Tridimensional Scenes Management and Optimization for Virtual Reality simulators

Juan P. D'Amato[1,2], Cristian García Bauza[1,2], Marcos Lazo[1,2], Virginia Cifuentes[1,3],

[1]Instituto PLADEMA, Universidad Nacional del Centro de la Provincia de Buenos Aires, Campus Universitario S/N, 7000 Tandil, Argentina
[2]Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina
[3]Comisión Nacional de Investigaciones Científicas de la Provincia de Buenos Aires (CICPBA), Argentina

{jpdamato, crgarcia, mlazo, cifuente}@exa.unicen.edu.ar

Abstract. Virtual Reality (VR) simulation offers a new paradigm for realistic control and operation training, employing 3D digital objects and environments to create immersive experiences. The virtual scenarios should be as similar to the real ones as possible, to improve training. Furthermore, obtaining a digital version of these scenarios is a complex process that involves several tasks including mathematical modeling and 3D detailed geometry generation, among others. This paper presents some novel software tools that easier the development of a 3D virtual subterranean simulator. By providing tight integration between traditional CAD software and visualization engines, these tools are successfully applied to the construction of several virtual scenarios. Moreover, some strategies to optimize digital resources to allow loading and visualizing large maneuver exercises along realistic train trajectories in desktop computers are presented.

Keywords: Graphics Computing, Geometry, Computational Optimization, Resource Administration, Training Simulators.

1 Introduction

Virtual Reality (VR) environments along with simulation techniques are valuable tools to practice and learn and can be applied to different disciplines and trainees. Being a trustworthy human-machine interface, these tools reduce the gap between the operation model and the real environment. The advantages of immersive based simulators for operators training are: the visual stimulation produced by the recreation of situations of the real world in a fully interactive fashion, the opportunity to develop knowledge and to practice skills and interact with other students protecting them from unnecessary risks. Many works show the advantages of VR for education [1, 3, 5, 6, 10, 11].

The training experience requires realistic and interactive operational environment visualization. For this purpose, high performance personal computers, and joint work

Á. Rocha et al. (eds.), *New Advances in Information Systems and Technologies*,
Advances in Intelligent Systems and Computing 444,
DOI 10.1007/978-3-319-31232-3_23

of different disciplinary groups are necessary to model the problem in a computational way [4]. A fundamental part of this team is the graphic design work. The designers model the scenarios with tridimensional geometry elements coming from CADs, blueprints, elevation maps or others. Generating these scenarios generally takes a long time. Therefore, they need various tools to ease the geometry organization [14].

In particular, a subway simulator is being developed by a group of designers. The scenario, composed by the stations and the tunnels, is represented by tridimensional models which contain hundreds of thousands of polygons and hundreds of high quality textures. Additionally, a complete scenario also includes animated objects (such as people, vehicles, among others) to get a realistic rendering. These scenarios consume plenty of memory that should be optimized in order to guarantee that an operator could work in real time without compromising the visual quality. Finally, the optimized scenarios are visualized in a rendering engine like Ogre3D [2].

In this paper, some techniques that reduce hardware requirements are presented. Particularly, it is described an image and geometry analysis algorithm that eliminates object redundancies and reduces images size. Furthermore, a dynamic scenarios load algorithm specially adapted to use during a subway driving training process is described. These tools were successfully applied in the creation of several virtual environments related to Argentinean train companies.

This article is organized in the following way: the materials and methods are described, then the result of the strategy proposed is presented and, finally, the conclusions.

2 Materials and Methods

The main purpose of working with Virtual Reality is to present to operators, who are being trained, multiple real world operational situations. Under this paradigm, trainees can face operators with abnormal functioning or risky situations, and evaluate how they react, as presented in [13, 15].

The VR method is only effective if the whole situation is greatly recreated, using both hardware controls and computational methods. The sensation of immersion is only achieved recreating in a digital and precise way a normal operational environment. It is well established that the coherence of a combination of simultaneous stimulus and the immersion sensation are key factors for training [7, 12].

The realistic 3D scenarios can be modeled using design software, photo sets, and detailed views, among others; generating sets of meshes and textures. When the 3D scenarios are very large, for example when modeling 10 kilometers tunnels, the designer's team may find it difficult to handle them and, thus, they need extra customized tools.

