An infrastructure for remote virtual exploration on PDAs

Rosario De Chiara, Ugo Erra, Andrea Petta, Vittorio Scarano, Luigi Serra
ISISLab
Dipartimento di Informatica ed Applicazioni "R.M. Capocelli"
Università degli Studi di Salerno
{dechiara, ugoerr, vitsca}@dia.unisa.it,
{andrpet, luigser}@gmail.com

Abstract

In this paper we present a prototyped system to enable the virtual exploration of a complex virtual environment. Our approach exploits Quest3D as main rendering engine, its output is conveyed toward users PDAs to allow them to explore using the PDA as a (mobile) interface to the virtual environment. An important aspect of the system is that it relies on an off-the-shelf PC and low end wireless network. Some early results showed that the prototype is able to easily manage 5 PDAs. Suggested fields of use of our system are virtual cultural heritage, educational virtual environments, videogames.

Keywords— **Computer Graphics, Rendering engine, Remote visualization**

1 Introduction

Personal Digital Assistants (PDAs) availability at cheapest price is growing steadily in the last years. In particular, PDAs with integrated WLAN (wireless LAN) and cell-phone capabilities accounted a significant part of this growth. As a matter of fact, the PDAs are playing a significant role in our daily life, given their growing capabilities, interoperability with desktops/laptops, smarter applications and increased coverage of Wireless Local Area Network in SOHO (Small-Office, Home-Office) as well as in public places (airports, theaters, restaurants, malls, etc.).

Research in the field of visualization has been exploring the field of remote visualization for a while, now. In fact, when applied to virtual reality, recent interest has provided the study of *remote virtual exploration*, i.e. the ways of enabling one or more users to explore a complex virtual environment without being physically connected to the place where this complex environments is kept and manipulated.

Different approaches have been proposed to allow this kind of exploration. Architectures where heavy-duty servers/workstations/cluster are employed for the remote rendering are studied in the area (some results are provided in section 2) while some results point to the application of remote visualization in the field of protection of (sensitive) 3D models when rendering on remote devices, or to hybrid architectures where the (limited) capabilities of 2D rendering of the remote devices are used, after an initial (3D to 2D) preprocessing performed server-side.

Our paper is meant to explore the capabilities of widely available and cheap hardware to allow remote visualization. Our prototype uses an off-the-shelf PC and a low bandwidth (IEEE 802.11b) wireless network to implement a system of remote exploration with Windows Mobile PDAs (Personal Digital Assistant). Our preliminary results show that no significant reduction in performances is obtained with upto 5 mobile devices.

1.1 Remote exploration

Remote exploration systems allows different typology of explorations to remote users, the intent is to let users to exploit computational resources from remote, and using such resources to render complex scene that would not "fit", in most general sense, on the user hardware. An instance of this approach is, for example, the X Consortium system, whose architecture enabled users to remotely exploit a graphics system using an X Server [6].

A commercial solution available from Silicon Graphics is SGIVizserver [3] that implements a so-called "*Visual Area Network*" (VAN) in which the interactive graphics generated by powerful visual servers distributed to remote, thin clients. VAN enables remote users to manipulate and visualize large data sets. SGIVizserver is application independent and using it does not require any particular programming style, its use is transparent, every application that correctly use OpenGL can be made aware of SGIVizserver infrastructure, and the output of every call to OpenGL API is redirected to remote devices.

2 Previous and related work

In this section, we present a brief overview about remote virtual exploration solutions.

One of the most popular remote virtual exploration solutions is QuickTime VR [5]. QuickTime VR is a technology pioneered by Apple that allows a series of panoramic 2D images to be transformed into a 3D scene viewable by a user. Despite the appealing photorealistic results the main disadvantages are low level freedom of the user and the high bandwidth needed in order to download the entire environments. Our solution offers the typical freedom of 3D virtual environments as for instance videogames. Moreover, in our framework there is no need to download the virtual environments while the user explore it.

