DISTRIBUTED VIRTUAL REALITY AUTHORING INTERFACES FOR THE WWW

I. Varlamis, M. Vazirgiannis

Dept of Informatics, Athens University of Economics & Business, Patision 76, 10434, Athens, HELLAS

I. Lazaridis

Dept. of Information and Computer Science, University of California, Irvine

ABSTRACT

Electronic commerce is emerging as an important domain of integration and enhancement of more specific technologies and research efforts. It is clear that the role of WWW in this context is a cornerstone as the medium of information dissemination. A trend in e-commerce is to provide to the potential customers the ability to view and "try" the products in a persuasive 3D representation. We have designed and implemented a system for WWW enabled interactive design & visualization of a room, definition of pieces of furniture and placement of domestic appliances. The system conveys a generic approach for distributed creation and update of virtual worlds as means of interaction and information dissemination in an e-commerce context.

1. INTRODUCTION

Electronic commerce is emerging as a domain of integration and enhancement of more specific technologies and research efforts. It is clear that Web's role in this context is a cornerstone as the medium of information dissemination. A trend in e-commerce is to provide to the potential customers the ability to view and "try" the products in a persuasive 3D representation. Moreover the users want to be able to view the products in their own environment.

We have designed and implemented a system for WWW enabled interactive design of a room, definition of pieces of furniture as well as placement of domestic appliances. The user may define for each object, its size, position, and also the inter-object spatial relationships. Certain integrity constraints are checked during this definition. The system generates on the fly a VRML representation of the specifications and renders it at the client. The user alters and visualizes the world and may save it for further reference.

The system architecture follows the 3-tier scheme where the client is independent of the specific scheme and architecture of the database that resides on the server. The database server contains information on the various products, along with links to the VRML descriptions of these products. The application server accesses the database contents, to serve the clients' requests. In the client side, a Java applet offers a highly interactive interface, communicating with the application server to provide the user with on-the-fly VRML-based 3D visualization. The VRML based room description, and all relevant files (images and VRML prototypes of furniture items and appliances of the room), are sent to the user in a compressed format which they user may save on his machines. Thus the Java security constraints are overcome safely. The system conveys a generic approach for distributed creation and update of virtual worlds as means of interaction and information dissemination in an e-commerce context.

2. Related Work

There are several tools, which produce enormous code for simple geometry (using more complex geometry –IndexedFaceSet- to define it with thousands of 3Dcoordinates, such as AC3D). In [4], SCORE, a distributed object oriented system for 3D real time visualization is presented. Though the system lacks methodology and a clear model for spatial navigation.

In [1], the concept of virtual environment (VE) is introduced. The main feature this system is its navigation means, which is spatial, and the interactivity facilities. VEs are described through three fundamental elements: context (for structuring the environment), object (which can be passive, reactive or active and movable or non-movable), and users that can activate objects, collect or move them. In [2] an interesting approach is presented for uniform modeling of geometry and behavior of a 3D graphics and animation framework. More specifically the authors specify separately geometry and behavior using corresponding nodes organized in two separate directed acyclic graphs, the geometry and the behavior graph. Moreover they provide rendering independent graphics objects that can be visualized by geometry nodes constrained by time and event-rated behavior nodes and finally provide high level 3D widget nodes that combine geometry and behavior nodes.

In [6] an approach for specifying virtual worlds is described. Spatiotemporal objects as integrated entities and thus apart from modeling their geometrical and positional characteristics (in an open and application-independent format), it also demonstrates how to relate them to any non-spatial attributes. Temporal evolution of objects is also taken into account. The requirement for maintaining a continuous view of an object through time is satisfied by linking two successive and explicitly stored states. Then, the state of an object at any point can be extracted by interpolating into the period formed by these states.

In [5] an integrated effort for building 3D animations is presented. The system provides a small set of modeling primitives, namely the graphical objects, their properties and the callbacks. The graphical objects are used for constructing scenes, the properties are time-variant and are used for animating various aspects of a scene while the callbacks are the primitives used to handle interaction.

