Extending WWW to support Platform Independent Virtual Reality

David Raggett, Hewlett Packard Laboratories

(email: dsr@hplb.hpl.hp.com)

Abstract

This is a proposal to allow VR environments to be incorporated into the World Wide Web, thereby allowing users to "walk" around and push through doors to follow hyperlinks to other parts of the Web. VRML is proposed as a logical markup format for non-proprietary platform independent VR. The format describes VR environments as compositions of logical elements. Additional details are specified using a universal resource naming scheme supporting retrieval of shared resources over the network. The paper closes with ideas for how to extend this to support virtual presence teleconferencing.

1 Introduction

This paper describes preliminary ideas for extending the World Wide Web to incorporate virtual reality (VR). By the end of this decade, the continuing advances in price/performance will allow affordable desktop systems to run highly realistic virtual reality models. VR will become an increasingly important medium, and the time is now ripe to develop the mechanisms for people to share VR models on a global basis. The author invites help in building a proof of concept demo and can be contacted at the email address given above.

VR systems at the low end of the price range show a 3D view into the VR environment together with a means of moving around and interacting with that environment. At the minimum you could use the cursor keys for moving forward and backwards, and turning left and right. Other keys would allow you to pick things up and put them down. A mouse improves the ease of control, but the "realism" is primarily determined by the latency of the feedback loop from control to changes in the display. Joysticks and SpaceBalls improve control, but cannot compete with the total immersion offered by head mounted displays (HMDs). High end systems use magnetic tracking of the user's head and limbs, together with devices like 3D mice and datagloves to yet further improve the illusion.

Sound can be just as important to the illusion as the visual simulation: The sound of a clock gets stronger as you approach it. An aeroplane roars overhead crossing from one horizon to the next. High end systems allow for tracking of multiple moving sources of sound. Distancing is the technique where you get to see and hear more detail as you approach an object. The VR environment can include objects with complex behavior, just like their physical analogues in the real world, e.g. drawers in an office desk, telephones, calculators, and cars. The simulation of behavior is frequently more demanding computationally than updating the visual and aural displays.

The Virtual environment may impose the same restrictions as in the real world, e.g. gravity and restricting motion to walking, climbing up/down stairs, and picking up or putting down objects. Alternatively, users can adopt superpowers and fly through the air with ease, or even through walls! When using a simple interface, e.g. a mouse, it may be easier to learn if the range of actions at any time is limited to a small set of possibilities, e.g. moving forwards towards a staircase causes you to climb the stairs. A separate action is unnecessary, as the VR environment builds in assumptions about how people move around. Avatars are used to represent the user in the VR environment. Typically these are simple disembodied hands, which allow you to grab objects. This avoids the problems in working out the positions of the user's limbs and cuts down on the computational load.

2 Platform Independent VR

Is it possible to define an interchange format for VR environments which can be visualized on a broad range of platforms from PCs to high-end workstations?

At first sight there is little relationship between the capabilities of systems at either extreme. In practice, many VR elements are composed from common elements, e.g. rooms have floors, walls, ceilings, doors, windows, tables and chairs. Outdoors, there are buildings, roads, cars, lawns, and trees etc. Perhaps we can draw upon experience with document conversion and the Standard Generalized Markup Language (SGML) [ref. 4] and specify VR environments at a logical level, leaving browsers to fill in the details according to the capabilities of each platform.

The basic idea is to compose VR environments from a limited set of logical elements, e.g. chair, door, and floor. The dimensions of some of these elements can be taken by default. Others, like the dimensions of a room, require lists of points, e.g. to specify the polygon defining the floor plan. Additional parameters give the color and texture of surfaces. A picture frame hanging on a wall can be specified in terms of a bitmapped image.

These elements can be described at a richer level of detail by reference to external models. The basic chair element would have a subclassification, e.g. office chair, which references a detailed 3D model, perhaps in the DXF format. Keeping such details in separate files has several advantages:

- Simplifies the high level VR markup format

This makes it easier to create and revise VR environments than with a flat representation.

- Models can be cached for reuse in other VR environments

Keeping the definition separate from the environment makes it easy to create models in terms of existing elements, and saves resources.

- Allows for sharing models over the net

Directory services can be used to locate where to retrieve the model from. In this way, a vast collection of models can be shared across the net.

- Alternative models can be provided according to each browser's capabilities.