Several works based on streaming geometry has been developed during years. In these approach only a small parts of the scene are transmitted. The user walk through a large scene residing entirely on a remote server and the client receive progressive data in order to match the rendering capabilities of the remote device and the available connection bandwidth. The reader can also refer for these works to [14, 20].

In [11] the authors proposed a system based on a streaming approach and an efficient use of the available bandwidth. Their approach use a bandwidth monitoring and rate control schemes to achieve high end-to-end throughput and low level frame rate fluctuation while the server use a rendering engine based on OpenGL. Our approach is to use an common PC with a state-of-the-art 3D engine in order to obtain a cutting-edge visualization technology on a thin clients as for instance a PDA.

Remote visualization is also useful when protection of modeling data is needed. In [17], the authors show that protection of 3D graphics content is possible while preserving widespread interaction. The challenge, in that case, is to employ server CPU-power to avoid client-side attacks that would allow reconstruction of the model geometry.

Remote visualization can be also hybrid, e.g., only part of the rendering is server-side and some graphical (2D) computation is required on client-side as well, as in [16], where couple of 2D line primitives are sent by the server to the client that renders locally on its screen.

3 System architecture

3.1 System hardware architecture

Our system hardware architecture is showed in Figure 1: the system is based on a simple 802.11b [13] WiFi network infrastructure which PDAs connect to.

The "rendering server" is an off-the-shelf PC with a quite common hardware configuration: CPU 3.40GHz Intel Pentium 4 with 1 Gb RAM, (motherboard Intel 945G Express) equipped with a *n*Vidia GeForce 7300 LE with 256Mb graphics board. This machine is relatively far from the computational power of graphical workstation and represents a common 1-2 years old PC that can be found.

The rendering server is connected to a common 100Mbit FastEthernet network. The WiFi network is implemented by an Alcatel Omniaccess 4308 appliance which coordinates 16 antennas that implements a WiFi covering over two floors in a building. The covering power of the WiFi network is not strictly functional but can be useful in some of the scenarios of use in which our system can be useful (see next sections). The WiFi network on which our early tests have been carried out is capable of providing bot 802.11b and 802.11g connectivity, that means both 11Mbit/sec and 54Mbit/sec capable connections.

We used 5 PDAs for the measurements, 3 of them are manufactured by HP and 2 are from Qtek. From the preliminary tests, it appears that the actual frame per seconds rate is bound by PDAs CPU speed, more than others details like processor models or system memory size:

- HP iPAQ H5550 equipped with a 400Mhz 32bit Intel XScale PXA255
- Qtek 2020i equipped with a 520Mhz 32bit Intel XScale PXA272
- Qtek S200 equipped with a 195Mhz 32bit Texas Instruments OMAP 850

All the 3 different models we used had an integrated 802.11b WiFi card, this means that even if our network infrastructure was able to provide an 802.11g connection we actually exploited an 802.11b connection that is, the available bandwidth was 11Mbit/sec.

3.2 System software architecture

The core of our system is the rendering engine that runs on the rendering server. The rendering engine we are using is Act-3D Quest3D [4], that is a complete system to implement interactive state-of-the-art virtual environments. It can be used to fastly design applications for visualization, games and screen-savers. We used version $3.5.2$ that includes support for artificial intelligence based on pathfinding algorithms, animations for large crowds and for vast amounts of vegetation, real-time shadows.

Everything in Quest3D is represented by "channels". A channel can store data or perform an action or both. For instance, for moving an object textured are needed three channel. One channel for the object, another channel for the texture and a third channel to translate the object.

Quest3D comes with many predefined drag-and-drop channels. These predefined channels are essentially C++ functions and often only wrappers for DirectX functions. An SDK is also available in order to extend Quest3D with new channels. In this way, new channels has been created for our application in a such way that can be possible to create any numbers of camera inside the virtual environments and each camera is driven by a remote device.