3. MODEL

The VR-Shop data model aims at representation of: i. entities present in a room, ii. inter-entity spatial relationships, and iii. spatial constraints so as to produce a coherent, presentable world. Essentially, we used our experience in generalized spatiotemporal modeling [3] to address the simpler issue of developing an

authoring tool for rooms.

3.1 VR-SHOP Spatial Data Model

The VR-Shop Data model includes objects, spatial relationships between objects and constraints stemming from the real world limitations that the model must address. There are five different object classes. A vector of dimensions fully defines an instance of a class (i.e. actual object). Each object is also placed in the room by an additional vector of placement data.

3.1.1 Size and Placement Data

In Table 1 the reader may see the classes of objects along with their size-related attributes.

Object-Type	Attributes
Room	Length, Width, Height
Door	Width, Height
Window	Width, Height
Furniture Item	Length, Width, Height
Appliance	Length, Width, Height

Table 1. Size related attributes of object classes

Another issue is the placement of the objects in the context of the room. We aim at the definition of a set of primitives that define in a declarative way the relative placement of objects. In Table 2 we give the placement attributes describing the inter-object spatial relationship within the room spatial composition.

Object-Type	Attributes
Room	<none>
Door	OnWallName, Distance, FromWallName
Window	OnWallName, Distance, FromWallName, HeightFromFloor
Furniture Item	Distance1, FromWall1, Distance2, FromWall2, OrientationAngle, HeightChoice, [FromItem | Height]
Domestic Appliance	Distance1, FromWall1, Distance2, FromWall2, OrientationAngle, HeightChoice, [FromItem | Height]

Table 2. Placement-related attributes of object classes

By employing a wall identifier and a value that represents distance from the wall in consideration we define the position of Doors and Windows. For window objects, the height with reference to the floor is also required. Furniture Items and Domestic Appliances require the following attributes for their full specification:

- Position of their geometric center's projection on the ground plan (two distances from perpendicular walls).
- Orientation Angle, a rotation of the object around a vertical axis passing through the object's geometric center
- Height information: the object is at a specific height, or "on the floor" or "from the ceiling" or "on" another item.

3.1.2 Constraints

The use of high-level declarative predicates defined in the VR-Shop data model, provides expressive power but allows for inconsistencies. Thus we have to consider the related integrity constraints. The constraints arise both from the geometric configuration of the objects and/or from the attached semantics (i.e. some objects are pieces of furniture while others are appliances).

Explicit referential constraints we check in our design are:

1. All object dimensions must be less or equal than the dimensions of the room.
2. The physical dimensions and placement of objects must not allow part of them to be outside the room
3. The intersection of any pair of objects must be either empty or at most of a surface. Objects cannot intrude into one another, since they are solids.
4. Each object must have at least one common surface with another object or with the room (no objects suspended in mid-air).
5. Furniture items may be placed on other furniture items, but not on domestic appliances.
6. Upon removal of an object, all objects that are "on" that object must revert to a consistent position. Our solution was to place all such objects "on the floor".
7. Cyclic placements of objects "on" each other are not allowed.

The above list of constraints is by no means complete. Some semantic constraints have been avoided while others (#5) have been included. We might have disallowed the placement of certain types of device ("Refrigerator") on others ("VCR"). The freedom to perform such actions is left to the user, who will hopefully take care not to permit such inconsistencies.

4. AUTHORING TOOL

The aim is the retrieval of objects and their placement in a virtual room. The authoring tool is used to specify the room (dimensions, colors etc), and the objects in the room (size and placement)

The specification of the room is a three-step process: i. definition of the room shape, dimensions and color, ii. placement of doors and windows and iii. placement of any other object inside the room.

In the current version of the room editor, the room is rectangular; thus three parameters (length, width and height) are required. The four walls are identified as front, left, back and right.

