Go to contents 16

JCAD-VR: Java Collaborative Architectural Design Tool in Virtual Reality

A Java3D based scalable framework for real-time, multi-platform VR environments

CONTI, Giuseppe; UCELLI, Giuliana; MAVER, Tom
ABACUS, University of Strathclyde, Glasgow, UK
http://www.strath.ac.uk/Departments/Architecture/

This paper proposes a framework that provides the architect with a tool that uses Virtual Reality (VR) as part of the design path. It offers the possibility to deploy a system capable of assisting the design profession during the early stages of the design process. This way VR becomes the means for a new experience where the architect can, free from constraints of the 2D world, create and manipulate the space she/he is designing.

The idea upon which JCAD-VR is being built is that all the users present in the virtual world have to be able to share the same virtual environment in a "transparent fashion" where the user interface, instead of the traditional menu/windows based layout, it is part of the virtual world itself.

The aim is to provide the designer with a tool for creating 3D-shapes in a shared VR environment, thus allowing the design to be shared as it evolves.

Keywords: *Collaborative Design, Virtual Reality, Java 3D, Distributed Environment.*

Introduction

Traditionally many architects have experienced the need to prove their design proposals using physical models: even the most accurate 2D paper representations are usually not suitable enough to explain and transmit the complexity of some architectonic ideas.

If the use of a third dimension is nowadays part of the daily practice, the "CAAD community" is only now experiencing the move from static representation, based on 2D renderings or pre-recorded animations (considered as a sequence of 2D images), to dynamically generated 3D representations. Real-time navigation and interaction, typical of VR environments, provide just that fluent interface and that entirety in the exploration of the design proposal that is the main lack in all CAD packages commonly in use.

Furthermore, the increasing growth of computational resources and hardware power eases the access to desktop VR applications making it a truly feasible approach in everyday practice.

Although VR is a quite mature technology, it is seldom utilised throughout the design process: often in fact it is just used as a more powerful presentation technique. Moreover, the recent growth of network-based virtual communities and the use of avatars have brought a new level of complexity to the meaning of virtuality, providing the technology for remote presence and collaborative experiences.

Background

In two case studies part of a previous research project of the authors (Ucelli, Conti, Lindsay and Ryder, 2000), the research team worked closely with both architects and engineers experimenting the use of VR in a real ongoing project.

Go to contents 16

During the creation of 3D models continuous access to all the information regarding the design progress and changing priorities were provided. Architects and engineers were invited to evaluate and discuss the project in a Virtual Reality lab.

The outcome of the research confirmed what other authors had stated (Dorta and LaLande, 1998): “the design process is made up of two activities which are graphic ideation or conceptual design and communication […] the first is a formative process dealing with creating and evolving ideas [active phase]; the second is a descriptive process aiming at presenting to others fully-formed ideas [passive phase] “[1].

Unfortunately the result of the experience, if on the one side it has proved the huge potential of VR for reducing designs “flaws”, on the other side it has highlighted that the present use of VR is limited only to the last phase of the design process, what we call the passive phase, when the visualization takes place and the result of the design process is eventually shown to the client (fig 1).

It is evident from the Figure 1 that the use of modeling and Rapid Prototyping (RP) is confined, with only a few exceptions (such in the innovative architectural practice of Frank O. Gehry) at the end of the creation process after which all the design choices have been taken.

The idea upon which the JCAD-VR framework is founded is to anticipate the use of VR within the active phase thus taking full advantage of the technology. The aim is to provide the designer with a tool for creating 3D-shapes in a shared VR environment, thus allowing the design to be shared as it evolves. This paper will report the present state of the JCAD-VR framework and will highlight its future development.

JCAD-VR: a framework

The idea upon which JCAD-VR is being built is that all the users present in the virtual world have to be able to share the same virtual environment in a “transparent fashion” where the user interface (UI), instead of the traditional menu/windows based layout, it is part of the virtual world itself.

