USE OF A REAL TIME 3D ENGINE FOR THE VISUALIZATION OF A TOWN SCALE MODEL DATING FROM THE 19TH CENTURY.

Pascal HUMBERT[1], Christine CHEVRIER[1] and Didier BUR[1]

[1] MAP-CRAI Centre de Recherche en Architecture et Ingénierie

École Nationale Supérieure d'Architecture de Nancy
2 rue Bastien-Lepage – B.P. 40435 54001 Nancy Cedex - France

humbert,chevrier,bur@crai.archi.fr

Keywords: architectural heritage, laser scanning, game engine, town scale model

Abstract:

In this paper we present a research related to the virtualization of an old town scale model. This is an experimental work that mainly focuses in establishing a valid workflow from data remote sensing to the real-time on-line walkthrough in the 3D model.

The real scale model encompasses the town itself and the surrounding countryside. Because the scale model is kept safe in containers, nobody has access to this information. Providing an on-line virtual model will unveil both the scale model and its attached information to the general public. In this paper we present the application we have developed for the visualization of the virtual model, using a game engine. After an introduction, each of the following topics will be developed:

Choice of Unity software: although Unity is a real time 3D engine devoted to video game development, we justify our choice to develop in this environment and explain how its API is appropriate to achieve our goals. Terrain: scanned with a hand-held laser scanner, we focus on how we have overcome some difficulties to optimize the terrain management. Buildings: when some geometry is imported in Unity, new textures are created. We have developed a modeling environment so that all the buildings easily share a set of common textures. Vegetation: the challenge was to create new kinds of vegetation that look like the ones on the scale model, that's-to-say not so realistic. We also have developed a tool to "plant" vegetation on the terrain at accurate locations.

Results and future work: we present result analysis and extrapolation of the workflow to the entire scale model that is twenty times greater in size than the part we have modeled. The workflow has been set keeping in mind that the visualization of the whole scale model on the web will be a challenge: 4000 buildings and more than 870 000 vines on a large terrain.

1. INTRODUCTION

Scale models of towns were realised and used in the past mostly for studying military strategies. It was almost the only way for the authorities to have a better understanding of the local peculiarities of the topography, urban specificities and to define attack and siege tactics. They are now museum items and can be seen as a quite accurate depiction of cities as they were at that time, more understandable by the general public than plans, paintings or engraving. Comparing cities then and now, adding information to landscape and buildings by means of virtual reconstructions of such scale models is a convenient way to widely spread this cultural heritage knowledge side.

2. RELATED WORKS

Several works that undertook the modeling of scale models are well known. The modeling of Prague scale model by Antonín Langweil in 1826-1837 is one of these projects [1] [2]. It involved a large team. The relatively small size of parts eased the digitizing (1.6 x 1m for the biggest at a scale of 1/480). It is in much better condition than our model of the city of Toul (France) but the building textures contained thin structures of drawing in Indian ink making textures and their digitalization more delicate than ours.

In the 3D modeling of Beijing city scale model [3], the authors used stereo image pairs of the object to enhance the quality of edges of the scanned model. The point clouds are improved but no 3D geometric modeling is made and no semantic is given to the model. In our project each building needs to be semantically identified for the ultimate web application.

In the 3D modeling of Rome scale model [4] [5], models of complex buildings are manually created [6] and the others are computed with the help of procedural and parametric modeling techniques [7] [8].

Among these projects, only the Prague application is completed, others are still in progress. Prague visualization is performed via a CD-Rom so there is no real-time via the web. For the Rome project, the team will have to handle large databases. Virtools has been used to realize some tests on-line

Since a few years, numerous works have explored the use of game engines for interactive visits of 3D architectures with semantics [9], terrain design, and collaborative design. In [10], the didactic potentials of 3D interactive exploration were particularly emphasized and a learning system proposed.

