x_2, y_2, z_2)$ of the space $E^3$, whose coordinates were calculated by formulas (3), are also different:

$$\begin{aligned} x1 &= \varphi(u1,v1),\ \ y1 = \psi(u1,v1),\ \ z1 = \chi(u1,v1), \\ x2 &= \varphi(u2,v2), y2 = \psi(u2,v2),\ z2 = \chi(u2,v2), \end{aligned} \tag{4}$$

The set *S* of the points *M* (*x,y,z*) whose coordinates *x*, *y*, *z* are defined by (3), where the functions $\varphi, \psi, \chi$ in the closure $\overline{D}$ of the domain *D* possess the described property, is called a simple surface. The simple surface that is a plot of the function defined in the 3D space $z = f(x, y)$is referred to as the freeform surface *F*. The terrain representation based on the scalar field is a totality of a base surface *P* (in the same coordinate system as *F*) and the related altitude map. Any surface may used as the base surface, however, surface used in practice are simple surfaces such as planes, ellipsoids, or cylinders. The altitude map is a 2D rectangle called hereafter a perturbation domain $D_P$ of the base surface *P*, and the perturbation function *h*(*u,v*) is given inside this rectangle. The altitude map in turn determines the perturbation. The domain of *h*(*u,v*) is $D_{h(u,v)} = \{U, V\}$, where *U* and *V* are the size of the rectangle. The altitude map and the base surface are related as follows: there exists a transformation $G(\Re^3 \Rightarrow \Re^2)$ from the coordinate system of *F* and *P* to coordinate system of the map. This transformation is usually a parallel projection. The value of $h(G(d_F))$ characterizes the deviation of the point $d_F$, on the surface *F* from the point *dp* that is the projection of this point onto the surface *P*. In other words, the value of *h* (*G*($d_F$)) is equal to the scalar of vector

$$\vec{v} = (\vec{d}_F - \vec{d}_P) \tag{5}$$

Therefore, the domain of the terrain can be defined as a set of point in $\Re^3$, which are defined by the vector equation

$$\vec{F} = G(\vec{v}) + \vec{n} \cdot h(G(\vec{v})); \forall \vec{v} \in \Re^3, \tag{6}$$

where $\vec{n}$ is the normal to the base surface.

If the vector $\vec{v}$ is outside the perturbation domain, the vector $\vec{n} \cdot h(G(\vec{v}))=0$ and $\vec{F}$ is the vector on the base surface. Thus, for prescribing the form of the perturbing surface we can use a table of numbers, and the function *h* can be represented by a function of interpolation by pivotal

© 2015 Conscientia Beam. All Rights Reserved.

values taken from the table. In this case, we may assume that a scalar field is given in the perturbation domain $D_p$. The function $h$ has the form:

$$h(u,v)=f_0+(f_1\text{-}f_0)(v\text{-}m\_v), \qquad (7)$$

where

$$f_0 = (1\text{-}(u\text{-}m\_u))\ \text{table}\ [m\_u,m\_v\ ])+(u\text{-}m\_u)\text{table}[m\_u+1][m\_v],$$

$$f_1 = (1\text{-}(u\text{-}m\_u))\text{table}[m\_u,m\_v+1])+(u\text{-}m\_u)\text{table}[m\_u+1][m\_v+1],$$

where $m\_u$ is the integer part of u, $m\_v$ is the integer part of v, and table[$m\_u$][$m\_v$] is the $m_u$th and $m_v$th elements of the table.

The terrain *F* based on the scalar field is specified by means of the surface and the perturbation function (the table of numbers, which characterizes the deviation of the surface *F* from the base one at check points). This paper considers representation of volcano Bandai based on the base planes. In this case, the transformation *G* is a parallel projection directed oppositely to the normal vector of the base plane. We will use the notion of the terrain *F* as a combination of the base planes and the perturbation domain; it may have a rectangular contour or be defined by vector equation (6).

## 3. RENDERING METHOD

It is proposed to describe complex geometric objects by defining (in the scalar form) the second-order function of deviation from the basic surface or (in the simplest form) from the basic plane [23]. A terrain is a particular case of such objects; it is defined by means of the basic plane and the perturbation function defined in an infinitely long parallelepiped. Values of the perturbation function are specified at the parallelepiped cross-section by a 2-D height map. As a basic surface we may use a plane, and then the direction of the carrier plane normal must match the longitudinal direction of the parallelepiped - the region of perturbation function definition.

Since during rendering it is necessary to estimate the maximum function on a three-dimensional or one-dimensional interval, then maps of the level of detail are preliminary composed for efficient calculation. The initial data form the level n if the array dimension is $2^n$ x $2^n$. Data for the level n-1 are obtained by choosing a maximum from four adjacent values of the level n, the rest three values are not considered further, i.e., we obtain a $2^{n-1}$ x $2^{n-1}$array. The zero level consists of only one value, that is, the maximum all over the height map.

