A Hybrid Rendering Technique to Navigate in Large Terrains Using Mobile Devices

José M. Noguera, Rafael J. Segura, Carlos J. Ogáyar
University of Jaén. Departamento de Informática.
Campus Las Lagunillas s/n.
23071 Jaén, Spain
Email:{jnoguera, rsegura, cogayar}@ujaen.es

Robert Joan-Arinyo
Universitat Politècnica de Catalunya.
Escola Tècnica Superior d'Enginyeria Industrial.
Diagonal 647, 8. 08028 Barcelona, Spain
Email:robert@lsi.upc.edu

***Abstract*—We describe a hybrid client-server technique for remote adaptive streaming and rendering of large terrains in resource-limited mobile devices. The technique has been designed to achieve an interactive rendering performance on a mobile device connected to a low-bandwidth wireless network. The rendering workload is split between the client and the server. The terrain area close to the viewer is rendered in real-time by the client. The terrain located far from the viewer is portrayed as view-dependent impostors, rendered by the server on demand. A prototype has been built and an exhaustive set of experiments covering several platforms, wireless networks and a wide range of viewer velocities has been conducted. Results show that the approach is feasible, effective and robust.**

***Index Terms*—Terrain navigation, Terrain rendering, Mobile computing, Adaptive streaming.**

I. Introduction

Nowadays, mobile devices like mobile phones and Personal Digital Assistants (PDA) are used as interactive guides to real environments, and offer features such as Global Positioning Systems (GPS), visualization of maps and terrain rendering. In personal navigation and Geographic Information Systems (GIS), terrain rendering plays an outstanding role. Here, real time terrain rendering is a must.

Many visualization and streaming techniques have been developed for desktop computers and workstations. However, there are several technical limitations that preclude using them in mobile devices. They lack of the CPU power required to properly handle complex data structures and the huge amount of data involved in terrain representation. Besides, mobile devices in general do not include 3D graphics hardware thus GPU-based solutions cannot be considered yet. Moreover, the limited memory featured along with slow wireless commercial networks pose great difficulties to process large amount of data on mobile devices.

In this paper we describe a technique for wireless remote streaming and rendering of terrains on mobile devices using low bandwidth networks. We propose a client-server hybrid rendering approach. The client renders the geometry of the terrain close to the viewer, whereas the distant terrain is portrayed as impostors, rendered by the server and streamed to the client. Since impostors represent terrain distant from the viewer, they do not need to be updated unless the user's position in the virtual environment changes beyond a given threshold. The approach provides tools to dynamically split rendering task between the server and the client according to the resources available in the client and the network congestion.

As a proof of concept, we have implemented a prototype and an exhaustive set of experiments has been conducted. It achieves terrain visualization at interactive rates on mobile devices. Results show that the approach is feasible, effective and robust.

II. Previous Work

According to where the geometry-rendering task is performed, current networked rendering methods can be divided into two categories: server-based techniques and client-based techniques.

In the first category, a dedicated remote rendering server is in charge of performing the geometry rendering task and streaming images to a client over a network. To achieve an interactive rate, these techniques stream a massive volume of images to the client, what can easily result in a congested network. Also, rendering capabilities of modern mobile devices are wasted. In the second category, 3D rendering tasks are delegated to the client. This approach reduces the streaming load. However, the client must provide the computational power required to render good quality and to manage a depth enough viewing distance.

Many server-based techniques have been reported in the literature. For example, Lamberti et al [1] and Jeong and Kaufman [2], offer solutions of server-based rendering techniques of generic scenes on mobile devices. Their approaches are based on the use of image and video compression algorithms. Jeong and Kaufman claim to achieve a speed of five frames per second (FPS) using a 802.11b radio interface on a PDA. Wen et al. [3] propose a similar server-based rendering technique targeting terrain navigation. However, measures of the client performance and network usage are not reported.

Boukerche et al. [4], [5] and Pazzi et al. [6] have presented alternative methods for server-based approaches by using scheduling mechanisms and partial streaming of images. However, these approaches severely limit how the viewer can move and do not perform well in dynamic scenes.