Doors and windows have size (width and height). Their placement is defined by the wall they are on and the distance from another wall (plus distance from floor for windows). For example: "window W1 is on the left wall and 5m from the front wall".

The list of the available objects (furniture/appliances) is read from the database. Dimensions (length, width, and height) must be defined for pieces of furniture but are fixed for domestic appliances (a given TV model has known dimensions). The position of each object is specified using a set of parameters:

- Its minimum distance from a pair of adjacent walls.
- Its rotation around the vertical axis.
- Its position along the vertical axis.

The position of an object in a room is easily defined via the minimum distance from the nearest pair of walls. Thus we can easily say that an object is adjacent to a wall (minimum distance 0).

We allow rotation of objects only around the vertical axis since it is typical for furniture or appliances to be rotated around other

axes. To define the position of an object along the vertical axis, we provide 4 "level" options: i. on floor, ii. over another object, iii. at a height from floor, and iv. from the ceiling.

The room model previously described is complete enough to describe a usual room with its contents, but manually inputting all the necessary parameters is tiresome especially for an average Web user. Thus we added a 2D interactive space, which is both visualization and working area (see Figure 1).

One can set the room dimensions by dragging a rectangle in the drawing area. He can place doors, windows and other objects simply by clicking somewhere in the drawing area. Feedback from users indicates that the 2D interface is very useful as an overview method. The ability to precisely specify parameters with precision is retained but the 2D interface allows for the quick creation and visualization of rooms.

During the design phase, the user is able, at any time, to preview the room's state in a 3D VRML representation. This 3D preview is very helpful, both as a motivating factor (the user sees what he builds) and as a validation mechanism for possible design errors.
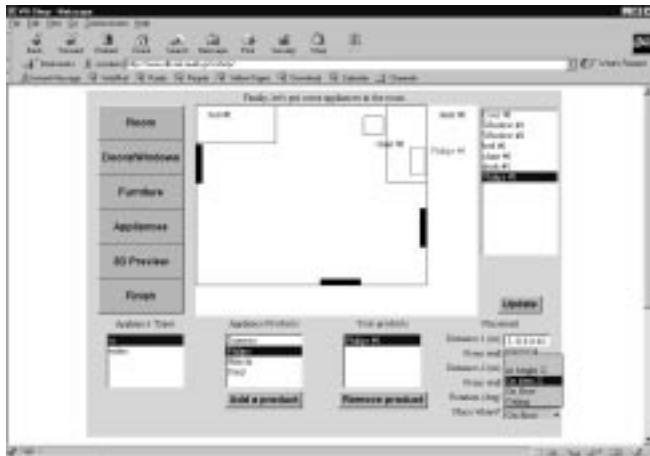

Figure 1. The 2D interface for interactive placement of objects in the room

5. Dynamic distributed VRML generation

The central idea of the proposed system is the provision of an authoring tool for distributed creation of 3D worlds that can be used across the WWW. This tool must integrate a database that contains information on available appliances. The generation of the VRML files is done either in the user's browser or in the server. The functionality of the three-tier architecture is presented below.

5.1 Room authoring tool

The room-authoring tool has to be available on the web. Thus we are concerned about the amount of data transferred across the network, security factors, friendliness and responsiveness of the interface etc.

A requirement for a web-based application is to limit the data size transferred per transaction. In our case only the parameters needed for the creation of the room rather than actual VRML code. The VRML file is then created locally at the user's machine.

