A Collaborative Environment for Urban Landscape Simulation

Pedro Brandão Silva
INESC Porto
Departamento de Engenharia Informática, Faculdade de Engenharia, Universidade do Porto,
Rua Dr. Roberto Frias, 4200-465,
Porto, Portugal
pedro.brandao.silva@gmail.com

António Coelho
INESC Porto
Departamento de Engenharia Informática, Faculdade de Engenharia, Universidade do Porto,
Rua Dr. Roberto Frias, 4200-465,
Porto, Portugal
acoelho@fe.up.pt

Rosaldo J. F. Rossetti
LIACC
Departamento de Engenharia Informática, Faculdade de Engenharia, Universidade do Porto,
Rua Dr. Roberto Frias, 4200-465,
Porto, Portugal
rossetti@fe.up.pt

***Abstract*— Digital maps are very commonly used in the analysis, administration and representation of urban spaces. Despite the degree of information they can provide, they still pose some difficulties in decision making due to the three-dimensional nature of such ever-changing settings. This paper presents a collaborative solution for large virtual environment recreation aimed at urban landscape simulation. By providing an extensive interactive three-dimensional virtualization of a real-world city, multiple users may contribute with additional data, to either improve the model fidelity or to preview the impact of certain urban changes. This is achieved by employing procedural modeling methods to generate a basic three-dimensional city model from real-world data and a set of parameters, which can later be refined through human intervention. A multi-user simulation platform has been conceived, allowing the collaborative management of the model data, as well as the simulation of possible urban landscape changes, introducing therefore more advanced analysis and representation features to study and discuss the impact of certain decisions on the urban landscape.**

Visual Interactive Modeling, Visual Interactive Simulation, Procedural Modeling, Urban Environments

I. INTRODUCTION

The creation of virtual urban environments, corresponding to real-world settings, has become a very important subject of development in virtual environment simulation applications. By reproducing highly detailed structures through interactive three-dimensional models, they can serve multiple activities such as urban planning, training and education, among others.

Their extensive and complex nature, however, makes their production very expensive and time-consuming, demanding many researchers to look for semi-automatic methods to develop them. By querying real-world data sources, such as geographical information systems (GIS), the process of building such models can be automated, reducing the need for human intervention. However, such sources are not often detailed enough to obtain accurate models, which makes it imperative to use empirical data on the urban environment to amplify the existing information. While this might be an exhausting task for an individual alone, when shared with multiple co-workers, not only it can be done faster, but also more accurately, since it introduces the possibility for group discussion. Ultimately, this can lead to a greater level of fidelity.

Modeling virtual urban environments may constitute an end itself, as long as urban planning is concerned. An architect and urban planner may be interested in simulating the construction of a new building and see how it looks like when placed in the urban landscape. Police, Fire and Health Departments may obtain information to outline new strategies or discover possible problems related to building access, when emergency situations in the building arise. Neighboring residents may as well evaluate the construction and comment on how it may end up improving or interfering with their life standard.

If the building models are finished and accurately represent the real world, they can serve additional purposes. Advertising, for instance, by letting visitors learn about the location of certain services; virtual tourism, serving as a means for visitors to get to know other world locations much quicker, from their visual appearance to any related background information; training, such as military, flight, driving, etc. that take place in such urban settings, but may be too dangerous to be performed in the real counterpart; and other studies that may use urban environments as their primary scenery. Further motivations and uses had also been envisaged in [1].

Despite the potential use of three-dimensional virtual urban environments for analysis, discussion and management, no approach has yet focused on fully addressing these possibilities. This can be explained by the difficulties that arise when coping with the extensive amounts of information that these virtual environments may contain. The high number of detailed and textured models would require elevated bandwidth to download and high computing specifications for any user to be able to manage their visualization.

Another problem lies on the collaborative edition of the urban features, a task that should be easy to handle by any user. Yet, it should be powerful enough to model complex designs if desired. In order to allow real-time collaboration, the created features should be synchronized among all participating users immediately. In the end, if the results are to be used for further studies or simply to amplify the existing information, there is an additional need to somehow map the edited information into other data formats.