The entire project is based on client-server architecture where every user logs into a virtual world and starts sharing designing tasks with other users. The entire structure is organised in an object-oriented

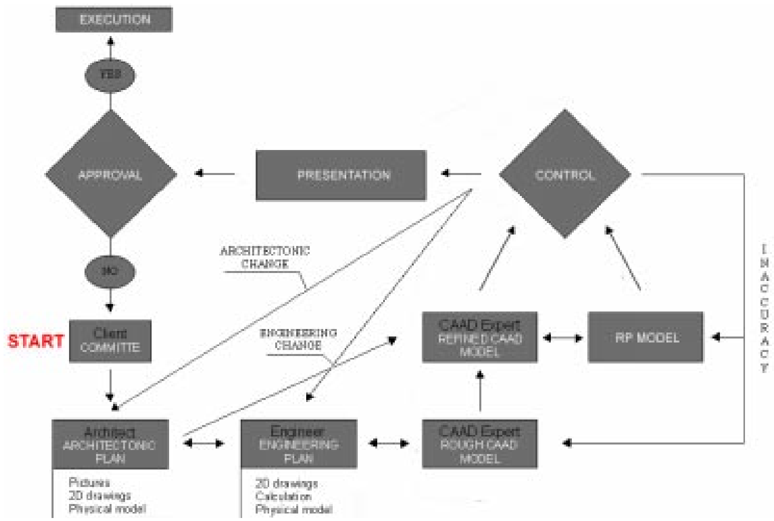

Figure 1. The role of visualisation inside the design process where the graphic ideation (active phase) is represented with the architectonic and engineering planning and the communication (passive phase) is represented by the CAAD/RP section.

Go to contents 16

fashion, where each module is able to fulfill to a certain task and it is independently coded. This approach has allowed the delivery of an initial functioning core of the JCAD-VR system whose capabilities will be expanded in the near future with several modules currently under development (fig 2).

The entire framework is handling the virtual environment through two closely connected parts: a 3D engine and a services unit each made of different modules. In a human body analogy the former might represent the heart while the latter might be considered as a nervous system.

The 3D engine unit

The 3D engine is the broad part of the framework that handles all the information regarding the “visible” aspects of the virtual world. It includes the code necessary to create and modify geometric entities (**geometry core**), to show the interface (**interface core**) and to deal with several different display devices (**visual core**).

The first module of the **geometry core** handles the creation of 2D and 3D objects: the last ones being both geometric primitives (cones, boxes, spheres etc.) and architectural entities (walls, slabs etc.). Although from the visualisation point of view a wall might be seen as a box, the system treats it in a completely different way. In fact while a box is just regarded as a simple shape without any further quality, the wall instead is handled as an entity owning “topological“ properties: it is first of all made of two different surfaces (internal and external faces) and a core. It can be the parent of another object (such as a window or a door) and it can hold other types of information such as number of windows attached to it.

Quite obviously the *geometry module* will also provide the means for attaching materials to objects and add lights and objects from a library to the virtual world through the *database module*. Further development will implement real-time shape recognition routines. This would allow the user the freedom of drawing shapes in the virtual world that

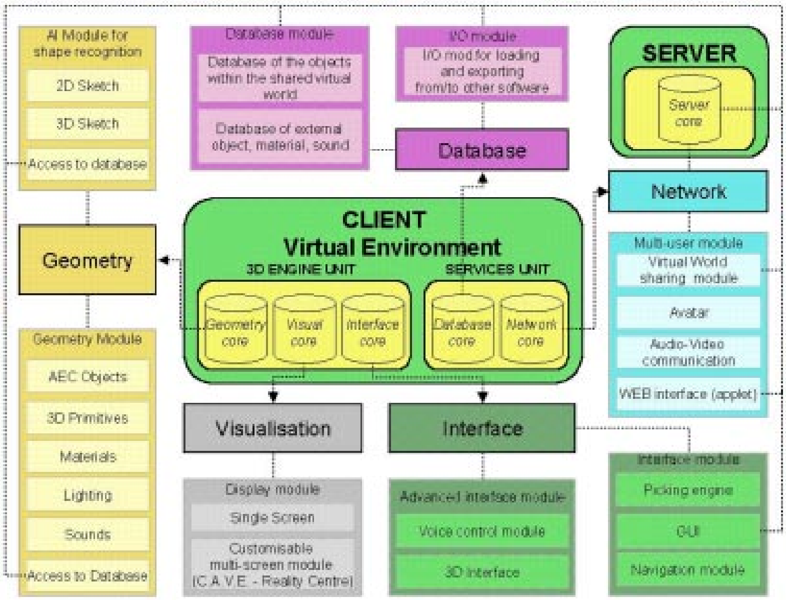

Figure 2. The general overview of the different modules

Go to contents 16

would be converted in 3D objects by the system. The complexity of such a process requires some Artificial Intelligence (AI) routines and it has convinced the authors to code the AI module only in a second stage of development.