Most networked rendering techniques dealing with large terrains found in the literature fall in the client-based category.

A great amount of work has been done on the subject of large terrain rendering, and a comprehensive overview of this subject is beyond the scope of this paper. For a detailed survey about networked multi-resolution terrain rendering methods, we refer the reader to [7] and [8].

As far as we know, solutions in the literature concerning mobile devices do not consider the two features that characterizes them: small bandwidth wireless networks and limited computing capabilities. Our approach combines both rendering techniques, server-based and client-based, allowing a better use of the available computational resources and network bandwidth, which is crucial to wireless streaming applications on thin devices.

III. The Hybrid Approach

In this section we describe our approach to develop wireless remote streaming and rendering of terrains on mobile phones making use of low bandwidth networks available in the market.

Our approach is basically a hybrid rendering technique where the rendering task is split between a remote server, generally featuring high-end hardware and software resources, and a mobile client, usually with very limited resources.

The main tasks in charge of the server are: 1) To store the terrain, 2) To supply the client with small chunks of terrain close to the user's position to be displayed and 3) Rendering and sending to the client impostors for the terrain that is far from the users current position.

The main client tasks are 1) Rendering the terrain close to the user's position according to the required level of detail, 2) Displaying the impostor that replaces the actual terrain in the background and 3) Requesting from the server updates for both the terrain and the impostor to be displayed as the users changes its position. The client dynamically adjusts its requests for updating terrain and impostors according to its own capacity and the network congestion.

To summarize, our approach offers the following advantages: 1) The terrain area to be rendered by the client can be small without reducing the depth of view. 2) Any terrain rendering technique can be used by the server to generate the impostors, including those based on GPU and, 3) Since screen resolution of mobile devices is small, usually in ranges like 320×240 and 640×480, impostors do not need to be generated at high resolution, thus saving bandwidth.

IV. Panoramas

In general, an impostor is a two-dimensional image that is used instead of a true three dimensional model to improve the rendering performance.

In our approach, impostors consist of two-dimensional synthetic images that simulate a wide view of a physical terrain placed in the background far from the viewer. These impostors are called *panoramas*, [4]. A panorama captures a visible view from a point in space in all directions. To visualize a panorama, it is first projected on the inner six faces of a cube.

Panoramas projected on a cube are usually referred to as *skybox* [9]. The construction of a skybox panorama is straightforward [4]. Each face of the cube covers 90 degrees of view both horizontally and vertically. The panorama is built by the server by placing its camera in the viewer coordinates in the client and making use of the terrain nearby. Then 6 orthogonal images are rendered, compressed using any standard image compression algorithm, such as JPEG, and they are sent to the client, which mappes them onto each of the six faces of the cube.

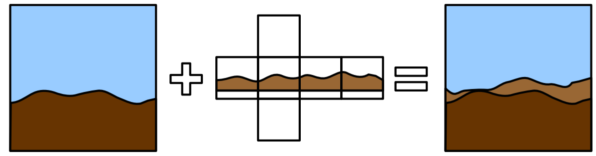
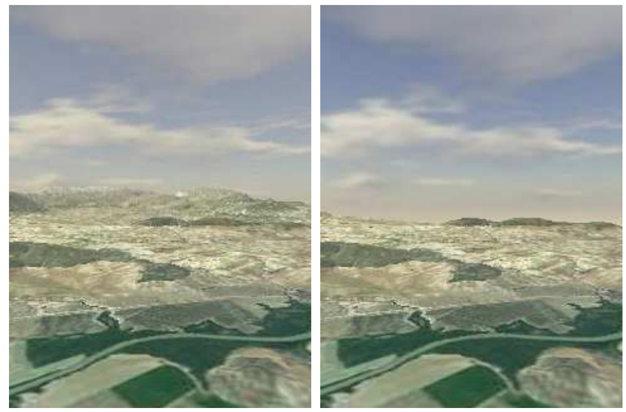


Fig. 1. Synthesis of the nearby terrain rendered in real time by the client and the distant terrain rendered by the server on demand.



(a) (b)

Fig. 2. Scenes rendered in a Nokia N95. a) Picture with panorama. b) Picture without panorama.