To address these issues, this paper presents a collaborative solution, allowing the recreation of real-world virtual urban environments, aimed at the simulation of urban landscapes. This relies heavily on the application of procedural modeling methods already developed in previous works [2], [3], integrated with an urban ontology described in [4] that allows the management of the urban data. The system follows the client-server architecture model. Users access the client side to view and interact with the virtual environment, which is generated on-the-fly on their machines. Parametric information describing the urban features is transmitted from the server to the clients, and is then used by the procedural modeling processes to generate the three-dimensional environments. These parameters can then be altered by the users to simulate urban changes or to improve the visual fidelity of the model. The concept and initial motivation for this project had been initially proposed in [1].

The paper is structured as follows. In the next section, some related work is presented. Afterwards, the solution is described, detailing its architecture and features, with a special emphasis on the collaborative and procedural modeling approaches. Some information regarding the implementation comes next, followed by the results section, which includes their discussion. Lastly, the conclusions are presented and some future work is suggested.

II. Related Work

This project fits into the scope of procedural modeling of urban environments, a research area which already possesses some interesting approaches, some of them dedicated to specific domains. Regarding the modeling of terrains, many authors have worked on three main approaches for their generation: fractal landscape modeling [5–7], physical erosion simulation [8], [9] and terrain synthesis from images or sample terrain [10], [11].

For road creation, most notable works include the use of L-Systems [12], [13] or tensor fields for its generation [14], and its necessary adjustment to the generated terrain [15], [16]. Many of these introduced a more interactive way to define and manipulate roads, by allowing the change of their parameters while viewing their results in real time.

When it concerns the modeling of buildings, a pioneer work is the one by Wonka et al. [17], introducing the split grammars, a new type of parametric set grammar based on the concept of shape brought up by Stiny and Gips [18], [19]. Based on this work, Müller et al. [20] developed the CGA Shape, a shape grammar capable of producing extensive architectural models with high level of detail. The implementation of the CGA Shape is integrated in the CityEngine framework [21]. Considering that text-based shape definitions were impractical to use by most artists, Lipp presented in [22] a visual editing system that introduced traditional modeling techniques, allowing a more intuitive and powerful control over each grammar aspect. To improve the process of creating building façades, Müller [23] presented a method that operates based on real world photographs. Finkenzeller [24], on the other hand, focused on more complex and less common façades that were not contemplated by the previous author. Focusing more on the use of real-word GIS data support, Coelho [25] presented in his work the Geospatial L-Systems, an extension of the parametric L-Systems that incorporates spatial awareness. His solution is integrated in a modeling tool called XL3D modeler, which provides interoperable access to various sources of information, allowing the reproduction of real urban environments. Concepts of both the CGA Shape and the Geospatial L-Systems served as the major inspiration for the work on the PG3D system [2]. Its fundamental design consists in performing procedural modeling directly on a spatial database management system, where the geographic data sources and the created models are saved. The modeling operations are built as stored procedures within the database, as a means to reduce the time to access the data.

Concerning collaborative solutions for world design, the most popular approaches are Google Earth, Second Life or OpenSim, which allow to recreate world scenes in 3D, offering their users an intuitive and collaborative environments. Google Earth [26], from Google, intends to be an application that can represent the whole world virtually. It is essentially based on aerial photographs and geographical information, but does provide, for certain areas, 3D representations, which can be extended and improved by any user through additional tools. Examples are Google SketchUp [27], a 3D modeling software, or Google Building Maker [28] which introduces multiple photograph junctions to create building textures, adjusting them to a coarse building model.

Second Life [29] is a collaborative virtual environment from Linden Lab, which intends to be a simulator of the real life with a high degree of interaction. The virtual environment allows the users to perform a total customization of their avatar and profile, as well as their interaction through chat, voice or avatar gestures. It is also possible to build almost every type of objects through the manipulation of simple primitives. OpenSim shares many similarities with Second Life, but has the advantage of allowing users to create their own servers; it also offers more flexibility in the development of further features due to its open-source nature [30].

Current existing tools on collaborative real-world recreation, such as Google Earth and Second Life already address the problem of large urban management using space partitioning techniques. Still, they focus mostly on the manual modeling of single structures, such as historical monuments or other touristic attractions, and less with the urban landscape as a whole. Also, users are faced with manual geometrical authoring tools (which might be hard to operate by the untrained user), producing models that fit on the environment, but lack a semantic description. As a result, the produced models are prone to two fundamental limitations: 1) they become harder to manipulate any further and 2) they cannot be

used to organize, integrate and amplify the information in commonly used digital maps.