3. PROJECT DESCRIPTION

The Museum of Plans-Reliefs located at Les Invalides in Paris and the SRI (regional centre for cultural heritage inventory) of the Lorraine region commissioned us the creation of a virtual remote accessed 3D object of a physical scale model dating from the 19th century: the scale model of the town of Toul in France (see Figure 1). Its overall surface is about 39 m^2 and it is composed of 20 parts (called tables). The biggest table contains the town (2.31m x 2.23m). The other tables contain hamlets and countryside. Among the 111 existing scale models, only 28 of them have been laser-cleaned and restored; they are exhibited at the museum in air-conditioned glass cases under specific lighting conditions. Unfortunately the one of Toul is kept in chests, consequently nobody has access to it. Art and architectural historians cannot access to all the precious information contained in such a model. So the aim of the project is first to give everyone the possibility to view a digital facsimile of the real scale model, it would also provide access to numerous detailed information such as plans, elevations, photos, texts, and so on.

This virtual model has to be accessible via the Internet by the largest public, allowing the end user to access parts usually invisible and offering an interface tool for documentation retrieval, regularly enriched and updated.

Several kinds of difficulties appeared:

Difficulties due to smallness: the scale model is at the scale of 1/600: such a small size is not compatible with accuracy, especially when dealing with architectural models. Buildings are a few centimeters high and openings are generally about one millimeter wide. The narrowness of the streets and the density of urban blocks made difficult the visual approach to some parts to take pictures or for the laser scanning.

Even at the smallest laser scanning resolution, hard edges were rounded out in the resulting 3D mesh. Beside the hand-held laser scanning, pictures were taken in high view angle (touching or getting close to the scale model was forbidden), making difficult the positioning and sizing appreciation of the openings.

Difficulties due to inconsistencies:

The physical scale model is neither a perfect depiction of the 2D elevations and plans that were first drawn prior to create it, nor a correct representation of the reality as it was when it was built. Numerous architectural inconsistencies occurred during the physical creation of the scale model in 1840. Errors also come from the methods used during the survey: at that time, lengths were measured in the real city using feet as unity of measure and the heights were estimated following a simple and visual proportion principle. So elevations were visually drafted: openings weren’t measured but only proportionally sketched. Also during the realization of the scale model, manual cuttings of the pieces (lime-tree wood, paper) used for the houses

contributed to multiply the inaccuracies (that's why walking in the virtual 3D model as a pedestrian is not desirable).

Difficulties due to the poor condition of the model:

Lots of paper pieces, representing house fronts, are unstuck or torn, notably modifying the buildings geometry and the opening shapes. Any mistake or imprecision becomes instantly obvious when one wanders in the virtual 3D model without scale and especially if one walks on the ground as a human being. The change of scale increases the architectural incoherencies: one millimeter error on the scale model corresponds to 60 centimeters in reality. Lighting conditions and restrictions (usually no more than 50 lux allowed) and dust deposit made the pictures difficult to take and tedious for photo-modeling use.

For all these reasons, both 3D and 2D data had to be carefully checked to determine if they were reliable. The final result of this experimental modeling was to produce a reconstructed model of the scale model, not as close to the real thing as possible, but sufficiently accurate to depict it and interactive enough for a real-time Web access.

Ergo this is not only an archiving process (laser scanned data has been stored separately) but also a proof of concept to set up a valid workflow in case of a complete modeling of the whole scale model (Figure 1) or of other scale models of the collection as well.

For more information about the creation of the 3D model you are invited to read [11] [12].