Following, we describe the methodology followed by our group for constructing a digital scene for VR using an automatic mesh building tool. The case of subway trains simulators is taken as an example.

2.1 Building 3D Scenes

The operator must recognize the environment to achieve an efficient and secure manipulation of the machinery. For this reason, the representation must consider all the particularities and elements that help subway drivers.

To create scenarios, first, in situ images are taken to capture the surrounding scenery elements. Then, exact positions are registered where the relevant objects are placed.The stations scenarios are completely different from each other, so they must be created by designers, paying special attention to indicators and representative paintings. As it can be observed in Figure 1, each station is different.

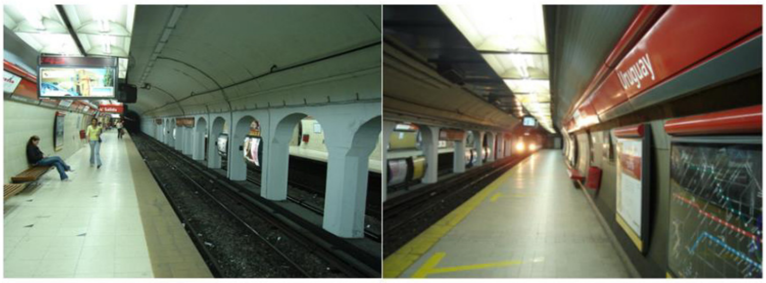

Fig.1. Two photographs of subway stations in Buenos Aires, Argentina.

On the other hand, tunnels have a repeated shape; consequently, they could be built automatically. Such geometry is constructed from real CADs that have altimetry, curves, tunnels shapes and other relevant information. Such data are filtered and validated, and then, using geometry extruding techniques, a first 3D model is generated. This model is stored in a standard 3D format that is shared between the designer and the programming group.

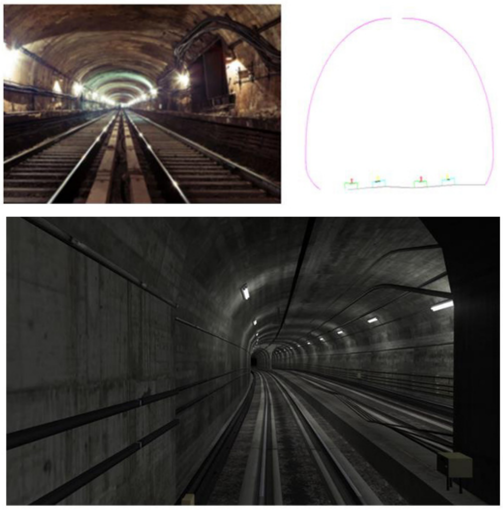

Fig.2.A real tunnel photograph (up-left), the CAD tunnel profile used (up-right) and a render of the extruded profile (down).

Finally, designers take these models, combine them with the stations, and add some relevant details, as brake indicators, lights, and other visual features. Then, using computer graphics techniques, illumination is customized and applied (see Figure 2). Figure 3 shows a final result of the generation, rendered in a commercial tool.

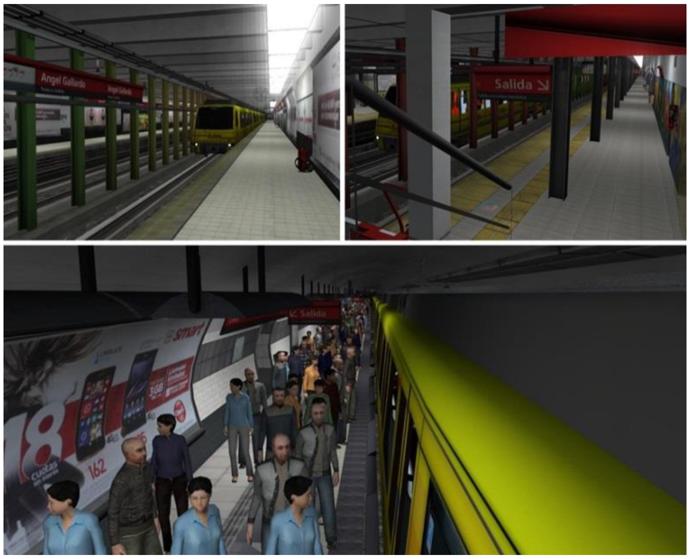

Fig.3. Different renders of a virtual station.