The camera mechanism has been used to provide the user interaction with the virtual environment: in Figure 3

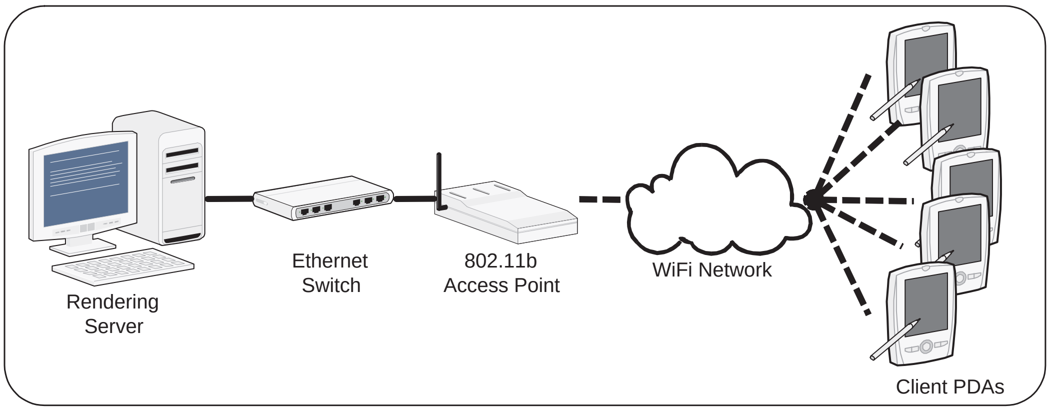

Figure 1: System hardware architecture

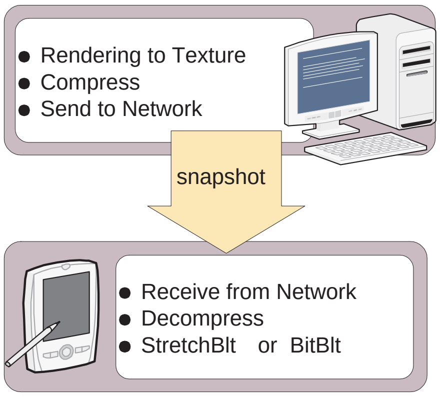

Figure 2: Rendering of a single frame.

is shown the application user interface that offers the traditional navigation capabilities that can be used by the user to walk, move the camera and to fly within the world.

The idea on which the system is based upon is quite straightforward: a PC, even a common off-the-shelf PC, has an hardware that is neither comparable to the hardware available on a PDA. Even more, the PC hardware can also exploit specialized units like the graphics hardware, that sports a GPU (Graphical Processing Unit) and at least 256Mb of specialized memory. All this available hardware can be used to render a frame for a PDA, and this frame can be sent to the PDA to remote visualize it. Considering the fact that a PDA screen usually offers quite small resolutions, not only the PC will be able to render a single frame, but could offer the rendering of more frames, one for each PDA connected to the system.

4 A reference scenario

Our research was motivated by the following scenario: building an interactive educational virtual environment based on a reconstruction of an archaeological site. The requirements of the reference scenario are the following:

- Moderate hardware/software requirements: the reconstruction must be able to be run on updated but currently available hw/sw platforms without requiring particular skills (nor funding!) to install it and maintain it.
- The reconstruction should be also used in stereo-projectors.
- The software architecture should allow easy interfaces with external devices, like augmented reality devices or location tracking positioning devices.
- The reconstruction should also allow cooperative and interactive visits.

4.1 The videogame and rendering engines

Our research has been focusing on the capabilities of the videogame and rendering engines to answer all these questions at once. As a matter of fact, 3D videogame engines are able to create highly engaging and immersive environments but are sparely used in spite of the commercial success and the high level of photorealism that is achieved by current software/hardware technology. The recent interest has stimulated research in the so called “Serious Games” witnessed by crowded tracks in the Computer Games Conferences and several well known non-ludic applications that are based on 3D videogame engines [8, 12, 18, 19]).

4.2 A real motivating example

Our interest in the remote exploration is generated by the collaboration with archaologists about the Paestum/Poseidonia archaeological site.