The use of Java on the client side was a simple way to create the

functionality we wanted for a web-based tool. An added incentive was Java's interoperability with VRML. For fully exploiting the advantages of the Java platform in terms of architecture distribution, we used a three-tier architecture with an applet in the user browser, a Java application running on the server and a database server that communicates only with the application.
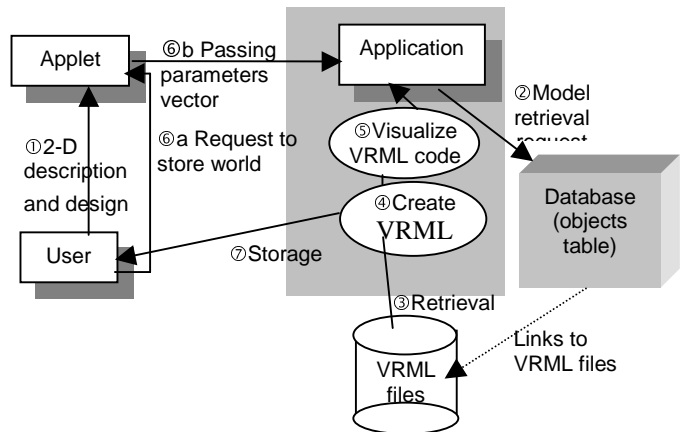

Figure 2. Room authoring tool architecture.

Before explaining in more technical terms the dynamic generation of the VRML world either in the server or the client it would be reasonable to describe the way this authoring tool works, as it is illustrated in Figure 2. The various tasks performed can be divided into two groups: i. interaction between the user and the application and ii. communication between the application and the database.

5.1.1 User-application communication

Interaction between the user and the application takes place in three distinct cases:

- When the user describes his room through the user-interface using both the 2D drawing area and the parameters input boxes. (step 1).
- Whenever the user selects to preview the room he designs and a 3D-visualization appears in his browser.
- When the user decides to store the room he designed in his local machine (steps 6 and 7).

In the first case the Java applet manages the user interface and input (see Figure 1). The application retrieves from the database all available appliances that can be placed in the room.

In the last two cases the main action that takes place is the creation of the VRML representation of the world. Although the cases look similar, the processing is completely different. In the case of the 3D preview all the processing is performed in the client machine, by the applet. When the user selects a 3D preview, the applet generates a VRML string and feeds it to the VRML browser. The result can be seen in Figure 3.

When the user asks for a 3D preview no data is transferred between the client and the server. All processing is done in the client. Thus, the 3D-preview process (which is quite common) does not incur any communication overhead.

Whenever the user requests to save the world locally, a file creation and packing process takes place in the server. A walk-around method is used to actually send the file to the client machine. The difference to the 3D-preview case is that the main

VRML file is temporarily created in the server and sent to the user. Only a vector of parameters is sent to the server (step 6b) where a file is created and along with all the necessary image and VRML files is sent to the user (step 7). This process happens only once, when the user finishes the design of the room; so the amount of data transferred through the whole authoring process is greatly reduced.
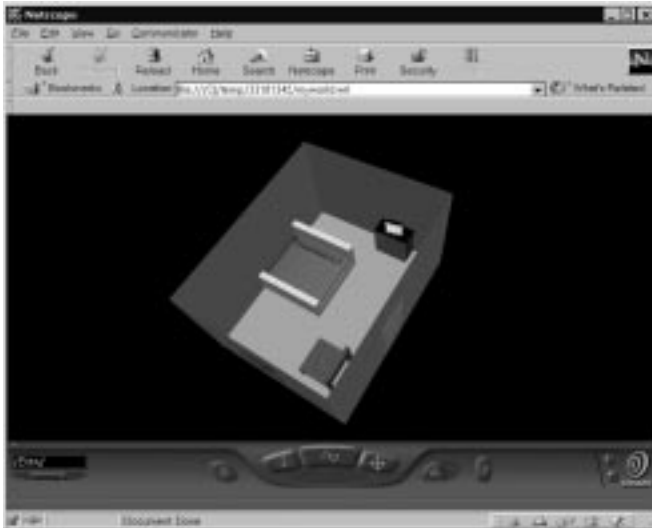

Figure 3. The visualization of the room during authoring

Communication between the application and the database occurs only for requesting the objects that are available for placement and the links to the relevant VRML files (steps 2 and 3).