The presented procedural approaches, on the other hand, are rather concerned with the mass modeling of the landscape as the whole. They can operate based on the information available in digital maps, and can also be used to amplify it. Since the modeling processes are guided by parameterized rules, it is possible for any user to achieve specific results by simply changing some values. Still, existing tools neither allow multiple user collaboration nor offer proper strategies to manage large urban areas as yet. This motivates for a solution that can address these missing features to allow a more manageable urban landscape simulator.

III. THE COLLABORATIVE SOLUTION

The proposed solution enables the recreation of real-world virtual environments for the simulation of urban landscapes. Users are able to visualize, navigate and interact with large areas of urban terrain, improving the existing data or simulating possible changes to observe their impact on the urban landscape. The main characteristic that differentiates this system lies on the introduction of procedural methods to collaboratively manipulate the virtual models. Contrary to the usual application paradigm of procedural modeling, this approach does not consist in generating a priori the urban models, which are then fed to the visual simulators. Instead, the models are produced in real-time on smaller areas around the users based on parameters stored on GIS databases. These parameters can be edited to produce varied results. This approach intends to solve the various issues concerning bandwidth and computing usage, user interaction usability and the integration with existing information.

A. *Architecture*

In order to introduce the support for multiple users, a client-server architecture has been adopted for the conceived landscape simulator (see Fig. 1).

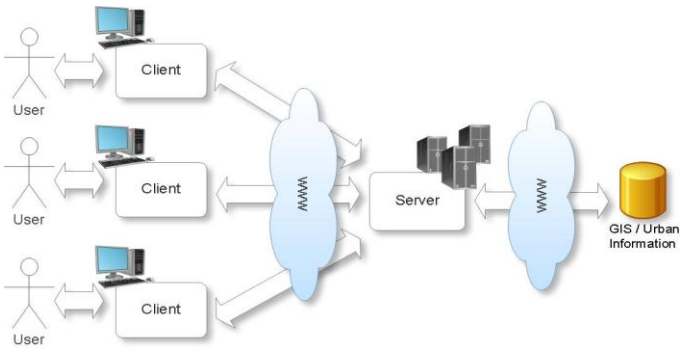

Figure 1. Client-server system architecture.

The client application is the main "window" of the system, with which the users interact. Visualization and input is handled on this side, which then communicates with the server. As for now, the considered target platforms are general-based computers; nonetheless future client platforms may include dedicated gaming consoles or mobile devices.

Acting as a central hub for all incoming client communications is the server, which contemplates an urban ontology, used to map multiple (local or remote) GIS databases into a single unified urban model [4]. This model contains parameterized information about the urban features, organized by their semantic relevance, spatial reference and feature type (e.g. buildings, windows, streets, trees). This approach makes it easier to handle and export such parametric data so as it can be used in further tools or studies.

The server is also the responsible for the management of user details, such as identification, location, direction, chat messaging and all kinds of features allowing multiuser support, as it is common in most multiplayer online games. In addition, the server's purpose resides on the management of user contributions to the urban model, by controlling its concurrent and asynchronous nature and by relaying the changes to all users.

B. *The Procedural Modeling Process*

The modeling of virtual urban environments replicating real environments leads to the need for data sources which describe the various features that compose them, such as its location, height and appearance. Due to the variety of sources in which such information may be stored, it is essential that they can be connected somehow by having some common reference, namely a georeference, i.e. its location on the earth's surface. Each urban feature (e.g. building, street, tree) has a geographical feature (e.g. a point, line or polygon, according to the Simple Features specification [31]), with which additional parametric data can be associated. Spatial databases that hold such features can then perform spatial queries to obtain urban features located at a certain location.

The modeling process, more thoroughly explained in [2], [3], can be summarized as follows. When a user visits a certain location on the client application, all the features surrounding the area are requested to the server, which then looks ups in its unified data model. There, spatial queries are performed to obtain all the features of the indicated types. Once this information is returned to the client, it is processed according to procedural rules. A simple example of how the modeling process can occur is depicted in Fig. 2, which shows how a 3D model of a building is generated from a set of parameters.