3 Method for Scenes Optimization

The 3D scenarios should be shown in real time, as far as the operators move along a virtual railway. As the scenarios are generally extensive, the resources (geometry and images) should be prepared and optimized carefully, not to affect the final 3D visualization.

To optimize the overall scenarios, three approaches are considered in this work: texture optimization, replicative mesh geometry elimination and dynamic loading. The first two strategies are applied before the simulator starts. The third one, as its name states, is applied in real time.

3.1 Optimization of Textures Sizes

The textures describe the surface of tridimensional objects. They are also used to apply visual effects as shadows or rugosity. The textures come from a material catalog or photographs assigned by designers to 3D objects. A typical scenario has thousands of images.

In general, designers use standard image resolutions for each image; however, in practice, countless images are either homogeneous (with few colors or repetitive patterns) or have a high detail that could not be appreciated when they are applied to little objects. Therefore, an ideal texture size is defined by the analysis of two features, homogeneity and 3D size.

The entropy [16] is considered the measure of texture homogeneity. It is the average quantity of information that an image contains. When all the pixels are equally probable (plain probability distribution), the entropy is maximum. When entropy is lower, information is higher. Therefore, the entropy of an image$X$ is denoted by $H(X)$.

Regarding the resolution, little objects, such as a drop tin or a small label could have a highly detailed texture which is not appreciated during simulation. It is possible to measure how much 3D space a texture occupies considering the 3D object dimension that utilizes them. We called M to each scenario object being $T(X)$ the normalized texture size in 3D space.

To combine both indicators, a linear combination was applied:

$$P(X) = \alpha H(X) T(M0) \quad \textbf{(1)}$$

Being α a normalization factor, $T(M_0)$ classifies objects sizes in three categories {1, 2, 3} regarding the surface S of the texture calculated in m^2, being:

1. if $S < 1\, m^2$
2. if $1\, m^2 \leq S < 5\, m^2$
3. in other case

According to $P(X)$, the texture is dimensioned in a smaller scale, preserving a minimum pixels resolution and maintaining the original aspect ratio.

3.2 Repetitive Objects Elimination

The designers usually reuse many objects when they create a complex scene: lights, signs, furniture, doors, among many others. When working with several scenarios, they clone objects and move them from one position to another, renaming them, increasing the overall memory size. To easier designer's task, it is desirable to automatically identify objects which are geometrically identical (clones) and eliminate them.

A metric to detect clones is proposed. Two objects are similar when they have the same material (same colors and textures) being the amount of the elements (vertex and triangles) and their geometrical arrangement is equal. Therefore, the classification criterion is reduced to the calculation of the minimum distance between the vertexes of two meshes M_X and M_Y as present in [9]. The formulation to calculate it is:

$$D(M_X, M_Y) = \sum_{x \in X} d(x, M_Y) + \sum_{y \in Y} d(y, M_X) \qquad (2)$$

where x is a vertex of the mesh M_X, y is a vertex of mesh M_Y, $d(x, M_Y)$ is the minimum distance from x to M_Y, and $d(y, M_X)$ is the minimum distance from y to M_X.

This process is applied to each pair of objects that have the same amount of vertexes. If $D(M_X, M_Y)$ is close to 0, it is considered that the objects are clones and a unique geometry instance is kept.

3.3 Dynamic Scene Loading Method

The train scenarios have an important difference in reference to the flight or driving simulators. While those simulators demand extensive open fields where the vehicles move in any direction, the movement of the train is restricted to the railway, like moving along a line. Therefore, it is possible to know in advance the next position of the observer inside the train and thus, organize the scene to be rendered.

Particularly, a subway line is defined as a section sequence. Each section can be a “tunnel” or a “station” having an associated 3D representation. As the entire scenario could not be kept in memory, the scenes are loaded in a dynamic way, considering only the coming section.