The resulting image is composed by the client by merging the terrain and the skybox panorama as illustrated in Figure 1. Figure 2 shows one example of terrain rendered with and without panorama.

In our approach, the server renders the terrain in the background as a panorama and the client renders the terrain close to the viewer at a given level of detail. However, terrain shown to the user in the screen must be seamless rendered. Therefore, we split the terrain into nearby terrain and panorama as follows. Let the view volume in the client be limited by the front and back clipping planes placed respectively at $zfront_c$ and $zback_c$ distance from the viewing point. Similarly, let the view volume in the server be limited by the clipping planes placed at distances $zfront_s$ and $zback_s$. Then, we require that $zfront_s = zback_c$, that is, the front and back culling planes in the server and client respectively are coincident. See Figure 3.

Clearly, the client renders terrain close to the viewer whereas the distant terrain is culled. On the contrary, the server culls the nearby part of the scene, and only the distant part is taken into account to render the panorama.

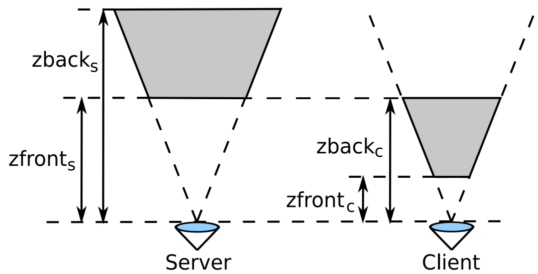


Fig. 3. Splitting the view volume as terrain to be rendered by the client and panorama.

V. Results

To evaluate and validate our approach, we carried out an extensive set of experiments. Exhaustive results are collected in [8].

We used the Puget Sound dataset. It is made of 16385×16385 samples at 10m spacing, 2 bytes height values with a vertical resolution of 0.1m. The texture map has 16385×16385 pixels, with a resolution of 10m per pixel.

The terrain was represented following two different levels of geometric complexity: low and high quality. These two levels limit the number of triangles used to portray the terrain rendered by the client.

The multi-resolution technique used by the client to adaptively stream and render the nearby terrain is detailed in [8].

For each test, we performed a flyover following a rectilinear path along the diagonal of the terrain. To avoid fake results, the terrain boundaries were never reached. The viewer always moves forward at constant speed of 150 km/h and at a constant height of 100m over the terrain. To allow the data to be partially fetched, we introduced a 15 seconds delay before the viewer started the flyover.

To study how our technique performs in real-world wireless connections, we carried out our tests using a popular mobile telecommunications technology: Universal Mobile Telecommunications System (UMTS), usually known as 3G.

All tests have been run with and without panoramas, that is, using both the proposed hybrid rendering technique and a pure client-based rendering technique. In both cases, the minimum viewing distance was 30Km. When using panoramas, they were placed 7.5 Km away from the viewer. The maximum allowed error was 5% and the resolution was 256×256 pixels per skybox face. Tests were conducted on a Nokia N95 with 332 MHz ARM11 CPU, 128 MB of RAM and a PowerVR MBX GPU which supports OpenGL ES 1.1.

Table I summarizes the results obtained when using our hybrid technique and a pure client-based rendering technique. For each quality level, the table includes values for the number of frames per second rendered, the average number of triangles rendered by the client in each frame and the total downloaded data measured in kB in each 300 seconds flyover. All values are averaged.

Throughout this discussion, streamed data included geometry, textures and panoramas. Our approach, in general, manages to achieve an almost constant frame rate and number of triangles rendered during most of the flyovers, see [8].

TABLE I
Averaged results yielded by the Nokia N95 during the 300 seconds flyovers at 150 km/h using an UMTS network

Rendering	Quality	Framerate	Triangles	Download
Hybrid	High	10,67	41244	1511
	Low	49,58	5194	409
Client	High	7,11	64310	1805
	Low	36,10	8661	396

Considering separately values in Table I, we see that, as expected, high quality rendering is consistently more demanding than low quality rendering.

