Real-Time 3D City Generation using Shape Grammars with LOD Variations

Pearl Goswell and Jun Jo

***Abstract*—**Creating3D environments, including characters and cities, is a significantly time consuming process due to a large amount of workinvolved in designing and modelling.There have been a number of attempts to automatically generate 3D objects employing shape grammars. However it is still too early to apply the mechanism to real problems such as real-time computer games.The purpose of this research is to introduce a time efficient and cost effective method to automatically generatevarious 3D objects for real-time 3D games. This Shape grammar-based real-time City Generation (RCG) model is a conceptual model for generating 3Denvironments in real-time and can be applied to 3D gamesoranimations. The RCG system can generate even a large cityby applying fundamental principles of shape grammars to building elementsin various levels of detailin real-time.

***Keywords*—**real-time city generation, shape grammars, 3D games,3D modelling.

I. Introduction

THE complexity involved in 3D environment is particularly evident when generating a large scale city containing many buildings. In order to generate a city model, an intensive amount of modelling, texturing and lighting work is required to produce realistic urban sceneries in 3D. Currently, most of these tasks are manually carried outby many 3D designers and heavily demandmany simple and repetitive tasks. This conventional approach makes the 3D modelling work laboriously intensive and time consuming, and usually requires a fast computer with a large amount of data storage [6]. As a result, the time and production cost for creating a 3D game increase significantly.This often causes financial pressuresto the developers resulting inquality degradation through repeatedly reusing the same assets or settings[15].This research presents the shape grammar-based Real-Time City Generation (RCG) system. The RCG system implements shape grammar rules to generate a 3D city in real-time,manipulating building elements in various levels of detail.The 3D city can be created by graphic designers during the game development process or while the game is being played in the user's computer.

Pearl Goswell completed Bachelor of Multimedia with Honours in 2010 and is now a PhD student at School of Information and Communication Technology, Griffith University in Australia. Her research interest is in3D modelling, real-time 3D games, Chaos theory and Shape Grammars. Her email address is pearl@pearlgoswell.com.

Dr. Jun Jo was awarded his PhD degree from the University of Sydney in 1994. He has conducted research in various areas including Computer Vision, Robotics, Computer Games and e-learning and has published over 100 refereed publications. Dr. Jo is currently taking the positions of the Director General of International Robot Olympiad Committee (IROC) and the President of Australian Robotics Association (ARA). His email address is j.jo@griffith.edu.au.

This paper introduces shape grammars to generate various 3D objectsefficiently and effectively and proposes an automatic city generation system for 3D games orany 3D applications.The efficiency and effectiveness of this approach will be discussed, in terms of time for development and the variety of the solutions.

II. Generation of 3D Environments for 3D Games

For the gaming industry, 3D models are an essential component. In order to create 3D models for games, there are a number of different 3D modelling programs, such as Maya, 3ds Max, Blender, Lightwave, Modoetc[14]. Utilising these 3D modelling applications, the actual process of developing a 3D model is mostly done manually, in order to follow the multiple stages in the working pipeline of 3D manufacture [14].

In most cases, generating 3D environments requires an expensive modelling pipeline [8]. As the quality of 3D contents has to be richer and higher to meet increasing expectations from users these days, it may take hundreds of 3D artists for years to create realistic 3D environments for a game, for example a city that is common for a modern game environment [15]. Complex work pipelines between level designers and environmental artists also add to the inefficiency issues imbedded in the development of the city modelling [6]. These complex work pipelines may ultimately affect entire production costs and often seriously causes other major issues, referring to that of quality control [15].

III. Shape Grammars: 2D and 3D

Shape grammars are a series of rules that define or set an arrangement of labeled shapes, including points, lines, faces and 3D objects. These rules can operate and generate complex architectural designs, or become descriptions of the forms of the generated design [3]. Shape grammars were originally influenced by the formal language theory of Chomskyan, which is about a set of formation rules for strings in a formal language [11]. Sinceshape grammars wereintroduced to architectural design by Stiny and Gipsin 1972, shape grammars have been refined and further developed by many researchers [15].

