Transactions in GIS

Transactions in GIS, 2012, 16(5): 619–633

Research Article

Hybrid of Shape Grammar and Morphing for Procedural Modeling of 3D Caves

Tomasz Zawadzki
Faculty of Electrical Engineering, Computer Science and Telecommunications University of Zielona Gora

Slawomir Nikiel
Institute of Control and Computation Engineering University of Zielona Gora

Korneliusz Warszawski
Faculty of Electrical Engineering, Computer Science and Telecommunications University of Zielona Gora

Abstract

Procedural modeling of three-dimensional shapes plays a significant role in many areas nowadays. Methods based on the automation of the modeling process offer a variety of three-dimensional structures, saving time and money. Geometry synthesis is currently used in many fields including digital cinema, electronic entertainment and simulation. There is a need to replace designers' work with intelligent automated algorithms, especially in the case of terrain modeling. This article addresses the problem of modeling virtual caves and tunnels and presents alternative solutions in the form of a hybrid system. The innovative approach combines two independent methods well known in computer graphics: shape grammars and shape morphing for modeling three-dimensional geometry. In the modeling process, it is possible to obtain the characteristics of 3D structures with non-spherical mesh topology. The objects and their transformations are described by functions, while production grammars define the geometry modeling process. The scene graph can be expanded by classic productions and optimized by morphing productions. Obtained shapes can be freely deformed in subsequent productions. The system offers control over the process of modeling and the resulting structure can be rendered up to a high level of realism. We also propose some measures that can be used to verify the modeling results: coefficients indicating the degree of convexity of three-dimensional model

Address for correspondence: Tomasz Zawadzki, Faculty of Electrical Engineering, Computer Science and Telecommunications, University of Zielona Gora, Licealna 9, 65-417, Zielona Gora, Poland. E-mail: t.zawadzki@weit.uz.zgora.pl

doi: 10.1111/j.1467-9671.2012.01322.x

topology based on the structure of inequality, the volume of the model, surface model and the number of model elements.

1 Introduction

The real-time simulations encountered in three-dimensional graphics hardly obtain a visually satisfactory effect in the short time available. Moreover, real-time simulation is not possible without the use of systems based on procedural modeling of geometry. Complex structures such as buildings, urban cities, facades, plants, terrains or caves in the early stages of computer graphics systems were manually modeled. Algorithms that enable full automation of this process help to achieve large savings of modeling time. Procedural methods can be used to create intricate objects and virtual scenes in real-time. Striving for visual realism of modeled 3D objects obtained in shorter time is the driving force for real-time systems. Technological progress promotes the development of systems based on procedural geometric modeling and introduces them to new areas of science. There is a constant development of new methods i.e. merging technology and dynamic systems (Clempner and Poznyak 2011, Di Trapani and Inanc 2010). However, we have to consider the increased complexity of the scenes, where details of objects now can easily stuck in a “bottleneck” of the rendering pipeline. Cinematography is an area where procedurally generated objects are widely used: cities and buildings (*Superman Returns* 2006), characters – Sally with procedural fur (*Monsters Inc.* 2001), etc. The electronic entertainment industry uses automated methods for modeling: space simulations (*Noctis* 2002), arcade games (*Darwinia* 2005), racing games (*Fuel* 2009), strategy games (*Majesty: The Fantasy Kingdom Sim* 2000), third-person shooters (*Just Cause* 2006), miscellaneous procedural effects (*Left 4 Dead* 2008), (*Borderlands* 2009), etc. Procedural systems are also used in the CAD systems such as: *City Engine* – procedural cities, *Houdini* – procedural animation, *Terragen* – procedural landscapes, and *Art of Illusion* – procedural textures.

1.1 Procedural Caves

Existing efforts in the virtual construction and visualization of 3D cave structures include the use of scanning hardware to obtain accurate spatial data of actual cave structures (Am Ende 2001). The scanned spatial data can be used then to visually reconstruct the real cave. However, 3D mapping of caves using the physical approach is an extremely painstaking and time-consuming process. Schuchardt and Bowman (2007) investigated the visualization of complex 3D cave structures. They researched whether immersive virtual reality provides a higher level of spatial understanding of structures that cannot be mentally visualized using traditional means such as 2D cave maps. The cave model used for their system was constructed from cave survey and measurement data obtained in an actual cave. The procedural creation of fully synthetic 3D cave models has previously been investigated by Boggus and Crawfis (2009a, b; 2010). Their work focused on the procedural generation of solution caves, which are caves formed by rock being dissolved by acidic water. In their research, they applied knowledge about the formation of solution caves in order to create cave models for virtual environments. Their method involved approximating water transport to create a coarse level of detail model for a cave passage (Boggus and Crawfis 2009a, b; 2010). They also demonstrated methods of generating 3D