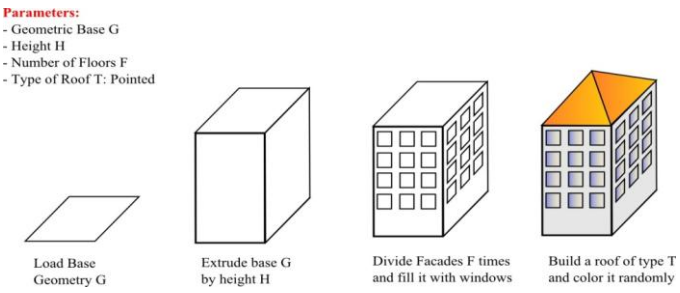

Figure 2. Procedural Modeling Process. Considering a set of parameters, three-dimensional building models are generated using those parameters and a certain degree of randomness.

One of the greatest advantages of this approach lies on the reduced data overhead. While storing and sending large model meshes would be unbearable for such great urban areas,

reducing the information to single parameters greatly reduces the need for data storage and transfer. On the other hand, changes are easier to perform, since a small set of the data is enough to carry out great modifications.

C. Progressive Area Loading

Urban settings cover usually extensive areas of terrain. Working on such great spaces in one go requires large quantities of geospatial data and parameters which, when used for three-dimensional model generation, result in elevated memory costs that may not be feasible and attainable by any commodity machine. Likewise, rendering these large areas at once may not be possible when real-time interaction is intended. This demands a strategy to find an optimal memory and graphical resource use based on what the user should be able to see at once.

The current solution introduces the concept of sector, which corresponds to a unit division of the world space. Each sector is described by a square of certain side length, which can be adjusted based on the memory and CPU of the client's machine. They may contain several pieces of urban GIS data concerning the various kinds of urban elements: buildings, streets, terrain elevation, etc. Entities that are located between more than one sector are stored as links in the others.

The loading process succeeds as follows: once the user logs into the system, his last position will be obtained. Using that position, the corresponding containing sector – the central sector – will be loaded, along with the 8 neighboring ones (see Fig. 3). These are the ones that will be visible to the user at once. When the user navigates out to a neighboring sector, this will become the central one, and the 8-sector loading process will take place again. This greatly reduces the area to be loaded and rendered at the time.

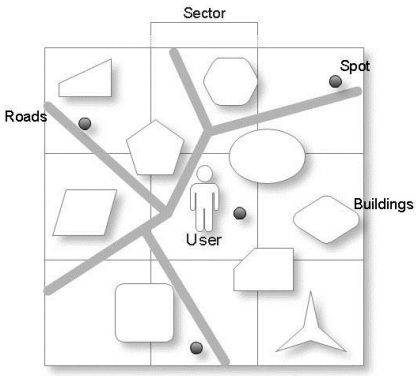

Figure 3. World division in a central and 8 neighboring sectors.

Sector loading is possible through the mentioned spatial queries which allow data to be loaded based on their geographical position. These queries can be indexed according to the sector division, introducing therefore accelerated accesses. Still, a sector caching strategy is used, meaning that recently loaded sectors will remain in memory, although they are not rendered unless it is necessary. This way, if a user navigates back, loading can be processed faster. If the user navigates to a distance greater than X (where X may be configured according to the used machine), the sector is discarded from memory, and will have to be fully loaded the next time it is visited.

After the sector information has been loaded, the data is used for the procedural generation of the three-dimensional models (as explained in the previous section), and is linked to the sectors they are inserted in. This is a purely client-side process, meaning that the server has only the task of loading the sector data from the database and transmiting it to the clients, which will manage the generation and the rendering component.

In order to avoid long loading and generation times, all these processes are meant to occur mostly in background threads. While an initial waiting time is inevitable, the loading and generation of the remaining models occurs while the user is interacting with the already loaded ones, without interruption, which leads to an improved experience and more fluid navigation.

D. The Collaborative Modification Process

When connected to the system, users are able to see each other and communicate among themselves. They are managed in a similar sector-oriented fashion, in order to introduce scalability and easier control. The main objective of the system lies obviously on the possibility of changing the various features of the urban environment by editing their generation parameters.

By default, users have a first-person camera view of the world, which they can control to navigate around the environment. When an urban element is to be edited, it can be selected and its properties changed. During the process of edition, the element remains locked, meaning that other users cannot change it at the same time. When the user is finished and chooses to save the accepted changes, the edited parameter values are sent to the server, to update the unified model [4]. At the same time, users navigating at the sector or nearby will have this same element updated with the recent changes. Users that do not have the affected sector currently loaded or cached do not need to have their sectors updated. Since the change will be processed on the server, they will obtain the latest changes the next time they visit that sector. Fig. 4 contains a visual explanation of the process.