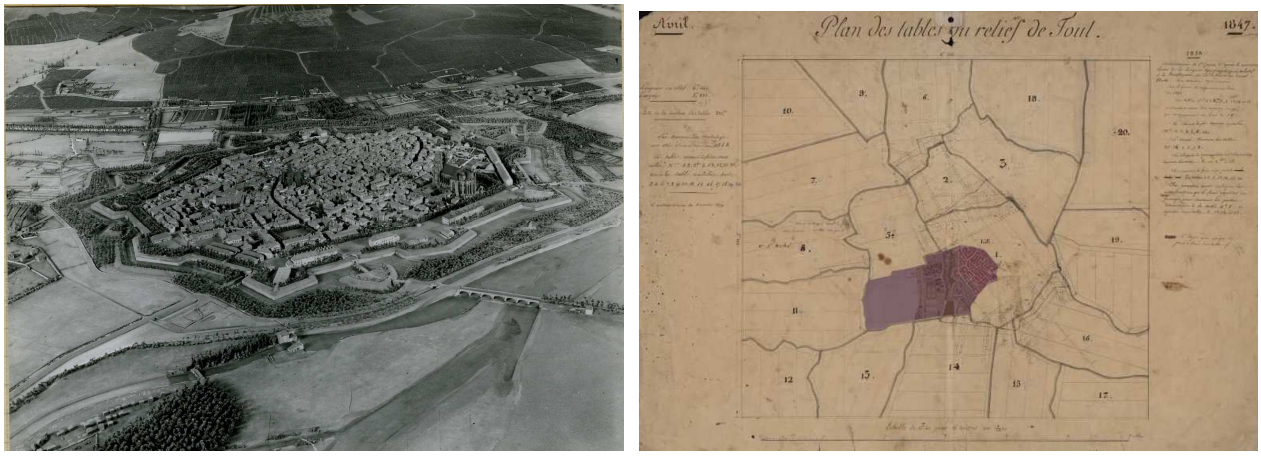

Figure 1: a) Picture of the physical scale model of Toul. b) Map of the tables of the scale model, filled areas are those modeled.

4. UNITY SOFTWARE

Unity, by Unity Technologies [13], is a real time 3D engine originally and specifically devoted to video game development (online or standalone cross-platform games). Among other advantages, it can handle huge databases, stream geometric asset bundles, pre-calculate lighting or perform real-time lighting, provide interactions and animations capabilities, attach behavior scripts to objects, while preserving a fluent visualization of the final product.

Unity applications can be played on the web via a cross-browser free plug-in on PC-Mac-Unix environments (as well as iOs, Android, Wii and Xbox consoles), so there is no need to write our own visualization software.

The development environment abstracts away the majority of platform differences, as it uses both Javascript and C Sharp coding languages. Building the final executable for any Web browser does not make any difference in the coding phase.

Our project needs such characteristics: the scale model is composed of more than 38000 pieces of vegetation, near 1000 buildings, which represents the twentieth of the whole model. Along with Unity's programming environment, it turned out to be the most convenient and efficient software.

Having written scripts helped us to simplify the handling of the project, facilitated the gathering of external data from various sources, and optimized the size of the database as well as the final application executable file and finally automated the modeling as much as possible.

5. THE TERRAIN

Unity terrain engine has an editor to create realistic terrains with sculpting tools and provides many ways to texture and populate the ground base of the scene, but these are mostly dedicated to natural landscapes. As we had to deal with a very particular and partially urban environment (many streets and houses, fortifications, rivers, bridges…), these built-in tools appeared at a first glance to be quite inaccurate and almost useless.

The global mesh acquired with a HandyScan laser scanner [14] was then cleared of all vegetation and man-made objects, and holes were filled. A script was developed to generate a precise height map of the landscape layer, based on this simplified mesh. Even at the highest resolution available, this Unity terrain turned out to be lacking of precision, but was usable for planting vegetation (see paragraph 7). Moreover, from a technical point of view, a first trial to import the overall terrain as a so-called "Unity mesh asset" quickly showed some issues: a 3D mesh cannot contain more than 65000 polygons and texture mapping is nearly unmanageable (no UV mapping possible), among others. So the entire mesh had to be split in two parts (one for each table), only slightly increasing rendering time and not noticeably slowing down the visualization. UV mapping texturing can hardly be completed within Unity, so the texturing of these two meshes was realized in the Maya environment, thanks to high-resolution top-down straighten photographs of the scale model, seamlessly mapped on each part. That is why our terrain consists of two layers: an invisible native Unity terrain (used for vegetation) placed just below the cleaned textured mesh parts. For the countryside table part, where accuracy is less important, the terrain model is a simplification of the digitalized mesh by means of a vertex-clustering process. For the town table, streets were also replaced by a decimation of the original mesh on their area.