The **interface core**, as the name itself says , obviously takes care of the graphic user interface (GUI). As previously mentioned the system aims to achieve an interface as “transparent” as possible. Here the term “transparency” is to be considered as the interface which is not a separate part of the 3D world but as an integral part of the virtual world itself. The idea behind it is that instead of the traditional menus and toolbars the user is immersed in an environment providing the means for the interaction itself. 3D menus pop up showing 3D icons and the 3D menus themselves can be moved for the convenience of the user. Visual feedback is provided, for example, in the form of rulers showing the size of objects or 3D icons showing the operation to be done on the object. A voice driven interface is due to be coded to help push further the level of “transparency” and enhance user friendliness of the interface. For the sake of completeness a traditional window based control panel is provided for advanced settings. On the first functioning core traditional pointing devices are used but support for 6-degree of freedom/virtual glove will be coded in the next future.

The **visual core** is the part of the framework that allows interfacing with the visualization devices. For the sake of flexibility the entire framework has been coded in a multi-platform language (Java). The obvious computational constrains imposed by the use of different hardware is solved by creating a structure that is flexibly scalable and can deliver images for a range of viewing devices, from the simple desktop monitor to the more complex tessellated screen [2] for immersive environments. The user can switch between two different modes according to the machine it is running on. When the software is loaded the user is asked to choose whether to work on a single screen or in multiple screen mode. At the present stage, on common PCs the video card displays the virtual world on a traditional window or on full screen. The system also runs on a Sgi supercomputer whose display is a Reality Centre. The internal architecture of this module is entirely flexible such that it might be easily adapted to allow use of other VR devices (C.A.V.E, H.M.D. etc.) (fig 3).

The services unit

The services unit is the part of the framework that handles all the information regarding the “management” of the virtual world. It is the backbone of the interconnection between users: it manages network connections and exchange of data between users (**network core**), it keeps tracks of the state of

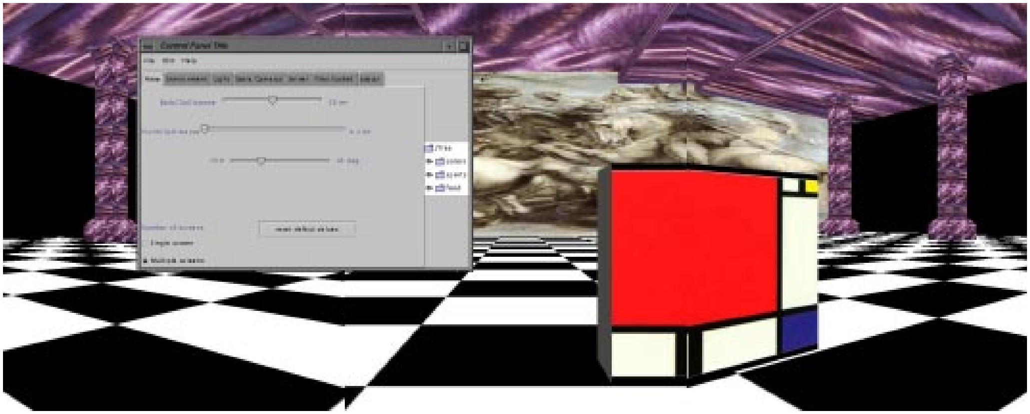

Figure 3. An image of JCAD-VR as it appears when running in the Reality Centre (multi-screen mode)

Go to contents 16

the virtual world in a database from which it also retrieves information from libraries of objects (**database core**).

As already mentioned the entire framework is based on a client/server architecture.

The services unit is indeed developed across two independently coded packages of the framework - *client* and *server*- and the **network core** is allowing the transmission of data between the two (fig 4).