The town of Poseidonia, established by Greek settlers around the end of the seventh century B.C. on the left side of the river Sele in the Salerno gulf, flourished in the successive centuries because to its wide territory and to its agriculture and was embellished with magnificent temples. In the fourth century B.C. it was occupied by an italic people, the Lucanians and, successively, since 273 B.C., it became a Roman town and its name was changed in Pæstum. The Romans changed the aspect of the city and removed some monuments that were a symbol of the Greek autonomy, they also built some new buildings like the *Forum*, the *Comitium*, some *Templa*, the *Curia*, the *Amphitheatrum* and the *Thermæ*. The maximum prosperity was reached in the last century of the Roman Republic until the firsts two centuries of the Empire (100 B.C. - 200 A.D.).

Currently, the site is about 2 square Km and it can be visited. Our motivating stimulous was to explore the possibility of a remote exploration on PDAs, of a detailed reconstruction of the site [9], that allows high interaction [10], connected by wireless to a rendering server and to a GPS receiver (with Bluetooth) whose input is used to provide user-positioning within the site (as we show in [15]).

5 Results

Our system is made by two main entities the rendering server and the PDAs connected by the WiFi network infrastructure. We have tested both the rendering server and the PDAs under different circumstances and how the network reacted.

A preliminary analysis we carried out is aimed to measure the time distribution of the different phases of rendering a single frame on the PDA, it is made by 3 phases (see the lower part of Figure 2): the PDA receives the frame from the network in a memory buffer, the frame is decompressed from the memory buffer to an image structure that is finally drawn on screen. The whole process (Receive, Decompress and Draw) is carried out using standard PocketPC libraries. The draw operation can be performed in two ways, using the `BitBlt` function or the `StretchBlt` function, the difference lays, as the name may suggest, in the fact that the `StretchBit` also stretch the image at a desired resolution, while the `BitBlt` just draws it on the screen at its original resolution.

File format discussion. One important decision to be taken is the file format of the frame that is sent from the rendering server to the PDA. We tested two different file formats JPEG and Bitmap. JPEG [1] is a compressed lossy file format, Bitmap [7] is an uncompressed file format: using JPEG allows to transfer a smaller file that requests an phase of decompression, while the Bitmap file format means to transfer a larger file that does not need any decompression phase before drawing it. To support our decision some tests have been carried out for both file formats: the first three rows in Table 1 report the time span of every phase using JPEG file format, while in the fourth row is reported the Bitmap file format case, it is clear how the total time span (receiving time + decompression time) is smaller in the case of JPEG files because of the size of the file, this is also supported by the results of Table 3.

Resolution	Receive	Decompress	`BitBlt`
200x140	3ms	47ms	17ms
220x160	4ms	47ms	21ms
240x180	5ms	47ms	27ms
240x180	*135ms*	*3ms*	*12ms*

Table 1: Time distribution in visualizing a single frame on the PDA. The frame is drawn with the BltBit operation. The first three rows report the time using JPEG file format. The last row (in italics) reports the times using the Bitmap file format for the frame. (HP iPAQ H5550)

Resolution	Receive	Decompress	`StrBlt`
200x140	3ms	44ms	44ms
220x160	3ms	49ms	45ms

Table 2: Time distribution in visualizing a single frame on the PDA. The frame is drawn with the StretchBlt operation. (HP iPAQ H5550)

Resolution	Average size	Operation
200x140	6077 bytes	`BitBlt`
220x160	6855 bytes	`StretchBlt`
240x180	9120 bytes	`BitBlt`
240x180	*54382 bytes*	`BitBlt`

Table 3: Average size of the frame at different resolutions. BitBlt is just the visualization of the frame, StretchBlt is the stretched representation of the frame. The first three rows report the average size using JPEG file format. The last row (in italics) reports the averaeg using the Bitmap file format for the frame. (HP iPAQ H5550)

Draw function discussion. After the receive phase the file is transferred to the PDA, the image is decompressed and its pixels are kept in a data structure that will be used to draw the frame on the PDA screen. This last operation can be carried out in two ways, as already stated: `BitBlt` or `StretchBlt`. Transferring a frame with smaller resolution means to move a smaller file and to have a lighter phase of decompression but, on the other hand, request a stretching operation to finally draw it at the desired resolution. Table 2 clearly state that there is no advantage in moving and decompress a smaller file because the stretching operation is computational heavy.