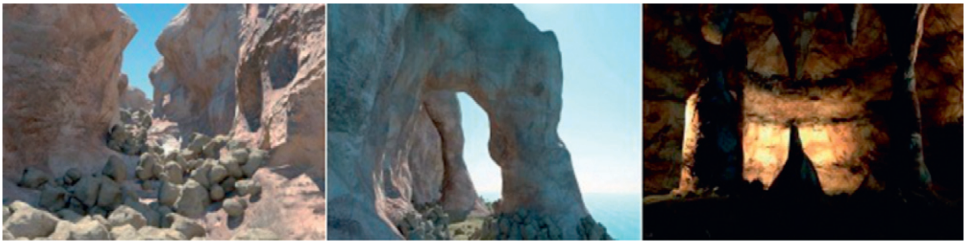


Figure 1 Left – a canyon with rocks detached from the cliffs, middle – a rock arch, right – a cave

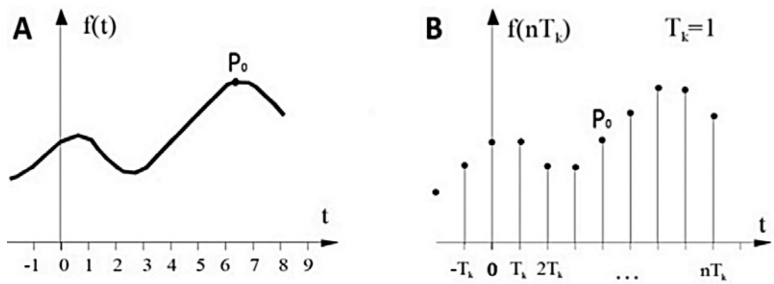


Figure 2 *A* – function describing morphing, *B* – function describing shape grammars, P_0 – point in R^3, *t* – time

cave models using cave patterns, and proposed that surface detail could be added using techniques like bump and displacement mapping. Johnson et al. (2010) examined the approach of using a cellular automata-based algorithm for the real-time generation of 2D infinite cave maps, for the purposes of representing cave levels in video games. However, the generation of 3D caves maps using this approach was left for future work. Peytavie et al. (2009), presented a framework for representing complex terrains, including caves, using a volumetric discrete data-structure (Figure 1). In addition, they proposed a procedural rock generation technique to automatically generate complex rocky scenes with piles of rocks. Their aim was to generate and display physically plausible scenes without the computational demand of physically-based simulations. Their approach mainly focused on the unique data-structure for efficient interactive sculpting, editing and reconstruction in customized high level terrain authoring tools, as opposed to a purely procedurally driven approach (Peytavie et al. 2009).

1.2 Shape Grammars

The research of Stiny and Gips (1972) and Stiny (1975) was aimed at supporting the design process using a "linguistic model of the generational system" and are precursors of shape grammars. The definition of shape grammars is analogous to the formal grammars and is graphically expressed in the language of words composed of symbols with different grammatical rules called productions (Stiny 1980). The origins of shape grammars have their roots in analytic geometry topics, used to give the opportunity to write a formal definition of all types of objects and transformations. These are examples of systems based on the model of generative linguistics. In a general sense, the definition

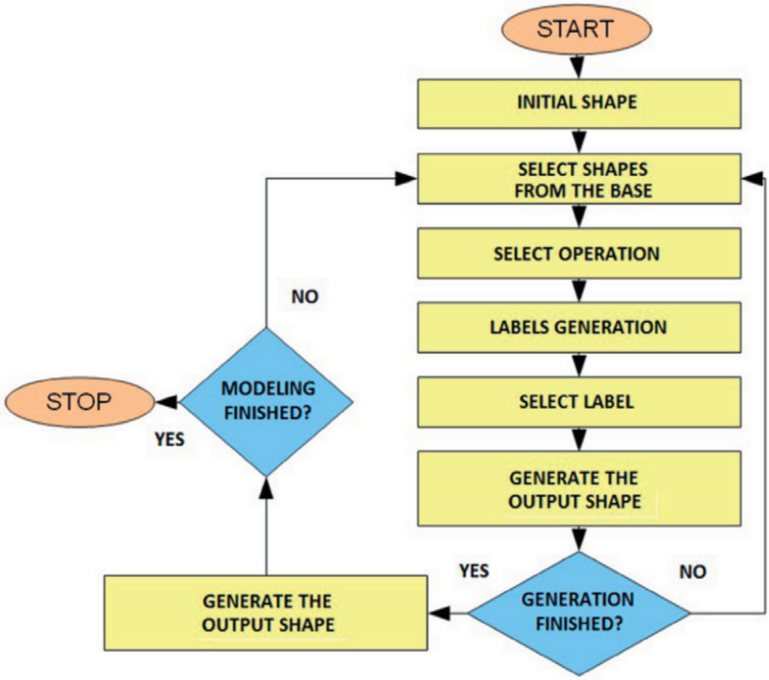


Figure 3 SG+M block diagram

of grammar does not dictate the shape of a field in which it could be used, but assumes that they will be representing the shapes of the n-dimensional space (in the case of three-dimensional graphics has dimension n = 3). The definition of a shape grammar is compatible with the standard definition of phrase structure grammar.