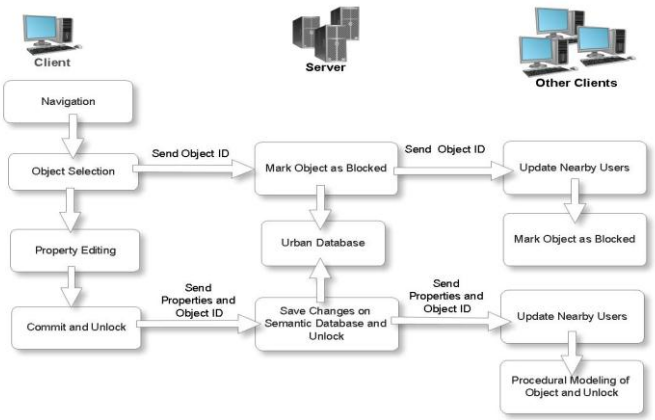

Figure 4. Client-Server communication concerning urban element edition.

Collaboration systems usually maintain a versioning system that allows them to keep a history of all performed changes. This works as a way to introduce undo and redo operations, but also to safeguard the system against possible vandalism. In the end, if the submitted changes have been verified and accepted as model improvements, they can be used to update the original data sources. This feature is integrated into the mentioned unified model and described more thoroughly in [4].

E. Implementation

Both client and server applications have been implemented in Microsoft's .NET C# 4.0. Its facilities concerning thread control and network communication, together with its managed and object-oriented nature have eased the development process.

As specified in [4], the GIS data is stored on a PostgreSQL database management system, enriched with the powerful features of geographic data management of the PostGIS extension. This has allowed data to be indexed based on the sectors, accelerating the access and loading processes.

In order to be able to visualize and interact with the virtual environment, the Microsoft XNA Framework [32], a popular game development framework, was used. This has allowed efficient processing and rendering of large numbers of models, while introducing a series of features that made the creation of an interactive application possible.

IV. Results and Discussion

At this stage of the work, both client and server applications have been implemented, as well as their communication protocol, allowing the interaction between the connected clients. An urban database featuring information on buildings, streets and sidewalks has been used as the main data source, whose content is shared only with authorized, registered users. Data load and model generation proceeds according to the sector division approach - users may navigate in the virtual world, and additional models are loaded in background, as the camera approaches them. This method has resulted in lesser CPU and memory strains, allowing the client applications to run smoothly even in less powerful machines. Rendering is restricted to a certain distance from the users, but still allowing them to visit the whole extents of the city (See Fig. 5).

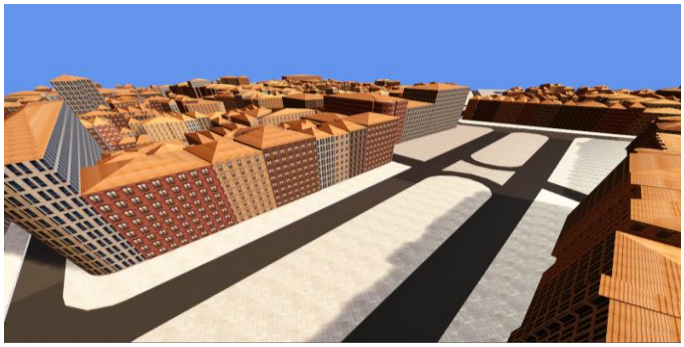

Figure 5. Overview of a section of the "Avenida dos Aliados", in the city of Porto, Portugal

Procedural modeling methodologies have been applied to generate the various urban elements, according to the available data sources, resulting in the generation of buildings, streets and sidewalks. While at this point only relatively simple geometric meshes have been obtained, their quality and level of fidelity are high enough to allow some test users to recognize many of the modeled settings.

Interaction with the generated urban elements has been implemented, introducing their parameterization (although still limited at this point), which is shown in Fig. 6. By default, a user views the world from a first-person perspective, using the keyboard to move and the mouse to control the camera direction. When an entity, such as a building, is focused, a second metaphor of interaction is introduced: a windowed GUI is presented, containing the various parameters that can be changed. A cursor allows the interaction with the window components, while the keyboard is used for value editing. When the parameters are changed, one can preview the effects and, when the intended result has been achieved, they can be stored and replicated through the server. During the edition process, the selected objects are given a whitish shade, to make their selection more noticeable (see Fig. 6).