6. THE BUILDINGS

The town table mainly contains the fortifications surrounding the city, streets and about thousand houses. In order to reduce the number of polygons and sharpen the ramparts walls, a model of the fortifications was redesigned on top of the relevant laser data, with Maya software.

6.1. Blocks and houses geometry

For each city block and house, a semi-automatic modeling process combining several technologies was developed: roofs edges retrieval, parallelism and planarity corrections, overhangs calculations, base walls footprints calculations, all this leading to a set of parameters allowing the creation of the main geometry of each building [12]. Additional elements (chimneys, lonely walls…) are then added thanks to a dedicated utility we developed, combining the parametric model, photos, and a library of architectural objects and laser data to locate chimneys. The openings are a particular case: on one hand they cannot be retrieved from the laser data (they are made of pieces of paper glued on the façade) and it was sometimes impossible to get a picture of each façade due to the narrowness of streets and inner courts; on the other hand, the final model cannot contain one texture per façade (of the wall and openings). When it was impossible to get a picture of some narrow or inaccessible parts, we used the paper drawings of the city blocks to add 3D openings. At the end of the modeling process, each house is composed of a mesh for the walls, a mesh for the roof slopes, one or more meshes for the openings, one or more meshes for the chimneys. For Unity, we combined all these elements in a single mesh to make it a unique clickable object. We worked block by block, importing each block into Unity the usual way. A script is then run that removes duplicated materials and replaces them by shared materials, adds a collider to each house, assigns each house a unique identifier, and finally attaches a behavior script to its mesh renderer so it becomes highlight-able and clickable. Dwarf walls, fences, entrances gates and railings were created with the same method than the trees rows (see paragraph 7.2). Rare and specific elements were manually added with Maya: several parapet walls and bench terraces.

6.2. Textures

When some geometry is imported into Unity, no matter the source format, new shaders, materials and textures are created from the input file. Since each house, in the parametric modeling stage is output as a particular mesh with its own shaders, an additional texturing step is required if one wants this house to share its textures with all the other houses. That's why we developed a script so that all the buildings share a unique set of common textures, assigned within the parametric modeler by object type: walls, roofs, chimneys… No re-texturing of individual houses assets was required within Unity. This simplifies the transfer of the geometry, allows a faster on-line loading time, decreases file sizes and offers fluid real-time display: textures amount plays a great role in minimizing draw-calls to the graphic card. Since photorealism was not really the essential concern, we used basic diffuse shaders (no effects, no normal mapping, etc), just flat diffuse mixed texture maps to fake torn and dusty paper. If needed in the future, modifying the overall aspect of the model all over the project would require a modification of only a few parameters of these shaders and would be very quick. The real terrain on the scale model is covered with glued sand and sprayed silk bits in the crops areas and drawn paper in the streets and courtyards. The 3D meshes of the terrain were mainly textured using on top-down views of the scale model for the countryside parts.

6.3. Links and names

Besides geometry, data must be attached to any house or monument in the model, especially a link to additional information sources such as web pages. Each building, street, road or monument can potentially direct the user to a web page that contains more information about it, through a link.

The final virtual model is hosted on a French Ministry of Culture server, among the various databases related to cultural heritage. These databases may change over time, information is added on a regular basis, so web pages and URLs may vary. Since a link to such a web page can vary over time, no link data were hard-coded in the model itself, but each house was given a unique code that is hold in an external file, as an array of keys (houses codes), houses names, and values (corresponding URLs). The code is a text string output by the automatic process that creates the building geometry. The house name and the URL are entered manually. The house name appears at the bottom of the web player window when the mouse is over a building. When the user hovers a house, both the code and the corresponding name are retrieved (for instance "BLOCK 41 – HOUSE 16", "Apothecary house"), this name pops-up in the interface, and the corresponding URL is called and sent to the browser if the user clicks on it. This external file is a simple yet effective way to avoid compilation of the application each time a minor modification must be done.