Authors can model objects at different levels of detail according to the capabilities of low, mid and high end machines. The appropriate choice can be made when querying the directory service, e.g. by including machine capabilities in the request. This kind of negotiation is already in place as part of the World Wide Web's HTTP protocol [ref. 3].

Limiting VR environments to compositions of known elements would be overly restrictive. To avoid this, it is necessary to provide a means of specifying novel objects, including their appearance and behavior. The high level VR markup format should therefore be dynamically extendable. The built-in definitions are merely a short cut to avoid the need to repeat definitions for common objects.

3 Universal Resource Locators (URLs)

The World Wide Web uses a common naming scheme to represent hypermedia links and links to shared resources. It is possible to represent nearly any file or service with a URL [ref. 2].

The first part always identifies the method of access (or protocol). The next part generally names an Internet host and is followed by path information for the resource in question. The syntax varies according to the access method given at the start. Here are some examples:

- `http://info.cern.ch/hypertext/WWW/TheProject.html`

  This is the CERN home page for the World Wide Web project. The prefix "http" implies that this resource should be obtained using the hypertext transfer protocol (HTTP).

- `http://cui_www.unige.ch/w3catalog`

  The searchable catalog of WWW resources at CUI, in Geneva. Updated daily.

- `news:comp.infosystems.www`

  The Usenet newsgroup "`comp.infosystems.www`". This is accessed via the NNTP protocol.

- `ftp://ftp.ifi.uio.no/pub/SGML`

  This names an anonymous FTP server: `ftp.ifi.uio.no` which includes a collection of information relating to the Standard Generalized Markup Language - SGML.

3.1 Application to VR

The URL notation can be used in a VR markup language for:

- Referencing wire frame models, image tiles and other resources

For example, a 3D model of a vehicle or an office chair. Resources may be defined intensionally, and generated by the server in response to the user's request.

- Hypermedia links to other parts of the Web.

Major museums could provide educational VR models on particular topics. Hypermedia links would allow students to easily move from one museum to another by "walking" through links between the different sites.

One drawback of URLs is that they generally depend on particular servers. Work is in progress to provide widespread support for lifetime identifiers that are location independent. This will make it possible to provide automated directory services akin to X.500 for locating the nearest copy of a resource.

4 MIME: Multipurpose Internet Mail Extensions

MIME describes a set of mechanisms for specifying and describing the format of Internet message bodies. It is designed to allow multiple objects to be sent in a single message, to support the use of multiple fonts plus non-textual material such as images and audio fragments. Although it was conceived for use with email messages, MIME has a much wider applicability. The hypertext transfer protocol HTTP uses MIME for request and response message formats. This allows servers to use a standard notation for describing document contents, e.g. image/gif for GIF images and text/html for hypertext documents in the HTML format. When a client receives a MIME message the content type is used to invoke the appropriate viewer. The bindings are specified in the mailcaps configuration file. This makes it easy to add local support for a new format without changes to your mailer or web browser. You simply install the viewer for the new format and then add the binding into your mailcaps file.

The author anticipates the development of a public domain viewer for a new MIME content type: video/vrml. A platform independent VR markup language would allow people to freely exchange VR models either as email messages or as linked nodes in the World Wide Web.

5 A sketch of the proposed VR markup language (VRML)

A major distinction appears to be indoor and outdoor scenes. Indoors, the scene is constructed from a set of interconnected rooms. Outdoors, you have a landscape of plains, hills and valleys upon which you can place buildings, roads, fields, lakes and forests etc. The following sketch is in no way comprehensive, but should give a flavour of how VRML would model VR environments. Much work remains to turn this vision into a practical reality.

5.1 Indoor scenes

The starting point is to specify the outlines of the rooms. Architects drawings describe each building as a set of floors, each of which is described as a set of interconnected rooms. The plan shows the position of windows, doors and staircases. Annotations define whether a door opens inwards or outwards, and whether a staircase goes up or down. VRML directly reflects this hierarchical decomposition with separate markup elements for buildings, floors, rooms, doors and staircases etc. Each element can be given a unique identifier. The markup for adjoining rooms use this identifier to name interconnecting doors. Rooms are made up from floors, walls and ceilings. Additional attributes define the appearance, e.g. the color of the walls and ceiling, the kind of plaster coving used to join walls to the ceiling, and the style of windows. The range of elements and their permitted attributes are defined by a formal specification analogous to the SGML document type definition.