6 Conclusions and future works

The prototype system is a proof of concept of an interesting way of approaching the problem of remote exploration. A comparison with traditional solutions for rendering on PDAs 3D scenes, like OpenGL ES [2], offers some thoughts:

- CPU power: PDA currently have slower CPUs respect to PC CPUs and our tests showed that the general performances of the system are bounded by the PDA CPU power.
- Battery life: a limiting factor for the CPU power is the battery life, a common mechanism to save battery is to slow down the CPU. This means that faster CPUs could mean less lasting PDA.
- Memory: a serious limitation to rendering of complex scene on a PDA is the tight memory real estate that implies that it will be hardly possible to keep a complex scene on the PDA.

It is worth noting that the whole system exploit off-the-shelf hardware for both the rendering server and the PDAs and does not require any particular network infrastructure allowing a cheap implementation.

6.1 Future works

The system offers some promising scenarios of use: the actual user experience can be expanded by developing complex virtual environment on the rendering PC: no matter how complex the scene is the system keep its interactivity because the PDA will keep visualizing frames as along as they arrive on the network, we tested scenes containing over 200000 polygons on 5 PDAs.

On the performance side further investigations are deserved by an image format more suitable for use on a PDA.

An interesting next step that we are planning is to let the whole system become a framework to host complex interactions among the users connected to it through avatars in order to provide functionalities like chat and online collaboration. In a wider sense the system can be used as a provider of interactive environments to a community of WiFi users, possible scenarios can be universities, implementing an information system for students, or a mall offering videogames to customers.

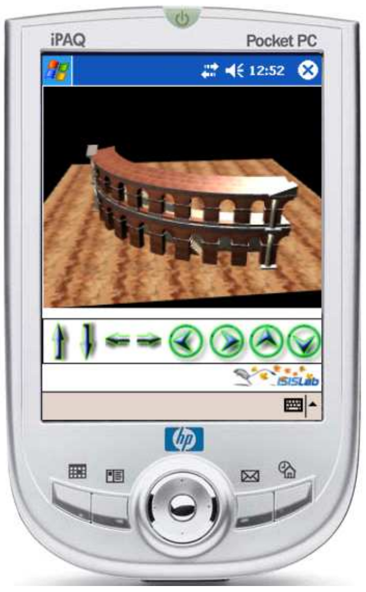

Figure 3: The current prototype's user interface.

References

[1] JPEG. `http://en.wikipedia.org/wiki/Jpeg`.

[2] OpenGL ES. `http://www.khronos.org/opengles/`.

[3] OpenGL Vizserver. `http://www.sgi.com/products/software/vizserver/`.

[4] Quest3d. `http://www.quest3d.com/`.

[5] QuickTime VR. `http://developer.apple.com/quicktime/`.

[6] The X.Org Foundation. `http://xorg.freedesktop.org/wiki/`.

[7] Windows and OS/2 bitmap. `http://en.wikipedia.org/wiki/Windows_bitmap`.

[8] Tim Altom, Melynda Buher, Michael Downey, and Anthony Faiola. Using 3D Landscapes to Navigate File Systems: The Mountain View Interface. In *Proc. of 8th Int. Conference on Information Visualisation (IV 2004), 14-16 July 2004, London, England*. IEEE Computer Society, 2004.

[9] Roberto Andreoli, Rosario De Chiara, Ugo Erra, and Vittorio Scarano. Interactive 3d environments by using videogame engines. In *IV '05: Proceedings of the Ninth International Conference on Information Visualisation (IV'05), London, England*, pages 515–520, Washington, DC, USA, 2005. IEEE Computer Society.