7. THE VEGETATION

Seen the significant number of trees, three concerns had to be addressed: low-poly modeling, accurate and easy location of vegetation, fluent and correct display.

7.1 Low poly modelling

Unity comes with various realistic vegetation types and species (mainly trees and shrubs) in a built-in library, as well as a procedural tree creator (such objects are called 'prefabs'). Using trees in Unity is obviously intended for creating a realistic vegetal environment. However on the scale model, vegetation does not look realistic: trees and shrubs are made of twisted copper wire, hacked silk and small silk twisted ribbons (Figure 2). The challenge was to create other kinds of vegetation that look like the ones on the scale model. Six main tree species, two bush types and a wine stock were identified. Each of these comes in a variety of three subtypes. The Unity tree engine was used to create each of the types, with a low-poly modeling in mind. Trunk texture is a repetitive wire-looking sample, leave textures were captured and outlined from macro pictures of the real scale model. The hierarchy was kept as simple as possible, just adding some branches to the trunk and a few leave groups.

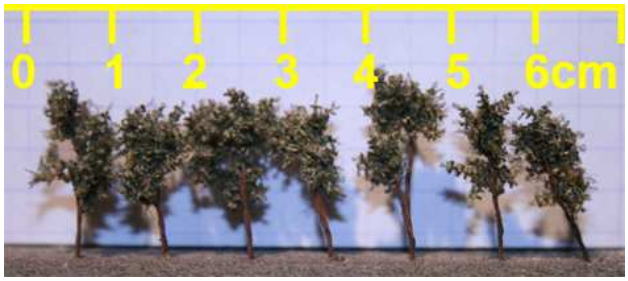

Figure 2: Trees on the scale model are made of twisted wire and silk.

7.2 Positioning the vegetation

A second problem was to accurately locate a bunch of vegetation pieces. The laser scanner data being awkward for that task, we developed a semi-automatic process that is able to generate or update all the vegetation of the terrain in a matter of seconds. First we used top-down photographs to manually digitalize lonely trees as points, ranks of trees as lines, and forests or groves as polygons, indicating the species and density to each geometric object as parameters. The resulting 2D data was then processed by a script to set the 3D coordinates of each trunk base, thanks to a top-down ray-cast method that intersect the terrain. Moreover, using this process overcame limitations of Unity terrain engine: one cannot accurately locate trees on a terrain with the native brush tools, which more or less randomly plants vegetation, and one cannot plant trees on other matter than a height map terrain. That's why our terrain mesh has a second invisible terrain underneath. This way, the tree prefabs can be easily modified and the whole vegetation could be instantaneously removed and updated if needed.

7.3 Use of billboards

A third problem was to preserve a fluid visualization of thousands of trees. To a certain extent, Unity comes to the rescue: one great feature is its ability to handle and display a hatful amount of vegetation pieces and to achieve that, the rendering engine is able to create on-the-fly a billboard for each vegetation 3D mesh. Terrain settings include thresholds to switch from 3D mesh representation to 2D billboards, transition distance, among others. Since 3D vegetation mesh instances are limited to 500, billboards are extensively used in our case because this parameter highest value is always exceeded. Two issues arose: first Unity produces a visible pop when switching from mesh to billboard representation, and secondly, the higher the viewpoint, the more the billboards seem to sink in the ground (because they are designed to be seen from a pedestrian viewpoint). As the number of vegetation elements is important (more than 38000) the distance threshold for swapping the representation is quite close to the camera. The changeover would then be seen if no particular attention was paid to the overall shape of each tree type. Finally, to avoid the 'sinking effect', a script was developed to raise the height map terrain as the camera viewpoint gets more and more top-down.

8. USER INTERFACE

8.1 User interface

Unity provides classes in a customizable interface to the GUI. The default skin has been used to select options and items from a pull-down menu: the user can examine the model in windowed or full-screen mode, choose a guided tour or freely inspect the model, display a map of the city or select a particular block to examine. Menu, options and map display are adapted automatically to any screen resolution both in the standalone application and in the web player. They were designed to take as less place as possible on the user screen.