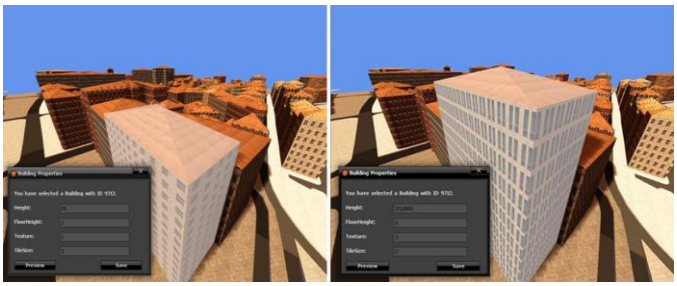

Figure 6. A building being modified by changing its corresponding parameters. On the left: the building when clicked the first time. On the right: after the height, texture, number of floors and number of windows per floor have been changed and the building has been regenerated with the new parameter values.

Users are represented in the virtual world by a simple avatar, being, at this point, essentially distinguishable only by their name and color. Although not being able to see their own avatars, they are perceived by other users that roam in the same area (see Fig. 7). At this point, the focus has been on providing location and direction awareness between users, allowing them, for instance, to make sure they are looking at the same objects while discussing it.

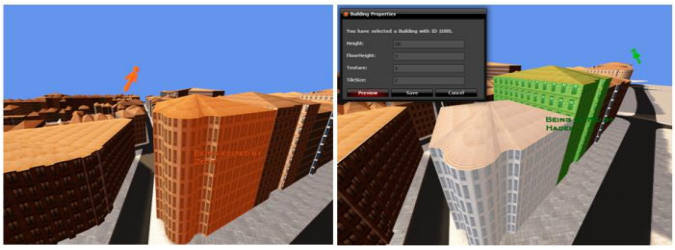

Figure 7. On the left: Building being edited by user "Zeus", as seen by user "Hades". On the right: Own building edition view led by "Zeus", while user "Hades" is changing the neighboring building.

Users can also see each other's effects on the environment. Once a user has started editing the properties of an object, it will become locked for the remaining users, therefore avoiding any possible edition conflicts. While an object is locked, the owner's color is used for shading that object on other clients, while displaying a warning message (see Fig. 6).

V. Conclusion And Future Work

In this paper, a collaborative solution for the recreation of virtual urban environments has been presented, aiming at urban landscape simulation. This is achieved by procedural modeling methods to generate on-the-fly a basic three-dimensional city model from real world data and a set of parameters, therefore reducing the data storage and communication overhead. By using a simulation platform following a client-server model architecture, multiple users can be connected to a shared virtual world. Here, they can collaboratively work on the urban landscape through a simple parameter editing interface, allowing them to contribute with their own knowledge of the city to refine the virtual 3D urban model.

The simulator includes a set of features that allows the collaborative work on the urban environment. While still lacking complex modeling capabilities, the system already presents high quality models that somewhat resemble real-world structures. Future work will concentrate on the improvement of the modeling rules that employ the available parameters (or even introduce some more). On the other hand, future plans include gathering a larger number of users, so that further urban information can be obtained. Also, their feedback on interactivity, visual performance and other issues will be considered in the process of improving the collaborative experience.

Although similar applications exist that allow this kind of virtual world edition, none actually makes use of procedural modeling as its main paradigm. This makes this kind of landscape simulation rather novel and therefore an interesting topic for further research.

Acknowledgment

This work is partially supported by the Portuguese government, through the National Foundation for Science and Technology - FCT (Fundação para a Ciência e a Tecnologia) and the European Union (COMPETE, QREN and FEDER) through the project PTDC/EIA-EIA/108982/2008 entitled "3DWikiU – 3D Wiki for Urban Environments" and through the Ph.D. Scholarship SFRH / BD / 73607 / 2010.

References

[1] M. Melo et al., "3DWikiU-3D Wiki For Urban Environments," in SIACG 2011, 2011, pp. 1-4.

[2] P. B. Silva and A. Coelho, "Procedural Modeling for Realistic Virtual Worlds Development," in SLACTIONS, 2010.

[3] P. B. Silva, A. Coelho, R. Rodrigues, and A. A. Sousa, "A Procedural Gometry Modeling API," in GRAPP 2012, 2012, p. 6.