[10] Roberto Andreoli, Rosario De Chiara, Ugo Erra, Vittorio Scarano, Angela Pontrandolfo, Luigia Rizzo, and Alfonso Santoriello. An interactive 3D reconstruction of a funeral in Andriuolo's Necropolis in

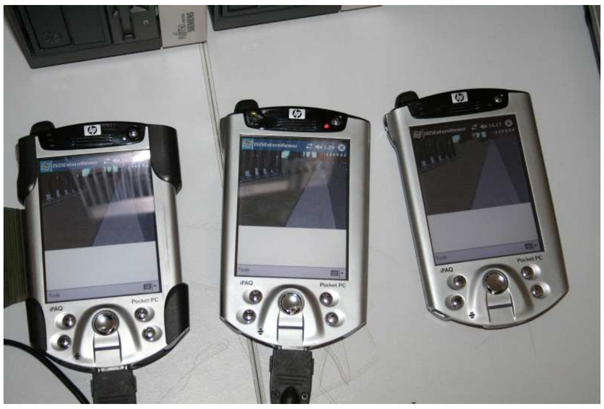

Figure 4: Three PDAs connected to the system

Paestum. In *CAA 2005 - Computer Applications and Quantitative Methods in Archaeology*, 2005.

[11] Azzedine Boukerche and Richard Werner Nelem Pazzi. Performance evaluation of a streaming based protocol for 3d virtual environment exploration on mobile devices. In *MSWiM '06: Proceedings of the 9th ACM international symposium on Modeling analysis and simulation of wireless and mobile systems*, pages 20–27, New York, NY, USA, 2006. ACM Press.

[12] Michael Christoffel and Bethina Schmitt. Accessing Libraries as Easy as a Game. In *Proc. of the 2nd Int. Workshop on Visual Interfaces for Digital Libraries. July 2002*, 2002.

[13] IEEE Computer Society LAN MAN Standards Committee. Wireless lan medium access control (mac) and physical layer (phy) specifications, 1997.

[14] Tapas K. Das, Gurminder Singh, Alex Mitchell, P. Senthil Kumar, and Kevin McGee. Neteffect: a network architecture for large-scale multi-user virtual worlds. In *VRST '97: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 157–163, New York, NY, USA, 1997. ACM Press.

[15] Rosario De Chiara, Valentina Di Santo, Ugo Erra, and Vittorio Scarano. Real positioning in virtual environments using game engines. In *Eurographics Italian Chapter Conference*, pages 197–203, 2007.

[16] J. Diepstraten, M. Grke, and T. Ertl. Remote Line Rendering for Mobile Devices. In *Proceedings of IEEE Computer Graphics International (CGI)'04*, pages 454–461, 2004.

[17] David Koller, Michael Turitzin, Marc Levoy, Marco Tarini, Giuseppe Croccia, Paolo Cignoni, and Roberto Scopigno. Protected interactive 3d graphics via remote rendering. *ACM Trans. Graph.*, 23(3):695–703, 2004.

[18] Jules Moloney and Lawrence Harvey. Visualization and 'Auralization' of Architectural Design in a Game Engione based Collaborative Virtual Environment. In *Proc. of 8^{t}h Int. Conference on Information Visualisation (IV 2004), 14-16 July 2004, London, England*. IEEE Computer Society, 2004.

[19] Eric Paquet and Herna L. Viktor. Visualisation, exploration and characterization of virtual collections. In *XXth Congress of the International Society for Photogrammetry and Remote Sensing, Commission V*, pages 597–602, July 2004.

[20] E. Teler and D. Lischinski. Streaming of complex 3D scenes for remote walkthroughs. In A. Chalmers and T.-M. Rhyne, editors, *EG 2001 Proceedings*, volume 20(3), pages 17–25. Blackwell Publishing, 2001.