To know how many sections must be visualized, the observer’s movement and the distance to the coming section are considered. For each update, the remaining time to the next scenes is computed based on the current speed. If it is smaller than a threshold (15 seconds), the corresponding scene starts to be loaded. In order to keep the refresh rate, the load is concurrent to the simulation, and it does not block the visualization. Given that the scene is composed by hundreds of objects and textures, and that it is a process that requires the use of a graphic card, a limited quantity of objects is created for each frame. After a frame is rendered, the objects are pumped to the GPU memory, assuring that the time is smaller than 10 milliseconds. Simultaneously, objects and materials that are no longer visible or needed are removed, and memory is freed.

Figure 4 shows the uploading and downloading scenes mechanics for two instants of execution time. This way, the memory and resources consumed by the visualizer are maintained almost constant, allowing subway representation of any length.

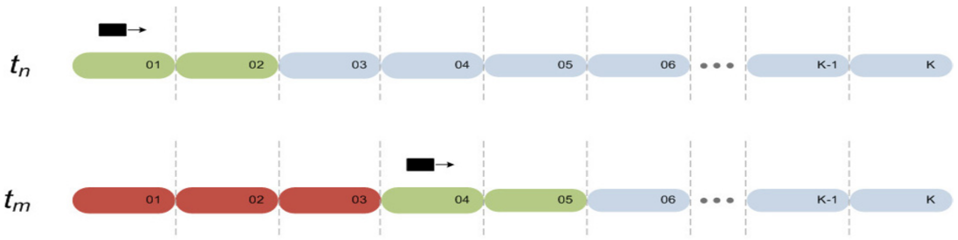

Fig.4. Uploading and downloading scenes mechanism in two time instants t_n and t_m: loaded scenes (green), unloaded scenes (light blue) and eliminated scenes (red).

4 Results

In this section, we present the results obtained by the application of the proposed methods. For this purpose, we used a scenario where a train goes through 3 stations over a 2 kilometers tour, at a constant speed of 50 kilometers per hour. This scenario is composed of 630 3D meshes and about 300 image textures; however, in some cases, the scenarios could be bigger than this.

For each method, we carried out a different experiment, as they are described below. For the tests, we employed an Intel i7 CPU, with 12 GB of RAM and a NVIDIA 980 GTX GPU with 3GB of RAM.

4.1 Texture Size Change Analysis

The GPU memory space required for the whole textures reaches 1850 MB, without considering the auxiliary textures created by the graphic engine.

Employing the texture analysis method, we detect four possible texture combinations:

- textures with a repetitive pattern (low entropy) mapped to small or large objects
- textures with a non-repetitive pattern (high entropy) mapped to small or large objects

Empirically, it was observed that low entropy textures could be reduced more without affecting visual perception. In the other case, the object size determines the optimization rate. Table 1 shows these cases with the estimated reduction percentage.

Table 1. Examples of textures used in the simulator.

Texture 1	Texture 2	Texture 3
High Entropy **Little Size** 50% reduction	**LowEntropy** **Intermediate/High Size** 25% reduction	**High Entropy** **Intermediate/High Size** 75 % reduction

After evaluating the whole textures dataset, we found that about 80% of textures could be reduced to half of their original size. In this case, the GPU' memory requirement falls below 654 MB.

In Figure 5 it can be appreciated how image quality is affected. For simulation purpose, the result is more than acceptable, thought.