2 Hybrid of Shape Grammar and Morphing

The hybrid is a combination of independent systems; in our case, the shape grammars and morphing. The system has the advantages of both methods and gives added value in modeling three-dimensional shapes. Morphing is described by continuous functions while shape grammars are controlled by discrete functions (Figure 2).

2.1 Construction of the Hybrid System

Virtual representation of shapes and operations is based on a tree structure and consists of three main elements: *the root* – the place from which the modeling process begins; *the child nodes* – the productions described using assemblies of functions; and *the list of shapes* where geometry is described by functions. The system SG+M is based on two independent algorithms. The first one is used to obtain the operations on shapes and is very fast (Figure 3), while the second one is used only to display the results and is quite complex.

The modeling process can be described as a tree of operations performed on the component shapes. The tree node is an operation (sum, difference, intersection or moprhing) while the leaf represents the selected shape. Aviable shapes are: *empty* (initial), *primitives* (sphere, cube, pyramid, torus etc.) and the *base shape* that changes during modeling proces. The base shape takes the abstract form realted to the basic shapes and the operations performed on them. In the first iteration the shape is empty, then it is a shape obtained from the previous production.

2.2 Functional Description of Three-Dimensional Shapes

An alternative approach is the functional description of the shapes. The proposed algorithm in this article is different in many aspects from the standard approach to CSG (Computed Solid Geometry) operations on solids. In contrast to classical algorithms we are not interested in an object made of triangles, but only in the function that describes space where solids are located. The algorithm performs operations only on functions or scalar functions describing the field for the selected shapes.

Definition 1

Let F be set of functions such that $f \subset F$ *and* $f: R^3 \rightarrow R$ describes some scalar field. The functions belonging to F represent solids where the interior is filled with positive values and the exterior with the negative complement and f describes solid A, P is a point in space R^3, and dP is the distance from point P to the edge of the solid. Then if: $dP > 0$; $P \subset A$ (point P belongs to the interior of the solid), $dP = 0$, P (point P is located on the surface of the solid), and $dP < 0$, P (point P is outside the solid surface). This defines primitive shapes (cube, sphere, torus, cylinder, etc.) that will be used in the modeling proces. The point P located in R^3 has three coordinates (x,y,z). For example, a sphere is described by the following function (Velho et al. 1975):

$$f(P) = P.x^2 + P.y^2 + P.z^2 - r \tag{1}$$

where is the function describing a sphere, P is the point in R^3, and r is the sphere radius.

The initial shape (I) is empty shape which is the root of the tree. The results obtained from the function of the distance defining symbol (L) are assigned to the shape. Shapes (S) are described using functions and productions (shape rules) (R) by assemblies on functions.

2.3 Operations on Shapes

The hybrid algorithm based on shape grammars and morphing has four main operations: three Boolean (sum, difference and intersection) and one morphing. The main difference between Boolean operations and the morphing in the SG+M algorithm is the fact that, in the first case we can receive an output model which can have some features of two input models (some parts of model topology can be the same) but in the second operation we obtain a morphed structure that is completely deformed. The morphing operation is always performed on the shape of the base and the recently added shape.

Definition 2

Let f and $f \subset F$ and $g \subset F$ describe solids A and B, respectively, while P is a point with coordinates (x, y, z) in Euclidean R^3. Operations on these solids are defined in the following way (Velho et al. 1975):

Sum:

$$f(P) = f \cup g = \max(f(P), g(P)) \tag{2}$$

Difference:

$$f(P) = f - g = \min(f(P), -g(P)) \tag{3}$$

Intersection:

$$f(P) = f \cap g = \min(f(P), g(P)) \tag{4}$$

Morphing:

$$f(P) = f * g = f(P) * (1 - a) + g(P) * a \tag{5}$$

where a is the morphing parameter. The shapes are described by the final function as a composite of the above functions.

2.4 Classic and Morphic Productions

The hybrid system needs new forms of so-called morphic production. It involves constructing a new shape based on percentage contributions of the current (base shape) and added shapes (Figure 4).

The classic shape grammar productions result in the next steps of the modeling process by operators of sum, difference, and intersection shapes. The sum operation extends the scene graph (classic production), but the morphing operation can augment this process (Figure 5).