A. Analysis and Evaluation of Art in 2D

Stingy and Gips defined shape grammars as taking shapes from a primitive, with shape specific rules and orientating them from pattern recognition formalism. In their papers, shape grammars were utilised as an original language for painting or sculptures in order to analyse their design patterns [11]. Since then shape grammars were popularly used to interpret and evaluate works of art right up to the mid 90's. During the period, a number of artists often utilised shape grammars to

algorithmically generate art works or to analysis and evaluate the aesthetic quality of paintings or decorative arts [16]. Daniel Sheets Dye was able to assemble traditional Chinese lattices, which wereconstructed between 1000 BC and 1900 AD, by using shape grammars in 1974 [12]. Koning and Eizenberg used shape grammars to grammatical analyse the spatial and massing arrangements of Frank Lloyd Wright's 'advanced ornamentation design theories' in 1981 [5].

B. Studying and Creating 3D Structures

Shape grammars are becoming a popular tool to study 3D structures such as historical houses [1]. In 1982, Stiny developed Kindergarten Grammars, which were based on the kindergarten method of Frederick Froebel [10]. Stiny's kindergarten grammars demonstrated the real value of the applications of shape grammars in architecture. As a result, shape grammars provided a new direction to researchers, who looked for design languages that were not as much technical, but rich enough to provide the starting point for complex and sophisticated designs [4]. Flemming used shape grammars to analyse and regenerate 19th century Queen Anne style houses in 1987 [2]. Knight extended the idea by developing a vocabulary of shapes and spatial relationships between shapes in 1992. This attempt was made to find a key compositional idea for shape grammars. In her research, constraint rules were used to build spatial relations in order to define how to lay shapes relating to each other [4].

3D shape grammar was recently renamed as Computer Generated Architecture (CGA) shape grammars andwas applied to the procedural modelling method in CG architecture [7]. The commercial value of shape grammars has also been re-evaluated by 3D industries, in terms of efficient generative methods for massive urban models [7].

IV. The Real-time City Generation System (RCG)

This paper proposes the shape grammar-based Real-Time City Generation (RCG) system that provides fast 3D modelling coupled with high levels of variety in assets, while using limited memory consumption. The RCG system designs and automatically generates 3D models by utilising shape grammar rules. As shape grammar contributes to providing various styles of 3D geometric models, the 3D environments, which are generated by the RCG, stimulate a user's continuous attention by preventing monotonous repetition. After the generation phase is completed, the evaluation phase can be customised and carried outby the usermanually or by the system automatically.

As Fig. 1 indicates, the RCG system generates 3D environments based on a specified design style defined in the initial state. There are two major phases involved in the RCG system. The Generation Phasegenerates 3D models by combining shape rules and 3D elements retrieved from the RCG Library. The Evaluation Phaseevaluates the usability of the solution by the user orusingthe pre-set evaluationcriteria,and determines whether the solution satisfies the pre-defined conditions.

A. Initial State (Design Phase)

The Initial state is the process setting up the condition or requirements of a desired 3D environment by setting up parameters and data. The input data includes Style (S), Regional Information (T) and Level of Detail (LOD) Information,which are needed to select suitable elements from the library.

$$I = \{S, T, LOD\}$$

S represents the Style of a building. The 3D elements in the library contain eleven different building styles and these styles are classified chronologically and geographically.