While determining the perturbation maximum, the characteristic size of the current interval projection is calculated, this governs the level of detail. A cruder approximation of the initial function is chosen for a larger interval. If a more accurate representation is required, then we perform bilinear or bicubic interpolation of values of heights from the last level of detail. Therefore a terrain model is coded as differential height map, i.e. the carrier surface is defined by algebraic means and only deviation from this basic surface is stored in the each node. Such a modeling method simplifies creation of smooth detail levels and shading. The data of height grid is not subject to geometry transformations as the triangle vertices do. The geometry transformations are only required for the carrier surface. During the recursive voxel subdivision on each level, the centers of the voxels onto basic plane are projected.

© 2015 Conscientia Beam. All Rights Reserved.

The computed coordinates, as well as in the case of ordinary RGB texture map, will define address in the so called "altitude map" or "shape texture". The altitude corresponding to this address and a level of details is calculated, and use it to modify coefficients of the plane or quadric equation. As a result will be obtained a smooth surface of arbitrary shape modulated with the values from the altitude map. But the problems solved by this algorithm require much more complicated methods within the traditional approach. Indeed, the common way to present terrain with polygons requires an abundance of polygons. Besides, the number of additional problems arises such as high depth complexity, hidden polygons removal, priorities, switching between levels of detail, clipping polygons by the pyramid of vision, etc. such problems do not appear in the proposed method. It is proposed to describe terrain by defining the function of deviation from the basic plane. Terrain is constructed on the basis of plane. Terrain is a composition of the basic plane and the perturbation functions f'(x,y,z) = f(x,y,z) + *r*(x,y,z), where *r*(x,y,z) is the scalar perturbation function. Put another way, r is a number computable by projection given voxel on the height map. Let's find coordinates of univariate bar - voxel $v_0$, which will be assigned pair vectors

$$P_0=(X0,Y0,Z0) \text{ AND } P_1=(X1,Y1,Z1),\ V_0=\{P_0,P_1\}. \qquad (8)$$

Further, coordinates of voxel $V_0$ by means of transformations G are converted in coordinate system height map:

$$\{(x0,y0,z0),(x1,y1,z1)\} \Rightarrow \{(u0,v0,h0),(u1,v1,h1)\}. \qquad (9)$$

Using the transformation matrix T in the height map coordinate system, which being multiplied to the matrix of geometric transformation M and gives a resulting matrix of transformation G. G=T*M;

$$T=\begin{bmatrix}1\,0\,0\,0\\0\,0\,1\,0\\0\,1\,0\,0\\0\,0\,0\,1\end{bmatrix} \qquad (10)$$

Then, voxel transformed coordinates *(u, v, h, a)* in coordinate system of height map are calculated from *(x,y,z)* voxel coordinates in model space by multiplying a vector of point in model space to matrix G.

$$G\begin{bmatrix}x\\y\\z\\1\end{bmatrix}=\begin{bmatrix}u\\v\\h\\a\end{bmatrix} \qquad (11)$$

Further, voxel subdivision on Z coordinate (or binary voxel subdivision) is used. At this stage, for the current level of recursion end vector of voxel being nearest with respect to the observer, is supposed equal nearest end vector of voxel preceding level subdivision. Far-away vector of voxel is calculated as a semi-sum of vector amount of near and far-away voxel preceding level subdivision.

$$P_{n\,i} = P_{n\,i-1},\ P_{f\,i} = (P_{n\,i-1}+P_{f\,i-1})/2\ .V_i = \{P_{n\,i}, P_{f\,i}\}, \qquad (12)$$

© 2015 Conscientia Beam. All Rights Reserved.

where $V_i$ is a voxel of i –level of recursion, $P_{n\,i}$ $P_{f\,i}$ is the coordinates of near and far-away voxel of i-level subdivision.