The use of classic productions results in adding a new object to the scene and connecting it to the base object. It is different, comparing it with the case when morphic productions are used. This work is focused on the highest density of the output model grid (affecting realism of the 3D structure) while the scene graph is optimized and some redundant productions and shapes are skipped. Morphic productions have an additional morphing parameter (M_P), that ranges from 0 to 100% and affects the deformation of the resulting model grid.

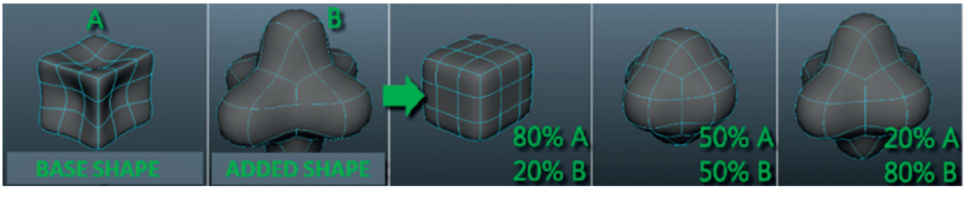


Figure 4 Presentation of the percentage contribution of shape A and B on the result shape obtained with morphic production

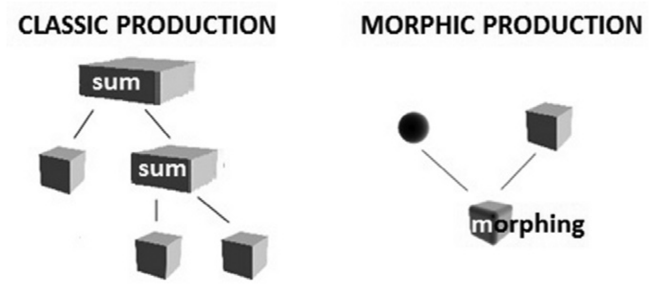


Figure 5 Example realizations of classical and morphic productions

Definition 3

The total sum of all production is given by:

$$P_T = P_C + P_M \tag{6}$$

where P_T represents all productions, P_C represents classic productions, and P_M represents morphic productions.

2.5 Binding and Labels

An important problem that was solved in the research was to determine the switching points of the grid for the main objects (primitives) and the objects created in the modeling process.

Definition 4

Bond (B) defines the way in which one of the shapes can be combined, which is described as the point P and the direction D in three-dimensional space:

$$B = (P, D) \tag{7}$$

Definition 5

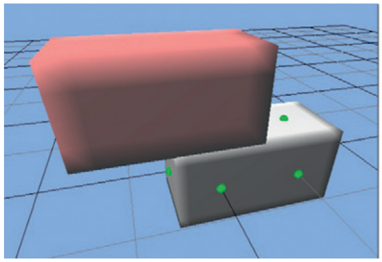
The label (L_B) is one of the possibilities of combining two shapes by using appropriately selected bonds. Two bonds make a label when the points P overlap and directions D are opposite.

This can be interpreted as the gluing of two walls facing each other in opposite directions, which allows the exclusion of some unnecessary connections:

$$L_B = (B_1, B_2) \Leftrightarrow (B_1.P = B_2.P \textit{ and } B_1.D = -B_2.D) \tag{8}$$

where L_B represents the bond, B_1, B_2 are labels, P is a point in R^3, and D is the direction for shape.

In addition, the algorithm provides that the bond does not have a direction if $D = (0, 0, 0)$ and then it can create a label with any other bonds. Switching points are fixed and



N = (0, 1, 0) , P = (0, sy, 0)
N = (0, -1, 0) , P = (0, -sy, 0)
N = (-1, 0, 0) , P = (-sx, 0, 0)
N = (1, 0, 0) , P = (sx, 0, 0)
N = (0, 0, 1) , P = (0, 0, sz)
N = (0, 0, -1) , P = (0, 0, -sz)
N = (0, 0, 0) , P = (0, 0, 0)

Figure 6 Bond points for a cuboid

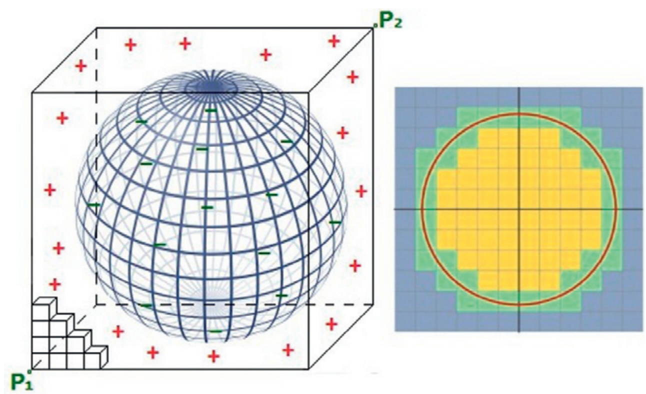


Figure 7 Cross-section distribution of cubes intersected by a solid. Green color depicts the selected cubes

defined without any algorithm. For example, for the cuboid the points are dependent on the shift in the x, y and z axis (sx, sy, sz parameters) (Figure 6).