8.2 Predefined walk-through

We have decided that the user would not be able to visit the 3D model walking in the streets or in the crops (Figure 3): having a pedestrian viewpoint doesn't make sense and could even give him a false interpretation of the scale model reproduction intents: the real scale model is always seen from a distance, say one half-meter. From that distance, the size of the openings for instance is very small. We consider that the virtual model must be seen at about the same scale to avoid misinterpretations of its accuracy. That's why the walk-

through are run from above. They follow major urban features (streets, ramparts, monuments). What is important for us is the global comprehension of the scale model at the urban scale, not the understanding of each constitutive object. Programmatically speaking, creating a new walk-through and its camera path just consists of digitizing waypoints on the map of the city and give that path a name. A script integrates it in the menu and if selected by the user, it smoothes the camera movement from a waypoint to the next. Two of such paths have been created to illustrate the thematic visits that will be set up on the complete model: ramparts and commercial street.

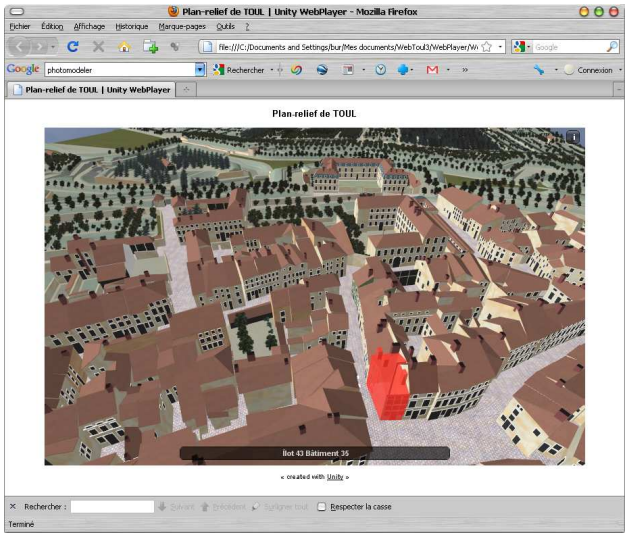

Figure 3: The virtual model displayed in a web browser.

8.3 Map locator

Each city block was given a number on the map. An option of the menu displays the original map of the city and a layout of clickable buttons to quickly move the camera near the corresponding city block. When the user clicks on a numbered button, a script searches in the assets which city block has such a number in its name, computes its bounding box and locates the camera with an offset so that the user sees the block full screen with the same orientation than the 2D map.

Thus, adding more city blocks doesn't need anything else than this generic locator.

9. RESULTS AND FUTURE WORK

Analyzing the process we have defined gives us some clues to further enhance the workflow when the whole scale model will be digitalized, having in mind that a completely automatic reconstruction is impossible, but diminishing phases where a human action is mandatory is envisioned.

An automatic segmentation of the point clouds (beyond the scope of this paper) as well as an image-based automatic positioning of the openings (currently under development). Keeping the data that are loaded in the web player as small in size as possible will induce streaming the 3D model as the user will visit it.

We shall probably define new 'game objects' structures to encapsulate existing ones ('tables', city, countryside), add some semantics to our objects (land and house use, still existing, construction date …) to be able to manage walk-through and thematic displays.

Other applications may arise: the scale model could be visualized on an interactive terminal in the museum, and a didactic game can be developed, featuring the principles we have developed in [10]. More visual information can be added to the model, for instance to compare the actual city to the former one, or to highlight elements thematically: agricultural areas and their types, particular topographic elements, use of buildings, military facilities and so on. Figure 5 below presents a comparison between Prague and Toul projects. It also presents an estimation for the entire scale model of Toul. The challenge will be to handle such a big model through the web.