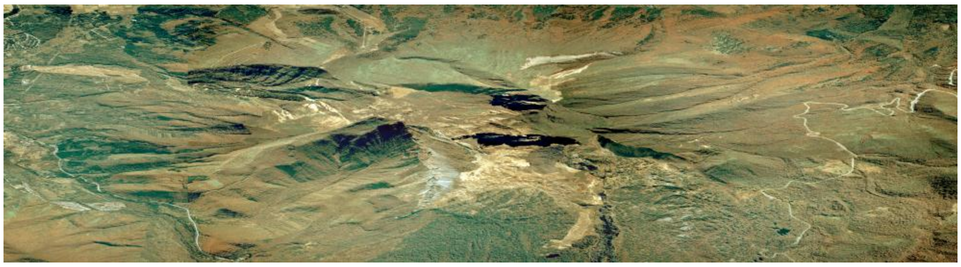

**Fig-1.** Volcano Bandai: top view.

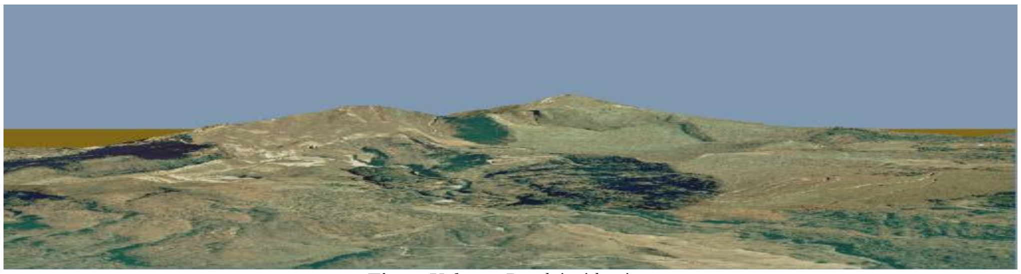

**Fig-2.** Volcano Bandai: side view.

By sizes of voxel projections corresponding recursion level is calculated level of detail. By u and v coordinates of points $P_{n\,i}$ and $P_{f\,i}$ is realized sample of maximum value from table presenting given level of detail. Calculated therefore number is a value of perturbation function of base plane.

On each stage voxel subdivision on its sizes is calculated level of detail. If level of detail not last level, then calculated height h is compared to value of height of given level Hmax, and if h>Hmax, then voxel subdivision stops.

1. define size of rectangle being voxel projection on the height map as a maximum of distance from the point {u0,v0} to the point {u1,v1 }-Lp;
2. from the inequality $\frac{1}{2^{level\ l}} < L_p < \frac{1}{2^{level\ +1}}$ define a level of detail ('level').

Figures 1 and 2 show a result of voxel-based terrain modeling without preliminary triangulation with bilinear interpolation of height values (height map resolution – 200x200),

## 4. IMPLEMENTATION AND PERFORMANCE

In our work two applications which visualize the volcano Bandai based on scalar perturbation functions have been realized. The first uses only CPU for calculations. The second uses GPU for calculation of depth, normal and illumination, and CPU for geometric transformations. For image display both versions used DirectX. Testing of productivity of the offered variants of realization has been made. Compute Unified Device Architecture (CUDA) from NVIDIA was used. CUDA is a model of parallel programming. Together with a set of software, she allows to realize programs in language C for execution on a graphics accelerator. Testing was performed on the processor Intel Core2 CPU E8400 3.0 GHz, GPU 9800 GT и 470 GTX. Test results are shown in Table 1.

© 2015 Conscientia Beam. All Rights Reserved.

**Table-1.**

| Height map resolution | E8400 | 9800 GT | 470 GTX |
|---|---|---|---|
| 128x128 | 802,65 milliseconds | 177,03 milliseconds | 21,09 milliseconds |
| 512x512 | 2033,6 milliseconds | 368,9 milliseconds | 42,51 milliseconds |
| 1024x1024 | 3275,6 milliseconds | 680,46 milliseconds | 72,35 milliseconds |

The main conclusion is that the computation time in terrain generation is practically independent of the height map resolution (See Table 1).

## 5. CONCLUSION

The main merits of proposed approach are the following: reduction of the load on the geometry processor and decrease of data flow from it to the render processor; the geometry processor works with the single basic plane; the right priority order is provided by the corresponding traversing of the tree and the set of masks; sufficiently simpler construction of terrain because the preliminary surface triangulation and the viewing pyramid clipping are unnecessary (to change the level of detail we use a mechanism similar to the usual texture sampling); the computation time in terrain generation is practically independent of the height map resolution and depends only on the screen resolution. Rendering method, described above, uses a graphics accelerator for most of the calculations. We can use parallel calculations in GPU to accelerate rendering. We successfully integrated proposed visualization method into the standard rendering pipeline. Verify the performance for the different scenes. For considered tests the application with GPU average ten times faster, than the version using only CPU.

Funding: This study received no specific financial support.

Competing Interests: The author declares that there are no conflicts of interests regarding the publication of this paper.

## REFERENCES

[1] R. Ferguson, R. Economy, W. Kelly, and P. Ramos, "Continuous terrain level of detail for visual simulation," in *Proceedings of Image V Conference*, 1990, pp. 144-151.

[2] L. Scarlatos and T. Pavlidis, "Adaptive hierarchical triangulation," in *Proceedings of Auto-Carto 10, Baltimore, MD*, 1991, pp. 234-246.