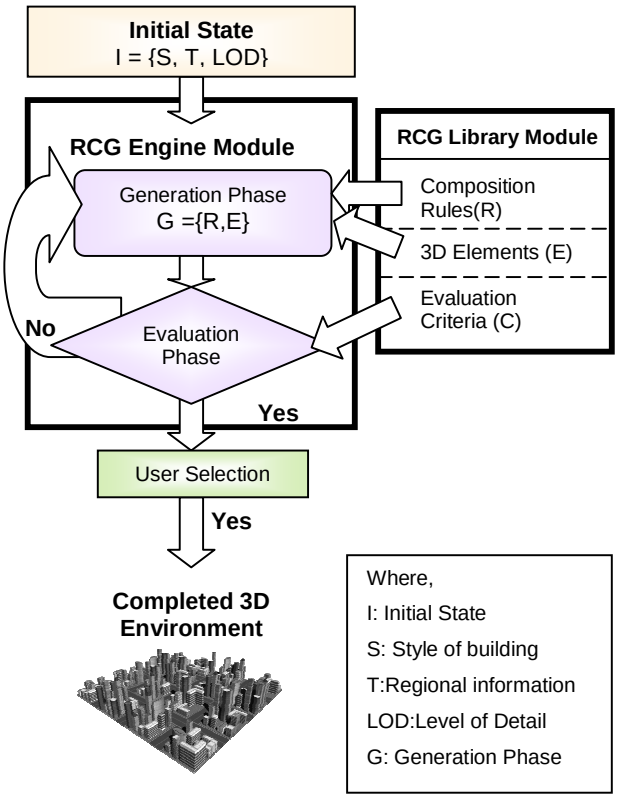


Fig. 1 TheRCG system

T represents Regional information and defines territorial design elements. This data is fundamental to compose a site map and to guide the generation process within the given context.T, fortheimplementation,containsthree different categories: mountain, coastal and inland regions. The sitemaps of each regionare composed of typicalstyles of metropolitan cities withterritorial data and provideunique regional characteristics to the city.

LOD defines the detail-level of building elements for modelling. As LOD determines the selection of both

composition rules and 3D elements from the 3D Element Library, it needs be specified at the initial state. The high LOD indicates that the user can control themodellingprocess from primitive design of the structure. In the low LOD, the system uses a number of pre-defined design sets in order to reduce the time of composition.

B. The RCG Engine Module

The RCG engine is divided into two major phases: the Generation Phase and the Evaluation Phase. In the Generation Phase, the system generates 3D models by assembling elements retrieved from thelibrary,using rule-based algorithms. During the Evaluation Phase, the solutions are evaluated by using the evaluation criteria in terms of context fitness and functionality. If not satisfied, this outcome will be disposed and the process will move back to the Generation Phase again.

The Generation Phase: A major role of the Generation Phase is interpreting and executing shape grammars based on information, where wasset in the initial state. The mechanism of the Generation Phase can be can be described as:

$$G = \{R, E\}$$

R represents rule-based algorithms or shape grammars with two variables,T (Regional Information) and LOD(Level of Detail). By using these variables, generation rulesarecustomisedand generates 3D buildings appropriately.

E indicates 3D building elements retrieved from the Library with two variables: S (Style of building) and LOD. Based on the values of these variables, a set of predefined primitive shapes or high-level building elements are summoned from the 3Dlibrary moduleand generate a desired 3D model or a city.

The Evaluation Phase: Aftera 3D environment is assembled in the Generation Phase, the legitimacy of the outcome needs to be evaluated,whether itfulfils the requirements of the initial setting and follows the guidelines of a selected site map. The system receives this constraint informationto evaluate the outcome, based on evaluation criteriaspecified in the library. For instance, if building A has the entrance door facing the road and building B has its entrance facing its backyard. Then the fitness value of building A may be higher than that of building B if the accessibility was considered as an evaluation criterion.A user mayadjust the number of evaluation criteria by monitoring manually, in order to optimise the process speed. Overall, the main role of the user or designer in the system is to evaluate the final outcome in the evaluation phase and to decide whether it hassuccessfully produced a solutionor needs to go back to the generation phase anditerate the process.

C. The RCG Library Module

The RCG Library is the storage of rule sets and 3D elements. There are three distinctive sections: a Composition Rule Library, 3D Element Library and Evaluation Criteria Library. The RCG Library is designed to respond to the Generation Phase immediately,in order to provide required elements,as soon as the user inputs data in the initial state.

Composition Rule Library: Composition rules, through their appropriate applications to building elements,generate a building and a group of buildings, a city. In the RCG, there are three categories in the rule sets with differentLODs. The simplest rule set is a district rule set, Fig. 2.For the experiment, a single district was made of sixteen blocks. Then each block contains four sites with a number of buildings. For instance, if the LOD is set up in a very low level,the rule sets of pre-built district will deal with 16 block sets. The user will then select and join theselected districts and generate a city in a short time, Fig. 2.

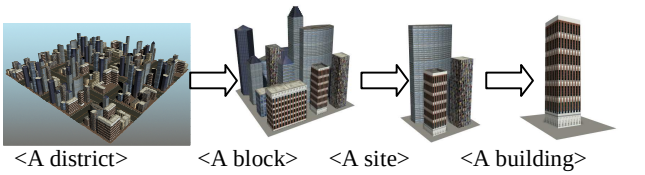


Fig. 2 A basic structure of a district

If theuser wants more complicated and creativecitydesign, the user can access the block, rather than district, rule set andcan manipulate and assemble blocks. Ablock rule set has four sets of sites and each building is summoned from the library based on the LOD. Operation with block rule sets takes more processing time to generate a 3D environment than using a district rule set. The site rule set comprises even higher LOD, and therefore involves more components, Fig.3.