5.1.2 Local storage of the produced world

An essential requirement in the creation of an authoring tool is the ability to store the result in a persistent format. In our case we also needed to store it locally, which is difficult because of the distributed nature of the application and the security constraints of Java applets. We created a multi-threaded server application that receives clients' requests. Whenever a user connects to our server a new thread of the application is spawned. When the user decides to store the final version of the room, all user-specified parameters are sent to the server via a simple protocol. The server-side Java application creates a VRML file, instantiating essentially a room from the user input. This file, other necessary VRML PROTO files and images are packed into a zip file. The only thing passed to the client (applet) is this file's URL address. The applet then initiates the download procedure. All this is hidden from the user who only gets a familiar "Save As" dialog.

When the user closes the connection, the server thread servicing this connection deletes the zip file from the server and exits. The storage process is illustrated in Figure 4.
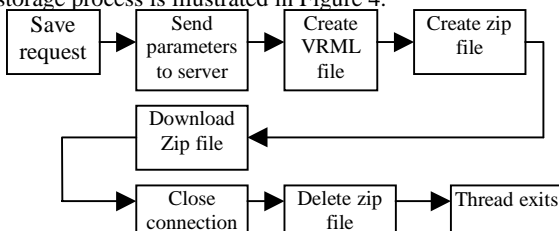

Figure 4. The "world storage" process

6. CONCLUSIONS

In the near past, efforts have been made towards the creation of environments that enable easier access to 3D visualization. There is a strong requirement for frameworks that fulfil such requirements even if the covered area is limited. Our effort could be classified in this category.

We have designed and implemented a system for WWW enabled interactive design of a room, definition of pieces of furniture as well as placement of domestic appliances. The user may define for each object, its size, position, and also the inter-object spatial relationships. Certain integrity constraints are checked during this definition. The system generates on the fly a VRML representation of the specifications and renders it at the client. The user alters and visualizes the world and may save it for further reference.

The contributions of our work are summarized in the following:

➢ declarative authoring model for creating virtual worlds putting emphasis on 3D spatial relationships and related spatial integrity constraints

➢ dynamic on the fly creation and modification of virtual reality worlds in a distributed environment.

➢ a generic, platform-independent approach for storage at the client site without violating the security aspects of Java

Further work will focus on extending the room model towards a generic approach for interactive 3D worlds. This effort will define interactive spatiotemporal scenarios in terms of complex 3D objects (any complex object can be defined through a set of elementary ones connected via a set of spatial relationships) and ECA-like rules that will provide a rich set of possible actions and ways to combine in them into fully-fledged interactive scenarios.

REFERENCES

[1] A. Diaz and R. Melster. "Patterns for modeling behavior in virtual environment applications", Second Workshop on Hypermedia Development: design patterns in hypermedia (in conjunction with Hypertext 99). Darmstadt, Germany, February 1999.

[2] J. Dollner and K. Hinrics, "Object-oriented 3D modeling, animation and interaction", in The Journal of Visualization and Computer Animation, vol. 8:33-64, 1997.

[3] M. Erwig, R. H. Gueting, M. Schneider, M. Vazirgiannis, "Spatio-Temporal Data Types: An Approach to Modeling and Querying Moving Objects in Databases", GeoInformatica Journal, Kluwer Publishers, to appear, 1999.

[4] R. Melster, A. Diaz and B. Groth. "SCORE - The virtual museum, development of a distributed, object-oriented system for 3D real-time visualization". Technical report 1998-15, TU Berlin, Germany. October 1998.

[5] Marc Najork, Mark Brown , "Obliq-3d: A high Level, Fast Turnaround 3D Animation System", , IEEE Transactions on Visualization and Computer Graphics, vol.1, no 2, June 1995,

[6] A. Vakaloudis, B. Theodoulidis. "The storage and querying of 3D objects for the dynamic composition of VRML worlds". Chorochronos Workshop, Aalborg, Denmark. June 19 - 20, 1998.