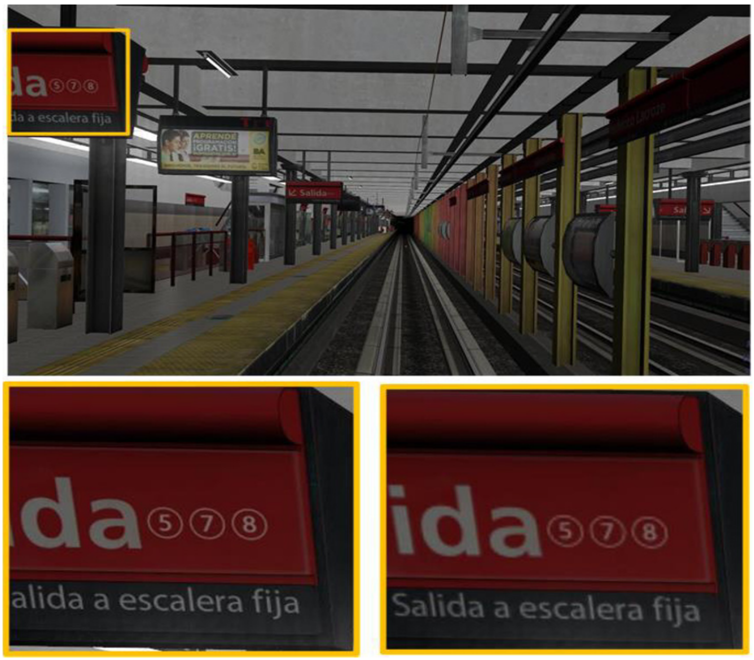

Fig.5. Scene captured **(**above**)**.Original image (below, left) and 50% reduced (below, right).

4.2 Repetitive Object Elimination Analysis

In this experiment, we registered the amount of memory necessary to store vertex and triangles information of the whole scene. After applying the object duplication detection method, we found the amount of replicated objects that fulfill the condition described in ec.2. Table 2 resumes the quantity of objects discovered for each scene and the memory that can be free.

Table 2. Duplicated object and unnecessary memory proportion visualization.

	#Objects	**#Repeated Objects**	**Memory use (%) Repeatedobjects**
Scene 1	161	59(36%)	17%
Scene 2	154	75 (48%)	8%
Scene 3	315	207 (65%)	30%

We found that between 36% and 65% of the objects were redundant. In general, these were low-poly objects (like boxes or artifacts), As a result, the amount of memory that can be freed is not so significant. In the case of more complex geometry objects, for example, an escalator or a platform, we found only one instance.

4.3 Dynamic Scene Loading Evaluation

Finally, we evaluate the dynamic scene loader. To carry out this experiment, we evaluated the method while the simulator is running, using the already optimized scenarios. In this scenario, the train moved from the first station to the final one, at a constant speed. We explicitly load objects and remove them when they are no longer needed.

We compared both cases; one when the whole scenario is loaded at the beginning and then, loading each part while the train is moving forward. In the first case, Figure 6 shows that the simulation needs about 2 GB of RAM. Once it starts, the memory allocation keeps constant. The first part (named common) corresponds to graphics engine textures and objects. Then, the dynamic loader is evaluated. As it was expected, the simulator requires less memory, about 1.2 GB of RAM. Even more, as loading part of scene is faster, the simulator is ready to run before the other case.

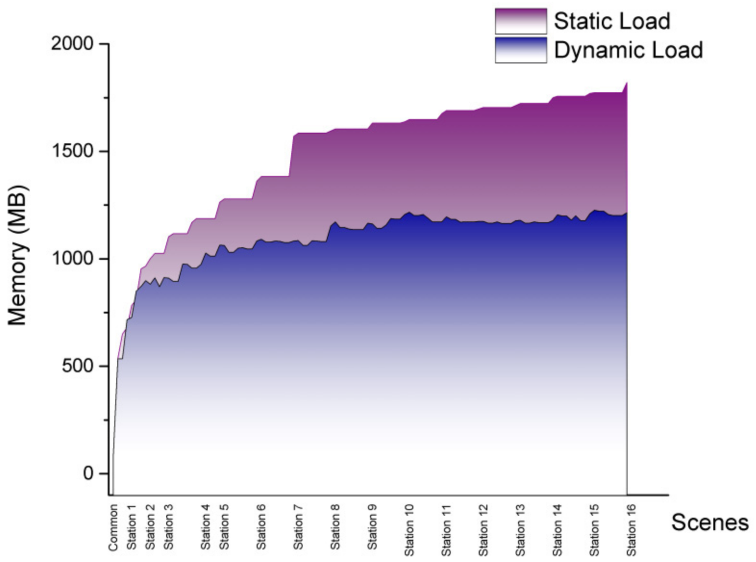

Fig.6. Resources used for the simulation initiation along 16 stations.