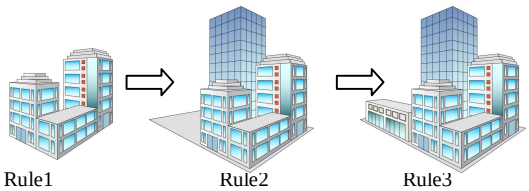
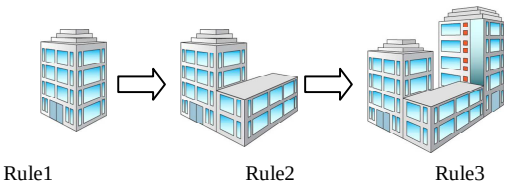


Fig. 3 Assemblingbuildingsto generate asite

In order to makethe city design more creative and diverse with a various building combination, the RCG system allows a user to access a building rule set.

Fig. 4 demonstratesa building rule set that assemblesseveral buildings into a building site by applying a number of rules. Rule 1 executes generating the first building, Building1, which has a basic building structure with pre-selected window, wall, roof and door. Rule 2 then attaches the second building to the Building1. The variables of the additional building such as x, y, z coordination for position, width and height will be decided by Rule 2. Rule 3 attaches another building part into the site with different variable values.



X coordination of part a: Xa
Y coordination of part a: Ya
Width of part a: Wa
Length of part a: La
Height of part a: Ha

Rule1: Select a type of wall, roof, door, and window and then construct part1 of the building.

Rule2: Based on the location of the door, attach part2 to the building.
*W2 = 0.5 * W1*
*L2 = 2 * L1*
X2 =X1 -(3/4) W1*
Y2 = Y1- L1
*H2.=O.5 * H1*

Rule3: Attach part 3 to the building. at (X1-(1/2) W1, Y1-(1/2)*H1)*
*W3 = 2 * W2*
*L2 = 0.5 * L2*
X3 =X2+ W2
Y3 = Y2 -(3/4) L2*
*H3.=3 * H2*

Fig. 4 An example of the Building Rule set with Mel, or Maya scripts.

3D Element Library: The 3D Element Library stores shapes, pre-assembledmodels and a number of different site maps,which will be used duringthe Generation Phase of the system. Based on the givenLOD, the RCG system accessesthe 3D Element Libraryand collectsbasic building components or models. The 3D element library containsnumerousstyle elements, which are essential to generate various building styles. These styles aresorted in aperiodical and geometricalorderand are ready to be used for the Generation Phase.

Evaluation Criteria Library: The Evaluation Criteria Libraryholds all the information needed toevaluate the usability of the 3D environments that are created by the system. Some of the evaluation constraints include the constraints for generation, allocationand direction.

The Constraint for Generation is neededto monitor whether or notbuildings are built ina buildable area,nota road or apark.To achieve this goal, The Constraint for Generationrefers to the site map provided by theuser inthe initial stage.

The Constraint for Allocationprevents undesirable allocation of buildings, for instance overlapping between buildings inthe Generation Phase.

The Constraint for Directionhelps buildings face their entrances to the main roads. In order to apply these constraints to the final solutionseffectively, the fitness level should be designed, tested and tuned appropriately.

V. Implementation

This paperaimed todemonstratethe effect of algorithmic shape generationand the usability of shape grammars in a 3D game environment. To demonstrate the effect, the GCG system was implemented with a districtof 16 building blocks in low LOD. Two city models, a modern city and an ancient Greek-style city, were adoptedand implemented.

A. Initial State

The RCG system starts creating acity by initialising a set of data: S(Style of building), LOD (Level of Detail)andT(Regional information). These elements carry all the initial information to build a city. Based onthis blueprint of the city, each component is summoned from the 3D Element Library to construct 3D models with shape grammars. The 3D models then become individual items of the model array, which is a temporary storage of 3D models.

B. Model Array

The model array is a virtual temporary storage and is formed before the RCG system starts assemblingbuildings, blocks and districts. The size and items of the model array may vary depending onthe LODs. Fig. 5 shows an example ofthe model array items, which are pre-built site sets with a low LOD for this implementation.