	Physical model					
	Date	Scale	Surface area m²	Tables number	Material	Condition
Toul (actual)	1846-1861	600	1.88	2	wood, paper	poor
Toul (entire)			38.82	20		
Prague	1826-1837	480	20	52	wood, paper	good
	Digital model					
	Date	Houses	Openings	slopes	Chimneys	Vegetation
Toul (actual)	2009-2010	974	4788	2300	577	38 000
Toul (entire)	2012	3742	20 000 est.	9000 est.	2400 est.	760 000 est.
Prague	2006-2008	2500		13 400	9100	5400
	Staff	Working hours	Photos	Laser data	Triangles	Web player tris
Toul (actual)	6	5 000	2 000	0.5 Tb	11 000 000	300 000
Toul (entire)	6	30 000 est.	20 000 est.	10 Tb est.	200 000 000 est.	6 000 000 est.
Prague	62 - 110	18 000	244 000	T8 Tb		

Figure 5: Comparison between Toul and Prague projects, estimated data for the entire project.

10. CONCLUSION

In this paper we have presented an overview of the techniques we have used to reconstruct a virtual model of an old town scale model, from the initial laser scanning step to the final on-line application [15]. We have justified our choices to develop the product within the Unity environment, listed the difficulties we have encountered and explained how we overcame them.

The workflow we have defined, being perfectible, has been set up in a way it will be easily used to reconstruct town scale models that are twenty times greater than the part yet modeled, in a shorter period of time.

11. REFERENCES

[1] Scale Model of Prague: http://www.langweil.cz/index_en.php (accessed April 2011)

[2] Sedlacek, D. and Zara, J.: *Graph Cut Based Point-Cloud Segmentation for Polygonal Reconstruction*. In Proceedings of the 5th international Symposium on Advances in Visual Computing 2009: Part II, Las Vegas, Nevada

[3] Zhu, L., Ma, G., Mu, Y. and Shi, R.: *Reconstruction 3d-models of old Beijing city stuctured light scanning*, 22nd CIPA Symposium, 2009 October 11-15, Kyoto, Japan.

[4] Guidi, G., B. Frischer, De simone, M., cioci, A., Spinetti, A., Carasso, l., Loredana, l., Russo, M. and Grasso, T.: *Virtualizing Ancient Rome: 3D Acquisition and Modeling of a Large Plaster-of-Paris Model of Imperial Rome*, Videometrics VIII, edited by J.-Angelo Beraldin, Sabry F. El-Hakim, Armin Gruen, James S. Walton, 18-20 January 2005, San Jose, California, USA, SPIE, vol. 5665, 119-133.

[5] Scale model of Rome: www.romereborn.virginia.edu (accessed april 2011)

[6] Madeleine S. and Fleury P.: *Reconstruction of Ancient Rome in Interactive Virtual Reality*, Proceedings of DMACH conference, 2010, Amman, Jordania, March 13-15, 169-182.

[7] City Engine: http://www.procedural.com (accessed april 2011).

[8] Dylla K., Müller P, Ulmer A., Haegler S. and Frischer B. :*Rome Reborn 2.0: A Framework for Virtual City Reconstruction Using Procedural Modeling Techniques*. Proceedings of Computer Applications and Quantitative Methods in Archaeology (CAA), 2009.

[9] Moloney, J., & Harvey, L.:*Visualization and Auralization of Architectural Design in a Game Engine Based Collaborative Virtual Environment*. IEEE 8th International Conference on Information Visualisation, London, 2004, IEEE Computer Society Press, pp. 827-832.

[10] Varano S., Truchot T., Bignon J-C.: *Ludic and didactic paths in a cultural heritage building : prototype of a learning system*, eCAADe 2009, 16-19 September, Istanbul, Turkey

[11] Chevrier C., Jacquot K. and Perrin J. P.: *3D modeling of a town scale model*, proceedings of the EuroMed Conference, Limassol, Cyprus, 8-13 Nov 2010, pp. 99-107.

[12] Chevrier C., Jacquot K. and Perrin J. P.: Modeling specificities of a physical town scale model, Proceedings of DMACH conference, 2010, Amman, Jordania, March 13-15, 103-117.

[13] Unity software: http://unity3d.com

[14] Creaform – Handy Scan: http://www.creaform3d.com

[15] Scale model Web site: http://www.museedesplansreliefs.culture.fr/