2.6 Grid Display Algorithm

For each point in R3, we can estimate its distance from the surface of the output shape associated with the CSG tree graph. This helps us to determine whether the point is inside, outside or on the surface of the shape volume. For all points, sampling is carried out at a constant distance and thus creates a scalar field. From the scalar field, at each point in space, it is determined whether a point is inside or outside the solid or may be on its side surface. In order to view this block plane it is not possible to select all points that are placed exactly on the surface, because the sampling density can be too large. We propose approximation sampling (size cube sample) and rely on the assumption that each cube has eight vertices in the sampling. It is drawn on the screen when there are at least two vertices that have the opposite sign for the distance. At this stage we check whether the surface intersects the solid cube (or the cube is entirely inside or outside of the solid). If the solid crosses the cube block, than we can display it (Figure 7).

We use space partitioning with cubes. When we determine all the boxes that are intersected by the shape, we can read the incidental edges, i.e. those that collide with the surface of the shape volume. Using linear interpolation, we can choose exactly the point

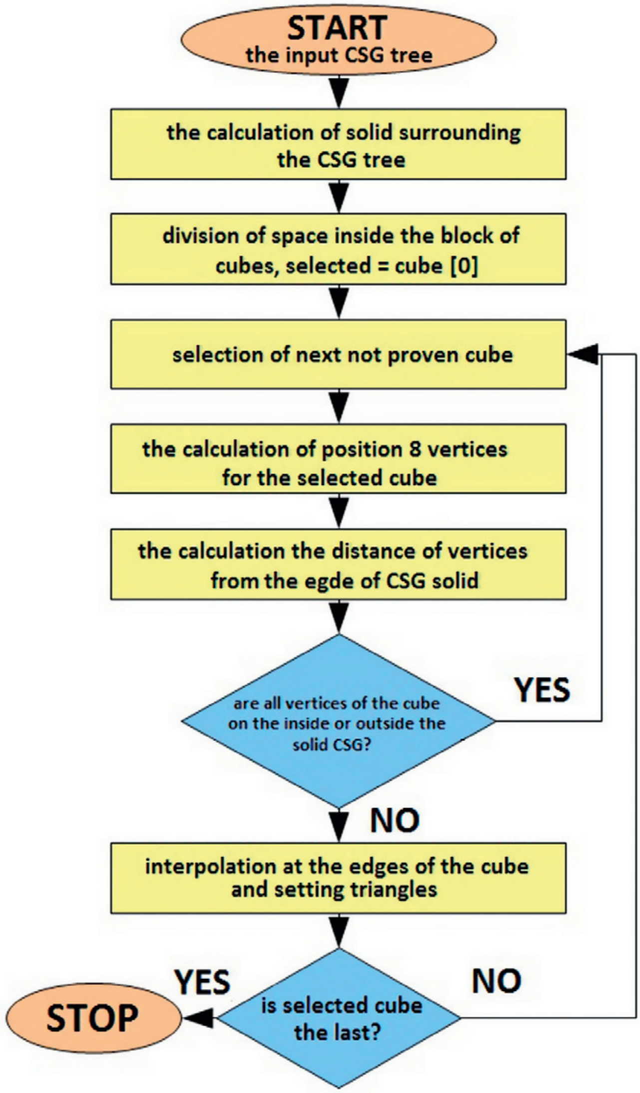


Figure 8 Mesh generation algorithm

of intersection of solid with each edge of the cube separately. From the previously prepared table of intersections (256 combinations) the system reads the triangles that form the points of intersection and displays them.

The algorithm gives an accurate approximation thanks to the CSG dependent density sampling. It generates shapes that cannot be achieved by standard methods of modeling. The outline of the algorithm is shown in Figure 8.

2.7 Mapping a Texture on the Surface of Three-Dimensional Object

In order to improve the realism of the modeled geometry we used the additional software to apply the textures on the final object. The 3D Studio Max software is dedicated to the