The **network core** is thus based on a server, several clients and the network allowing the communication. The server is the data-delivering unit that looks after the information to be broadcast. The clients are the users themselves who perform actions and queries, when active, and when passive, rely on the server for receiving data update. The intrinsic multiplatform nature of JCAD-VR, inherited from the language used to code it, allows the server to transmit data to a broad range of machines, from normal PCs to the supercomputer running the Reality Centre, and leaves the research team the freedom to test the software with several operating systems. The communication channel ensures the link between server and clients through a TCP/IP network. As an independent part of the framework the server has an autonomous and simpler interface that provides primarily information about the network system. A number of components are envisaged such as the communication status, the users on line and VR shared environments. Since the clients are communicating through independent processes in the future a further enhancement will allow the server to be capable of dealing with several VR environments simultaneously. At the present stage the network module is supporting use of avatars representing the users inside the virtual world, and it gives also the possibility to interact between the users through a chat system and a whiteboard for sketching. Support for voice communication is being considered for further development as well as a web based applet version of the client side of the software.

The **database core** includes an internal database, that keeps track of the creation or manipulation of the

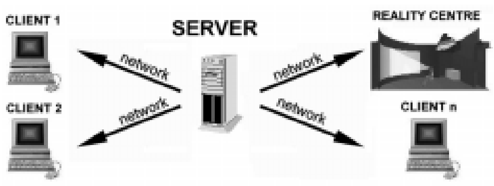

Figure 4. The client/server architecture of JCAD-VR where the server broadcasts to several clients including the Reality Centre.

objects in the virtual scene, and an external database through which users will be able to retrieve geometric primitives, materials, lights etc. The internal database is closely coupled with the **network core**. It is not only keeping track of what is happening in the user's virtual world but, most importantly, it receives, through the network, information sent by other users' internal databases. If a new object is created or its status is changed the system will upgrade the internal database of each user no matter who is doing the action. For the convenience of the user an I/O module will allow import/export of objects from/to other packages.

Technical overview

The entire system is coded in Java‘. The choice, even if less efficient in term of performances if compared with some other languages, offered indeed great flexibility, true scalability and last but not least fully multi-platform support. Moreover the use of Java‘ programming language became a natural choice when its 3D suite was released (Java3D‘). Its network-centric nature, its multimedia integration together with the use of native hardware acceleration (OpenGL) and multi-processors support (in the case of Sgi workstation) makes it the perfect choice for the development of a real-time multimedia collaborative system. Furthermore thanks to Java‘'s performance scalability and hardware independence the concept of CAAD has been pushed even further creating a VR environment that can co-exist between high-end supercomputers and common PCs.

The client application, in response to the obvious hardware limits imposed by the use of different hardware, has been written to be easily customised to run on PCs as well as on a Sgi supercomputer. The former are normal PCs whose video-card is displaying the virtual world only on a traditional window or at full screen. The latter is a 12-processors 6Gb Ram Sgi Onyx2 system running the Reality Centre at ABACUS, University of Strathclyde, Glasgow. When the JCAD-VR is launched on the Sgi it can take advantage of its computational power to stretch itself on a 5 metre wide 2 metre high tassellated screen where 3 Barco projectors create a 160 degree panoramic image.

Conclusions and further developments

The multidisciplinarity of this research is giving the opportunity to investigate collaborative design issues, the role of interfaces inside CAAD packages, the design process in the first stage of its conception, the use of Virtual Reality in architecture and last but non least, a number of technical issues.

As already pointed out, JCAD-VR is an ongoing project and several enhancements are planned for the next releases aiming to get that feeling of intuitiveness and that control over the design that should be the goal of every application using Virtual Reality.

Footnotes

[1] Dorta, T. and LaLande, P., 1998, p.144-148.

[2] A tessellated screen is a composite screen consisting of several projectors creating an unique high resolution image.

References

Ucelli, G., Conti, G., Lindsay, M. and Ryder, G.: 2000, from “Soft” to “Hard” Prototyping: A Unique Combination of VR and RP for Design, in proceedings from UkVRSig 2000, Glasgow.

Bertol, D.: 1997, Designing Digital Spaces, An architect’s guide to Virtual Reality, John Wiley & Sons, Inc.

Dorta, T. and LaLande, P.: 1998, The impact of Virtual Reality on the design process, in proceedings from ACADIA Conference, Quebec City.

Hughes, M., Hughes, C., Shoffner, M. and Winslow, M.: 1997, Java Network Programming, Manning, Greenwich.

Barrilleaux, J.: 2000, 3D User Interfaces with Java 3D, a guide to computer-human interaction in three dimensions, Manning, Greenwich.

Sowizral, H. A. and Deering, M. F.: 1999, The Java 3D API and Virtual Reality, IEEE Computer Graphics and Applications, May/June.