5 Conclusions

In this paper, some generic techniques to optimize large immersive simulator scenarios which achieve a high frame-rate with less hardware requirements are presented.

The texture optimization and the replicative mesh geometry elimination techniques are efficient and simple to implement, and they really easier the design team task. Both could be applied to optimize any tridimensional scenario.

On the other hand, the dynamic loading technique is more specific, because it depends on how the spectator/camera moves inside the scenario. Moreover, this technique relies on the graphics engine implementation and its performance could vary according to how the engine manages media resources.

In future versions, we will include some other techniques, like dynamic geometry and texture generation in order to reduce the memory requirement even more.

References

1. MartínezDurá R., ArevalilloHerráez M., GarcíaFernández I., GamónGiménez M., Rodríguez Cerro A.: Serious Games for Health and Safety Training. Serious Games and Edutainment Applications, Chapter 7, 107-124, Springer-Verlag London Limited, doi:10.1007/978-1-4471-2161- 9_7 (2011)
2. Ogre Open Source 3D Graphics Engine. http://www.ogre3d.org/ (2015)
3. Vénere, M., Cifuentes, M. V., D'Amato, J., y García Bauza, C.: Editor de escenarios para aplicaciones de Realidad Virtual. 34º Jornadas Argentinas de Informática e Investigación Operativa, ISSN 1666-1095 (2005)
4. Ferrington, G., Loge, K.: Virtual reality: A new learningenvironment. Comput. Teach. 19 (7), 16–19 (1992)
5. Li, G.C., Ding, L.Y., Wang, J.T.: Construction project control in virtual reality: A case study. J. Applied Sci. 6, 2724-2732 (2006)
6. Boroni, G., Garcia Bauza, C., D'Amato, J., Lazo, M.: Siper-Virtual Reality Simulator of Periscope. World Applied Sciences Journal, 813-817 (2012)
7. Sims, E.M.: Reusable, lifelike virtual humans for mentoring and role-playing, Comput. Educ. 49 (1), 75–92 (2007)
8. 3D Max Studio Modelling. http://www.autodesk.es/products/3ds-max/overview (2015)
9. Cignoni, P., Rocchini, C., Metro, R.: Measuring error on simplified surfaces, Computer Graphics Forum 17 (2), 167–174 (1998)
10. Schwebel, D., Combs, T., Rodriguez, D., Severson, J., Sisiopiku, V.: Community-based pedestrian safety training in virtual reality: A pragmatic trial. Accident Analysis & Prevention 86, 9-15, January 2016 (2016)
11. Grabowski, A., Jankowski, J.: Virtual Reality-based pilot training for underground coal miners. Safety Science 72, 310-314, February (2015)
12. Borsci, S., Lawson, G., Broome, S. Empirical evidence, evaluation criteria and challenges for the effectiveness of virtual and mixed reality tools for training operators of car service maintenance. Computers in Industry 67, 17-26, February (2015)
13. Takeda, J., Kikuchi, I., Kono, A., Ozaki, R., Kumakiri, J., Takeda, S. Efficacy of short-term training for acquisition of basic laparoscopic skills. Gynecology and Minimally Invasive Therapy, in press, available online 26 June (2015)
14. Mattioli, L., Cardoso, A., Lamounier, E.A., do Prado, P.: Semi-automatic Generation of Virtual Reality Environments for Electric Power Substations. Springer International Publishing Switzerland, Rocha et al (eds.). Advances in Intelligent Systems and Computing 353, 833-842, doi: 10.1007/978-3-319-16486-1_83 (2015)
15. Novak-Marcincin, J., Janak, M. Barna, J., Novakova-Marcincinova, L.: Application of Virtual and Augmented Reality Technology in Education of Manufacturing Engineers. Springer International Publishing Switzerland, Rocha et al (eds.). Advances in Intelligent Systems and Computing 276, 439-446, doi: 10.1007/978-3-319-05948-8_42 (2014)
16. Balian, R.: Entropy, a Protean concept. In Dalibard, Jean.Poincaré Seminar 2003: Bose-Einstein condensation - entropy. Basel: Birkhäuse, 119–144, ISBN 9783764371166 (2004).