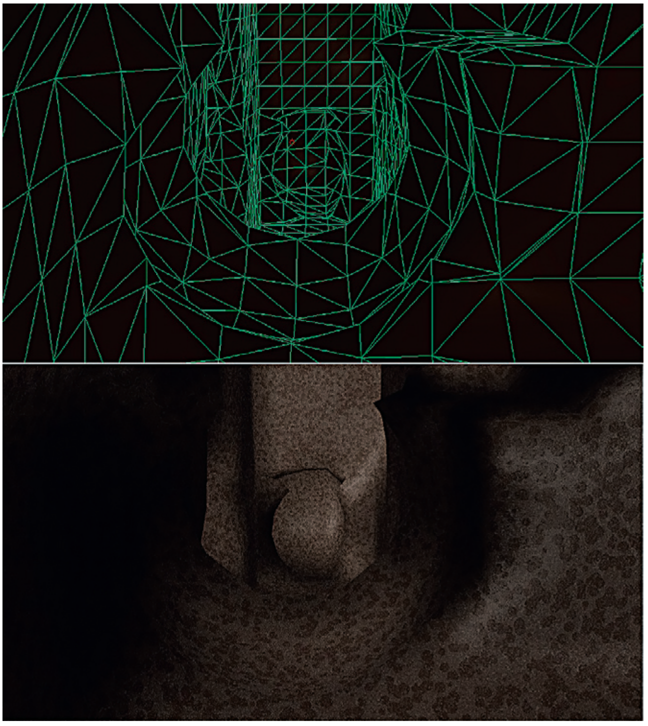


Figure 9 Obtained structure of the cave: top – grid model; bottom – the textured rendering

three-dimensional graphics and helped us with this task. Based on the created module, we exported the geometry of the model developed to the *.obj format, and then loaded the file into the external environment. In the program, the file model lacked texture and the geometry did not have artifacts on the surface. Imposition of texture and rendering enriches the visual effect of the SG+M algorithm and presents it in a more visually appealing form.

3 Results

In the algorithm we implemented the following parameters showing the influence on the modeling process: the level of detail in the model grid (L_D, 1..32), classic productions (P_C) – operations (sum, difference, intersection), morphic productions (P_M) – operation morphing, direction of the labeling (L_B, $-X$, $+X$, $-Y$, $+Y$, $-Z$, $+Z$ and random), the morphing parameter (M_P, 0..100%) and the production number for each shape (P_S, 0..10, this helps to assemble primitive shapes in more complex shapes).

The parameters of the experiment were $L_D = 10$, $P_C = 10$, $P_M = 2$, $L_B = +X$, $M_P = 52\%$, and $P_S = 3$. The resulting three-dimensional structure shows the topology of the caves (Figures 9 and 12) and based on the analysis of the results (from Figure 9), the performance of the proposed method is shown in Figure 10. All simulations were performed on a nVidia GeForce GTX 460M GPU, i7-2630QM CPU and 12 GB RAM.

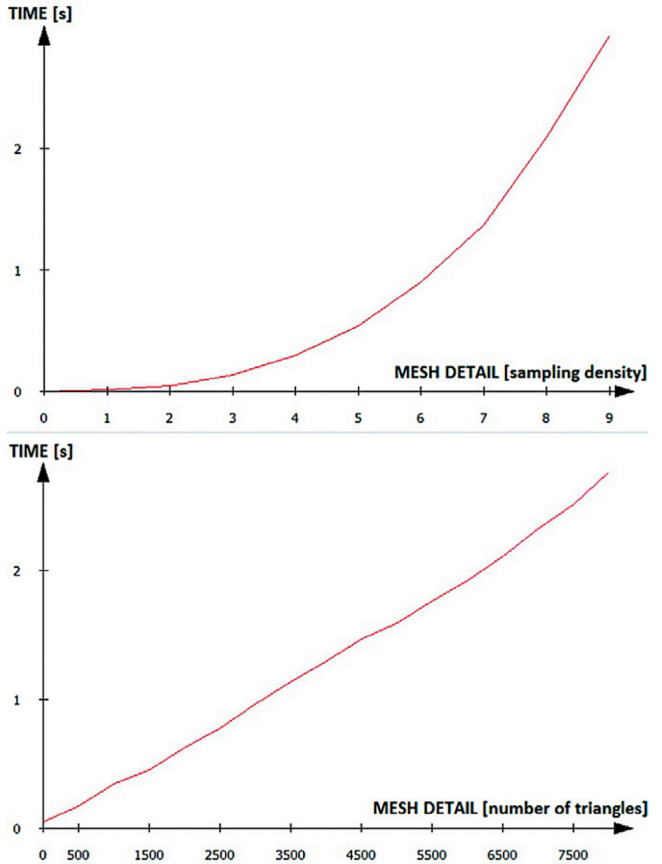


Figure 10 Time vs mesh detail (L_D, represented by sampling density) and number of triangles

4 Verification Methods

In our research we also considered development of valuable verification methods. We propose seven metrics that can be used to verify the geometry of the rendered model. The verification parameters are described below.