[3] L. L. Scarlatos, "Adaptive terrain models for real-time simulation," in *Proceedings of The Digital Electronic Terrain Board Symposium, Wichita, KS*, 1989, pp. 219-229.

[4] L. L. Scarlatos, "A compact terrain model based on critical topographic features," in *Proceedings of Auto Carto 9, Baltimore, MD*, 1989, pp. 146-155.

[5] L. Scarlatos and T. Pavlidis, "Hierarchical triangulation using terrain features," in *Proceedings of Visualization '90, San Francisco, CA*, 1990, pp. 168-175.

[6] L. L. Scarlatos, "An automated critical line detector for digital elevation matrices," presented at the Technical Papers of 1990 ACSM-ASPRS Annual Convention, Denver, CO, 1990.

*© 2015 Conscientia Beam. All Rights Reserved.*

[7] L. Scarlatos and T. Pavlidis, *Hierarchical triangulation using cartographic coherence* vol. 54. CVGIP: Graphical Models and Image Processing, 1992.

[8] L. L. Scarlatos and T. Pavlidis, "Real time manipulation of 3D terrain models," presented at the 1993 ACSM/ASPRS Annual Convention & Exposition Technical Papers, New Orleans, LA, 1993.

[9] L. L. Scarlatos and T. Pavlidis, "Techniques for merging raster and vector features with 3D terrain models in real time," presented at the 1993 ACSM/ASPRS Annual Convention & Exposition Technical Papers, New Orleans, LA, 1993.

[10] R. Pajarola and E. Gobbetti, "Survey on semi-regular multiresolution models for interactive terrain rendering. The visual computer," *International Journal of Computer Graphics*, vol. 23, pp. 583–605, 2007.

[11] P. Lindstrom, D. Koller, W. Ribarsky, L. F. Hodges, F. Nick, and G. A. Turner, "Real-time, continuous level of detail rendering of height fields," in *Proceedings of Siggraph*, 1996, pp. 109–118.

[12] M. Duchaineau, M. Wolinsky, D. E. Sigeti, M. C. Miller, C. Alrich, and M. B. Mineev-Weinstein, "Roaming terrain: Real-time optimally adapting meshes," in *Proceedings of the 8th Conference on Visualization*, n.d, pp. 81–88.

[13] S. Rottger, W. Heidrich, P. Slusallek, and H. P. Seidel, "Real-time generation of continuous levels of detail for height fields," in *Proceedings of WSCG*, n.d, pp. 315–322.

[14] H. Hoppe, "Smooth view-dependent level-of-detail control and its application to terrain rendering," in *Proceedings of the Conference on Visualization*, n.d, pp. 35–42.

[15] D. Cohen and A. Shaked, "Photo-realistic imaging of digital terrain," in *Proceedings of Eurographics 93, Barselona-Spain, 6-10 September 1993*, n.d, pp. 363-373.

[16] D. W. Paglieroni and S. M. Petersen, "Parametric heights field ray-tracing," in *Proceedings of Graphics Interface*, 1992, pp. 192-200.

[17] G. Vezina and P. K. Robertson, "Terrain perspectives on a massively parallel SIMD computer," in *Proceedings of CG International 91, Springer-Verlag*, 1991, pp. 163-188.

[18] P. Pitot, Y. Duthen, and R. Caubet, "A parallel architecture for ray-casting," *Computer Graphics*, vol. 89, pp. 463-472, 1989.

[19] G. Agranov and C. Gotsman, "Algorithms for rendering realistic terrain image sequences and their parallel implementation," in *Proceedings of Graphicon*, 1995, pp. 153-161.

[20] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," in *Computer Graphics, Proceedings of SIGGRAPH 87*, n.d, pp. 163–169.

[21] R. Shu, C. Zhou, and M. S. Kankanhalli, "Adaptive marching cubes," *The Visual Computer*, vol. 11, pp. 202–217, n.d.

[22] S. I. Vyatkin, B. S. Dolgovesov, and A. V. Yesin, "Parallel architecture and algorithms for real-time synthesis of high-quality images using voxel-based surfaces," in *GraphiCon 2000 Proceedings*, 2000, pp. 117-123.

[23] S. I. Vyatkin, "Complex surface modeling using perturbation functions, optoelectronics," *Instrumentation and Data Processing*, vol. 43, pp. 226-231, 2007.

*Views and opinions expressed in this article are the views and opinions of the author(s), International Journal of Natural Sciences Research shall not be responsible or answerable for any loss, damage or liability etc. caused in relation to/arising out of the use of the content.*

*© 2015 Conscientia Beam. All Rights Reserved.*