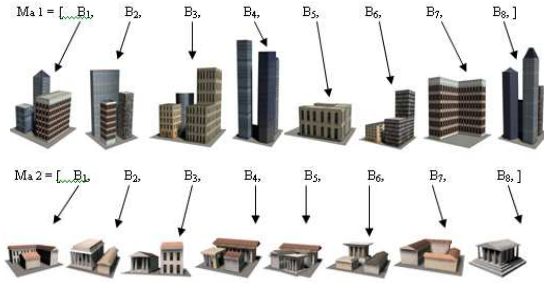


Fig. 5 Model array items (Ma1: modern style, Ma2: ancient Greek style)

By using the eight pre-built sites of each style inthe model array, 64 sites (16 block x 4 sites per a given block) should be generated and allocated in a district. To increasethelevel of variety and creativity inthe city, LOD was increased from the block-level to the site-level. A number of models stored in the array are varied depending on LOD and the features of each model arespecified by S. The process of model selection from the array and its instantiation can be described by Mel, a script language for Maya, as below:

```
globalproccreateNew(){
string $Ma[] = {"m1", "n1", "o1", "p1", "q1", "r1", "s1", "t1"};
// set up array for block
int $Ma_choice;
$Ma_size = `size($Ma)`;
// detect array size
$Ma_choice = `rand 0 $Ma_size`;
// pick random number from array size
select $Ma[$Ma_choice];duplicate -rr;
// duplicate selected model from the array
};
```

This example scripts describe howbuildings are selectedfrom the model array by using the site-set rule. The site-set rule is designed to summon pre-built buildingsfrom the model array. The array also stores pre-built sites for a higher LOD process.

C. Generation and Constraint Rules

The system can rotate or scale the target buildings, in order to add more variety to the solutions or to make the solution satisfy the constraints. Transformations on dimension or direction of models are easy but a very effectiveway to improve the solutions.

Once the model array is generated, considering the dimension of the selected site map, a number of blocksare selected and, embedding shape grammars,converted into the Default Building Generation Rule (DBGR). An example of the Mel script for the process is shown below.

```
$i = 0;
$number_of_blocks = 43;
while($i< $number_of_blocks){
createNew();
if($i< 11){
    $xPosition = $i*4 - 7;
    $zPosition = -7;
move $xPosition 0 $zPosition; }
}
```

As the DBGR does not considerroads, parks or a feasibility of the building site, it allocates buildings to anypossible spaces inthe sitemap, as inFig.6.

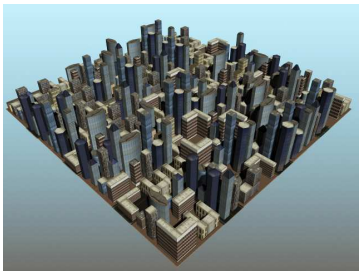
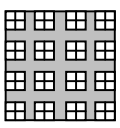


Fig. 6 Allocating buildings by the DBGR

After the DBGR is applied, shape grammars for the Default Road Generation Rules (DRGR) are executed and construct roads within each block.As Fig. 7 shows, dimensions of roads and blocksare based on the status of the sitemap. This process is performed in the early stage of the procedure and DRGR interprets the information relating to all the geometrical elements from the sitemap.During the road construction, if the system detects any conflicts between a building and a road, the target buildingwill be eliminated and replaced by a road immediately.



- *Width of road between each blocks :RWb = 4*
- *Width or height of one building site : BW = 4*
- *Dimension of Site Map: 8 * BW + 3 * RWb*
- *Number of total blocks : 64*

Fig. 7 Calculating dimension of the sitemap to execute DRGR

The rules also create non-residential components and natural environments, such as parks, beaches, hills and mountains. These elements are important to provide a realistic atmosphere to 3D games, no matter the target object is a metropolitan city or an ancient city. For this paper, oneresidential district and three non-residential areas wererandomly allocated in the sitemap, Fig. 8.

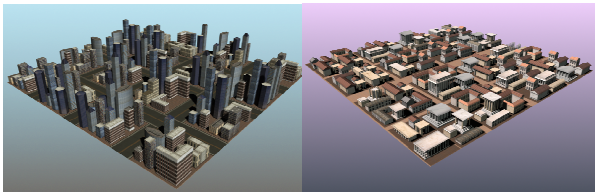


Fig. 8 A district (left: modern city, right: ancient Greek city)

VI. Discussion

In this paper, the RCG system wasintroduced as an assistant or a replacement tool for the manual modelling method. This section willdiscuss the efficiency, in terms of time and cost, and effectiveness, in terms of creativity or variety, of the concept.