[4] T. Martins, P. B. Silva, A. Coelho, and A. A. Sousa, "An Urban Ontology To Generate Collaborative Virtual Environments For Municipal Planning And Management," in GRAPP 2012, 2012, pp. 1-4.

[5] B. B. Mandelbrot, The fractal geometry of nature. Wh Freeman, 1983.

[6] G. S. P. Miller, "The definition and rendering of terrain maps," ACM SIGGRAPH Computer Graphics, vol. 20, no. 4, pp. 39-48, 1986.

[7] K. Perlin, "An image synthesizer," ACM SIGGRAPH Computer Graphics, vol. 19, no. 3, pp. 287-296, 1985.

[8] A. D. Kelley, M. C. Malin, and G. M. Nielson, Terrain simulation using a model of stream erosion, vol. 22, no. 4. ACM, 1988.

[9] F. K. Musgrave, C. E. Kolb, and R. S. Mace, "The synthesis and rendering of eroded fractal terrains," ACM SIGGRAPH Computer Graphics, vol. 23, no. 3, pp. 41-50, 1989.

[10] H. Zhou, J. Sun, G. Turk, and J. M. Rehg, "Terrain Synthesis from Digital Elevation Models," IEEE Transactions on Visualization and Computer Graphics, vol. 13, no. 4, pp. 834-848, 2007.

[11] H. Hnaidi, E. Guérin, S. Akkouche, A. Peytavie, and E. Galin, "Feature based terrain generation using diffusion equation," 2010, vol. 29, pp. 2179-2186.

[12] Y. I. H. Parish and P. Müller, "Procedural Modeling of Cities," no. 2001, pp. 301–308, 2001.

[13] G. Kelly and H. McCabe, "Citygen: An Interactive System for Procedural City Generation," no. 7, 2007.

[14] G. Chen, G. Esch, P. Wonka, P. Müller, and E. Zhang, "Interactive procedural street modeling," ACM SIGGRAPH 2008 papers. ACM, Los Angeles, California, pp. 1-10, 2008.

[15] E. Bruneton and F. Neyret, "Real-time rendering and editing of vector-based terrains," Computer Graphics Forum, vol. 27, no. 2008, pp. 311-320, 2008.

[16] E. Galin, A. Peytavie, N. Maréchal, and E. Guérin, "Procedural generation of roads," 2010, vol. 29, pp. 429-438.

[17] P. Wonka, M. Wimmer, F. Sillion, and W. Ribarsky, "Instant architecture," in ACM SIGGRAPH 2003 Papers, 2003, vol. 22, no. 3, pp. 669-677.

[18] G. Stiny and J. Gips, "Shape Grammars and the Generative Specification of Painting and Sculpture," in Information Processing '71, 1972, pp. 1460-1465.

[19] G. Stiny, "Introduction to shape and shape grammars," Environment and Planning B, vol. 7, no. 3, pp. 343-351, 1980.

[20] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. Van Gool, Procedural modeling of buildings, vol. 25, no. 3. ACM, 2006.

[21] Procedural Inc., "3D Modelling Software for Urban Environments," Procedural, no. 26/2/2011. 2009.

[22] M. Lipp, P. Wonka, and M. Wimmer, "Interactive visual editing of grammars for procedural architecture," ACM SIGGRAPH 2008 papers. ACM, Los Angeles, California, pp. 1-10, 2008.

[23] P. Müller, P. Wonka, G. Zeng, and L. V. Gool, "Image-based Procedural Modeling of Facades." 2007.

[24] D. Finkenzeller, "Detailed Building Facades." IEEE Computer Society, Karlsruhe, pp. 58-66, 2008.

[25] A. Coelho, M. Bessa, A. A. Sousa, and F. N. Ferreira, "Expeditious Modelling of Virtual Urban Environments with Geospatial L-systems," Computer Graphics Forum, vol. 26, no. 4, pp. 769-782, 2007.

[26] Google, "Google Earth." 2010.

[27] Google, "Google SketchUp," no. 31/8/2011. 2011.

[28] Google, "Google Building Maker," no. 31/8/2011. 2011.

[29] Linden Research Inc., "Second Life," 2012. [Online]. Available: http://secondlife.com/.

[30] OpenSimulator Community, "OpenSimulator," 2012.

[31] Open Geospatial Consortium, "Simple Features." 2010.

[32] Microsoft Corporation, "App Hub," no. 18/6/2011. 2011.