4.1 Surface Area of the Model

After generating the grid we obtain triangles that make up the model. For each triangle we have three vertices (V_1, V_2, V_3) in the three-dimensional space described by x, y, and z coordinates. There is a formula for the surface of the triangle that was so described based on the vector product (Figure 11):

$$S = \frac{1}{2} * ((V_2 - V_1) \times (V_3 - V_1)) \tag{9}$$

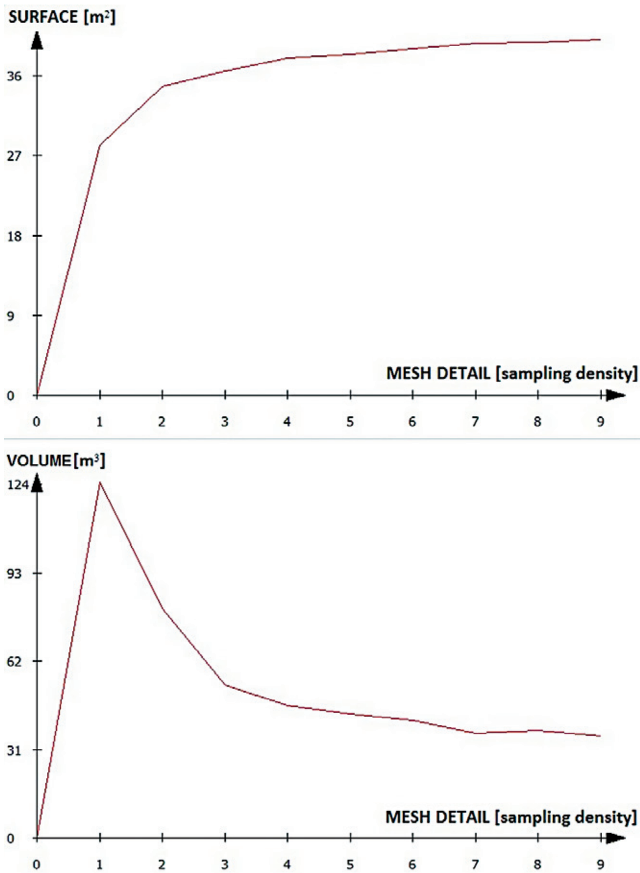


Figure 11 Surface and volume vs mesh density (L_D, represented by sampling density)

4.2 Volume of the Model

During the mesh generation process from existing productions we obtain sampled cubes. The number of vertices that are generated inside the solid are counted for all of the cubes. The interior is determined by the function describing the shape. If $f(p)$ is a function of the distance from point p generated from a solid cube, counted when all eight vertices of the cube $V_1..V_8$ meet the condition $f(V_i) < 0$, the volume is calculated by:

$$V_M = N_C * (C_E) \wedge 3 \tag{10}$$

where V_M is the volume of the model, C_E is the cube edge ($C_E = 1/\textit{grid density}$), and N_C is the number of cubes inside.

4.3 Concavity and Convexity Factor

All three-dimensional structures for which the convexity ratio is below 100% have concavity on the grid, but for those for which ratio is equal exactly 100% the mesh is convex, e.g. a cube.

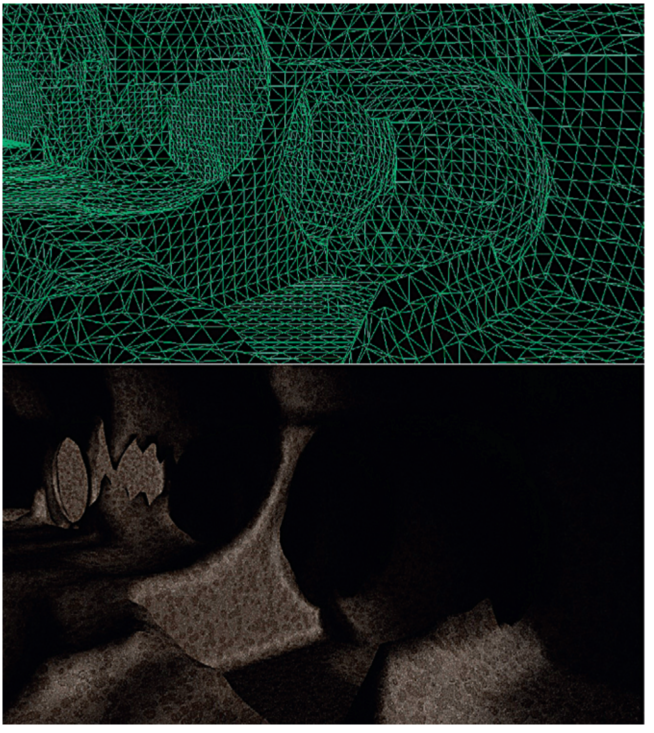


Figure 12 Obtained structure of the cave for $L_D = 32$: top – grid model, bottom – textured rendering

The concavity and convexity factor is defined by the following formula:

$$C_F \approx \frac{100 * T_G}{T_G + T_L} \tag{11}$$

where C_F is the concavity and convexity factor, T_G is the the number of triangles between which the angle is greater than 0°, and T_L is the the number of triangles between which the angle is less than 0°.