A. Efficiency

From the implementation of the RCG system, it is obvious that the use of shape grammar-based algorithmic approachsignificantly reduces the time and cost to produce a 3D city model, in comparison with the manual modelling method. For the comparison study, a skilled 3D graphic designer was involved to manually create a number of 3D models and the production time was measured. The models that were created by both the graphic designer and the RCG system were identical. Tables 1 and 2show the record of time spent to create asingle district using both the manual method and the RCG system. The differenceof the production time is too apparent and therefore a numerical comparison seems to be meaningless.The data for the comparison contains 16 blocks of one district. For this implementation, asingle metropolitan city involved 10 to 15 districts in the given sitemap.

TABLE I
CREATION TIME FOR A MODERN CITY

Method	***Total Creation Time***
manual method	218 min 11 sec
RCG system	0.864 sec

TABLE II
CREATION TIME FOR AN ANCIENT GREEK CITY

Method	***Total Creation Time***
manual method	359 min 05 sec
RCG system	0.973 sec

Table 3 indicates the times to create a site for the model array of each city by manual method. The manual method takes an average of 27 times longer time than that of the RCG system to generate target objects.

TABLE III
CREATION TIME FOR BUILDING SITES BY MANUAL METHOD

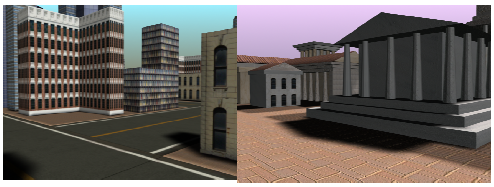
Items	Man made
site 1	40 min 29 sec
site 2	45 min 53 sec
site 3	49 min 18 sec
site 4	51min 26 sec
site 5	38 min 42 sec
site 6	44 min 37 sec
site 7	32min 29 sec
site 8	56 min 11 sec

a: Modern city

Items	Man made
site 1	30 min 34 sec
site 2	32 min 40 sec
site 3	25 min 07 sec
site 4	20min 46 sec
site 5	31 min 22 sec
site 6	44 min 37 sec
site 7	19min 41 sec
site 8	31 min 58 sec

b: Ancient Greek city

Another advantage fromthe use of the RCG systemin 3D modelling is the reduction of the rendering speed and the storage space. As the size of the city or the game increases, this effect becomes more significant. The RCG system can even further save the speed by creating or rendering only the building elements that are visible on the screen. Once the location and the angle of the view from the camera are detected, the DBGRcreates only buildings in the effective area or delete any buildings or structures out of the area.



(a) (b)

Fig. 9 visible Elements from first person view.

B. Variety

The RCG system can generate a multitudinousnumberof solutions by intermixinga few shape grammar rules and avaiable building elements. For instance, ifthere are three different building elements, which are door, window and roof, and each element has four types of doors, three types of windows and five types of roofs, then the possible solutions will be:

4 type_window x 3 type_door x 5 type_roof = 60 possible combinations

If we count all the generation rules and considervarious sizes, shapes and rotations, the solution numbers will increase dramatically. As Fig. 10demonstrates, the example building has four windows on each side of the building, four doors on the ground floor and one roof on the top. In this case,each window has four choices to select and there are 416 choices just for the window in one building. The same principles can be applied to a door or roof.

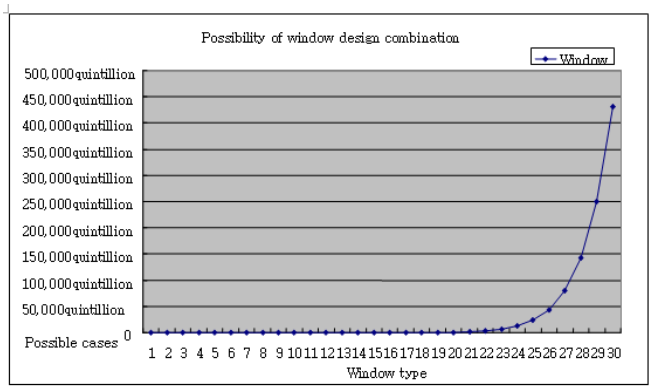
416 (window) x 34(door) x 51 (roof) = 65,568 cases

Fig. 10 The total number of possible combinations in a building

If the RCG system manipulates building elements ina high LOD, thetotal number of possible solutionswill increase even more dramatically. This is becausea higher LOD involves more building elements and therefore more combinations. An example case for the Greek style building is the below,Fig.11.