Comparing corresponding values in Table I, we see that the average number of triangles rendered by the client in the hybrid technique is smaller than the number required by the server-based approach, with reductions ranging from 35% to 40%. Consequently, the number of frames per second delivered by the hybrid technique is always higher than that delivered by the pure client-based technique. The increment of the frame rate ranges from 30% to 45%. This results in a continuity in the display smoother for the hybrid than for the client-based approach.

Concerning the network bandwidth required, the total amount of data downloaded in high quality tests with the hybrid technique is always smaller than in the client-server approach. Thus, in this case, the use of panoramas generated on-demand by a remote server does not increase the network usage. As a matter of fact, sending panoramas takes less network resources that sending the actual geometry in high quality tests. For low quality tests, however, the small amount of triangles transferred results in a smaller traffic in the network, similar in both rendering techniques.

The hybrid rendering technique allows our streaming architecture to increase the quality and the view distance of the scene without increasing the system requirements.

VI. Summary and Future Work

In this paper, we have presented a client-server hybrid rendering technique to navigate large terrains using commodity mobile devices with limites resources. The approach combines a fast and lightweight multi-hierarchical terrain representation with a 2D panoramic impostor generated on demand by a remote server. This approach allows progressively transferring complex scenes while keeping high visual quality. We have implemented a prototype and carried out a comprehensive set of tests that proved its effectiveness.

We plan to develop future work following several lines. First, as our architecture supports multiple clients connecting concurrently to the same server, we plan to perform exhaustive tests using a varying number of concurrent clients. Second, we want to study how the distance $zfront_c$ of the back clipping plane influences the performance and the network usage.

Acknowledgements

This work has been partially granted by the Junta de Andalucía, the Spanish Ministry of Education and Science, and the European Union ERDF funds under research projects P06-TIC-01403, P07-TIC-02773, TIN2007-67474-C03-01 and TIN2007-67474-C03-02.

References

[1] F. Lamberti and A. Sanna, "A solution for displaying medical data models on mobile devices," in *SEPADS'05: Proceedings of the 4th WSEAS International Conference on Software Engineering, Parallel & Distributed Systems*. Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2005, pp. 1–7.

[2] S. Jeong and A. E. Kaufman, "Interactive wireless virtual colonoscopy," *The Visual Computer*, vol. 23, no. 8, pp. 545–557, 2007.

[3] J. Wen, Y. Wu, and F. Wang, "An approach for navigation in 3D models on mobile devices," in *CMRT09: City Models, Roads and Traffic*, Paris, France, 2009, pp. 109–114.

[4] A. Boukerche, R. Jarrar, and R. W. Pazzi, "An efficient protocol for remote virtual environment exploration on wireless mobile devices," in *WMuNeP '08: Proceedings of the 4th ACM workshop on Wireless multimedia networking and performance modeling*. New York, USA: ACM, 2008, pp. 45–52.

[5] A. Boukerche, R. W. Pazzi, and J. Feng, "An end-to-end virtual environment streaming technique for thin mobile devices over heterogeneous networks," *Computer Communications*, vol. 31, no. 11, pp. 2716–2725, 2008.

[6] R. Pazzi, A. Boukerche, and T. Huang, "Implementation, measurement, and analysis of an image-based virtual environment streaming protocol for wireless mobile devices," *Instrumentation and Measurement, IEEE Transactions on*, vol. 57, no. 9, pp. 1894–1907, 2008.

[7] R. Pajarola and E. Gobbetti, "Survey of semi-regular multiresolution models for interactive terrain rendering," *The Visual Computer*, vol. 23, no. 8, pp. 583–605, 2007.

[8] J. Noguera, R. Segura, C. Ogáyar, and R. Joan-Arinyo, "A hybrid client-server based technique for navigation in large terrains using mobile devices," http://www.lsi.upc.edu/dept/techreps/llistat_detallat.php?id=1077, Dept. Llenguatges i Sistemes Informátics, Universitat Politècnica de Catalunya, Tech. Rep. LSI-10-3-R, 2010.

[9] J. Shankel, *Game Programming Gems 2*. Rockland, MA, USA: Charles River Media, Inc., 2001.