Its values fall in the range 0 to 100% and depend on the number of existing triangles with an angle of d greater than 0° and less than 0°. If $d > 0$ it is concave and if $d < 0$ it is convex. After summing up the relation between the edges of all triangles we will get information about the ratio of the solid (Figure 13.).

Research has shown that for objects with the topology of either a tunnel or a cave this ratio varied between 40 and 75%.

5 Conclusions

Nowadays, we can observe noticeable trends in applying new methods of modeling 3D caves in real-time. This is forced mainly by market demand in areas of digital entertainment and simulation for 3D gaming. Our article presents an innovative method for

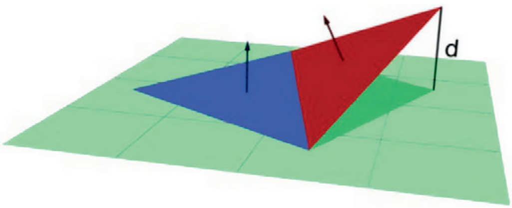


Figure 13 The idea of calculating the convexity (d – parameter defining the angle between neighboring triangles)

real-time procedural modeling of 3D caves and tunnels. By adding to the formalism of shape grammars the additional feature, morphing, we obtain greater variety and geometric complexity (highly influencing visual realism) of synthesized objects. The method offers an advantage compared with classical methods based on shape grammars; it optimizes the associated scene graph using a new feature – morphic productions. The morphing parameter allows us to establish the continuous percentage contribution of two input shapes to produce a single output object. This results in optimization of the modeling process and reduction of redundant objects. The proposed method behaves correctly when building 3D meshes based on non-spherical topology. Our further research will focus on development of 'material-driven' modeling, linking the range of morphing productions to the type of geological components.

References

Am Ende B A 2001 3D mapping of underwater caves. *IEEE Computer Graphics Applications* 21(2): 14–20

Asobo Studio 2009 Fuel (Movie)

Avalanche Studios 2006 Just Cause (Movie)

Boggus M and Crawfis R 2009a Explicit generation of 3D models of solution caves for virtual environments. In *Proceedings of the International Conference on Computer Graphics and Virtual Reality,* Las Vegas, Nevada: 85–90

Boggus M and Crawfis R 2009b Procedural creation of 3D solution cave models. In *Proceedings of the Twentieth IASTED International Conference on Modelling and Simulation,* Banff, Alberta: 180–86

Boggus M and Crawfis R A 2010 A framework for interactive modeling of three dimensional caves. In *Proceedings of the Sixth International Conference on Advances in Visual Computing (ISVC'10)*, Las Vegas, Nevada: 213–21

Clempner J and Poznyak A 2011 Convergence method, properties and computational complexity for lyapunov games. *International Journal of Applied Mathematics and Computer Science* 21: 349–61

Cyberlore Studios 2000 Majesty: The Fantasy Kingdom Sim (Movie)

Di Trapani L and Inanc T 2010 NTGSIM: A graphical user interface and a 3D simulator for nonlinear trajectory generation methodology. *International Journal of Applied Mathematics and Computer Science* 20: 305–16

Gearbox Software 2009 Borderlands (Movie)

Ghignola A 2002 Noctis (Simulation Game)

Introversion Software 2005 Darwinia (Simulation Game)

Johnson L, Yannakakis G, and Togelius J 2010 Cellular automata for real-time generation of infinite cave levels. In *Proceedings of the Workshop on Procedural Content Generation in Games (PCGames '10),* Monterey, California: 1–4

Peytavie A, Galin E, Grosjean J, and Merillou S 2009 Framework for modeling complex terrains. In *Thirtieth Annual Conference of the European Association for Computer Graphics* (*EUROGRAPHICS* 2009), Munich, Germany: 457–67

Pixar Animation Studios 2001 Monsters (Movie)

Schuchardt P and Bowman D 2007 The benefits of immersion for spatial understanding of complex underground cave systems. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST '07),* Irvine, California: 121–24

Stiny G 1975 *Pictorial and Formal Aspects of Shape and Shape Grammars*. Basel, Birkhauser Verlag

Stiny G 1980 Introduction to shape and shape grammars. *Environment Planning B* 7: 343–61

Stiny G and Gips J 1972 Shape grammars and the generative specification of painting and sculpture. In Freiman C V (ed) *Information Processing*. Amsterdam, North-Holland: 1460–65

Turtle Rock Studios 2008 Left 4 Dead (Movie)

Velho L, Gomes J, and Figueiredo L 1975 *Implicit Objects Computer Graphics*. Berlin, Springer-Verlag

Warner Bros. Pictures 2006 Superman Returns (Movie)