44 (windows) x 33(doors) x 51 (roofs) x 318 (columns) x 31 (stairways) = 387,420,779 possible solutions

Fig. 11 Possible solutions in a high LOD



Graph indicates how the solution numbers increase exponentially when the number of window types increases.

VII. CONCLUSION

3D modelling generally involves a complex and expensive process. In this paper, the RCG system was introduced to real-time 3D computer games. The major features of the system werecompared with the manual production method and itsusefulness was discussed in terms of time and cost efficiency, and variety. This conceptual model wasparticularly emphasized on the generation phase of the system. The RCG library was designed to contain all the necessary elements for the city generation process, including 3D building elements, shape generation rules and evaluation criteria. These elements were saved in a separate external file.

The implementation of the RCG system demonstrated many advantages over the manual modelling method, in terms of modelling speed, variety of solutions and less storage required. This system will be further developed by improving evaluation criteria.

REFERENCES

[1] H. H. Chau, X. Chen, A. McKay, and A. D. Pennington, "Evaluation of a 3D Shape Grammar Implementation", In &Gero, J.(Ed.). Paper presented at the Design Computing and Cognition'04, Massachusetts Institute of Technology, USA, pp. 357-376, Jul. 2006.

[2] U. Flemming, "More than the sum of parts: The grammar of Queen Anne houses". Journal of Planning B: Planning and Design, vol. 14, no 3, pp. 323-350, 1987.

[3] T. Knight, *Applications in Architectural Design, and Education and Practice*, Cambridge, Massachusetts: The MIT Press, 1999.

[4] T. Knight, "Shape Grammars in Education and Practice: History and Prospects", The International Journal of Design Computing, vol. 2, 1999-2000.
[5] H. Koning, and J. Elizenburg, "The Language of the prairie: Frank Lloyd Wright's Prairie Houses", Journal of Environment and Planning B, vol. 8, pp. 295-323, 1981.
[6] T. Ming, "City Generator, GIS driven Genetic Evolution in Urban Simulation", International Conference on Computer Graphics and Interactive Techniques archive SIGGRAPH '09, Louisiana, vol. 62, no 3, pp. 366-377, 2009.
[7] P. Muller, P. Wonka, S. Haegler, A. Ulmer, and L. Gool, "Procedural Modelling of Buildings", Journal of ACM transactions on graphics, vol. 25, no 3, pp. 614-623, 2006.
[8] P. Muller, G. Zeng, P. Wonka, and L. V. Gool, "Image-based Procedural Modelling of Facades", Journal of ACM transactions on graphics, vol. 26, no 3, pp. 85-93, 2007.
[9] A. Rollings, and E. Adams, *Andrew Rollings and Ernest Adams on game design*. USA, New Riders, 2003.
[10] G. Stiny, "Kindergarten grammars: Designing With Froebel's Building Gifts", Journal of Environment and Planning B: Planning and Design, vol. 7, pp. 409–462, 1982.
[11] G. Stiny, and J. Gips, "Shape grammars and the generative specification of painting and sculpture" in *Proc.* the Information Processing71, Amsterdam, North Holland Publishing Co., 1972, pp.1460-1465.
[12] G. Stiny, "Ice-ray: a note on the generation of Chinese lattice designs", Journal of Environment and Planning B: Planning and Design, vol. 4, no 1, pp. 89-98, 1977.
[13] M. Tapia, "A visual Implementation of a Shape grammar system", Journal of Environment and Planning B: Planning and Design, vol. 26, no 2, pp. 59-73, 1999.
[14] S. Tsiptsin, *Understanding Maya* by, USA, ArtHouse Media, 2007.
[15] B. Watson, P. Müller, O. Veryovka, A. Fuller, P.Wonka, and C. Sexton, "Procedural Urban Modelling in Practice", Journal of IEEE Computer Graphics and Applications, vol. 28, no 3, pp. 18-26, 2008.
[16] J. Gips. "Computer implementation of shape grammars", NSF/MITWorkshop on Shape Computation, 1999.