Rooms have fittings: carpets, paintings, book cases, kitchen units, tables and chairs etc. A painting is described by reference to an image stored separately (like inlined images in HTML). The browser retrieves this image and then applies a parallax transformation to position the painting at the designated location on the wall. Wall paper can be modelled as a tiling, where each point on the wall maps to a point in an image tile for the wall paper. This kind of texture mapping is computationally expensive, and low power systems may choose to employ a uniform shading instead. Views through windows to the outside can be approximated by mapping the line of sight to a point on an image acting as a backcloth, and effectively at infinity. Kitchen units, tables and chairs etc. are described by reference to external models. A simple hierarchical naming scheme can be used to substitute a simpler model when the more detailed one would overload a low power browser.

Hypermedia links can be represented in a variety of ways. The simple approach used in HTML documents for depicting links is almost certainly inadequate. A door metaphor makes good sense when transferring to another VR model or to a different location in the current model. If the link is to an HTML document, then an obvious metaphor is opening a book (by tapping on it with your virtual hand?). Similarly a radio or audio system makes sense for listening to a audio link, and a television for viewing an MPEG movie.

5.2 Outdoor scenes

A simple way of modelling the ground into plains, hills and valleys is to attach a rubber sheet to a set of vertical pins of varying lengths and placed at irregular locations: $z_i = f_i(x, y)$. The sheet is single valued for any x and y, where x and y are orthogonal axes in the horizontal plane. Smooth terrain can be described by interpolating gradients specified at selected points. The process is only applied within polygons for which all vertices have explicit gradients. This makes it possible to restrict smoothing to selected regions as needed.

The next step is to add scenery onto the underlying ground surface:

- Texture wrapping - mapping an aerial photograph onto the ground surface.

This works well if the end-user is flying across a landscape at a sufficient height that parallax effects can be neglected for surface detail like trees and buildings. Realism can be further enhanced by including an atmospheric haze that obscures distant details.

- Plants - these come in two categories: point-like objects such as individual trees and area-like objects such as forests, fields, weed patches, lawns and flower beds.

A tree can be placed at a given (x, y) coordinate and scaled to a given height. A range of tree types can be used, e.g. deciduous (summer/fall), and coniferous. The actual appearance of each type of tree is specified in a separate model, so VRML only needs the class name and a means of specifying the model's parameters (in many cases defaults will suffice). Extended objects like forests can be rendered by repeating an image tile or generated as a fractal texture, using attributes to reference external definitions for the image tile or texture.

- Water - streams, rivers and water falls; ponds, lakes and the sea. The latter involves attributes for describing the nature of the beach: muddy estuary, sandy, rocky and cliffs.
- Borders - fences, hedges, walls etc. which are fundamentally line-like objects
- Roads - number of lanes, types of junctions, details for signs, traffic lights etc.

Each road can be described in terms of a sequence of points along its center and its width. Features like road lights and crash barriers can be generated by default according the attributes describing the kind of road. Road junctions could be specified in detail, but it seems possible to generate much of this locally on the basis of the nature of the junction and the end points of the roads it connects: freeway-exit, cloverleaf junction, 4-way stop, round-about etc. In general VRML should avoid specifying detail where this can be inferred by the browsing tool. This reduces the load on the network and allows browsers to show the scene in the detail appropriate to the power of each platform. Successive generations of kit can add more and more detail leading to progressively more realistic scenes without changes to the original VRML documents.

- Buildings - houses, skyscrapers, factories, filling stations, barns, silos, etc.

Most buildings can be specified using constructive geometry, i.e. as a set of intersecting parts each of which is defined by a rectangular base and some kind of roof. This approach describes buildings in a compact style and makes it feasible for VRML to deal with a rich variety of building types. The texture of walls and roofs, as well as the style of windows and doors can be defined by reference to external models.

- Vehicles, and other moving objects

A scene could consist of a number of parked vehicles plus a number of vehicles moving along the road. Predetermined trajectories are rather unexciting. A more interesting approach is to let the behavior of the set of vehicles emerge from simple rules governing the motion of each vehicle. This could also apply to pedestrians moving on a side-walk. The rules would be defined in scripts associated with the model and not part of VRML itself. The opportunities for several users to meet up in a shared VR scene are discussed in the next section.

- Distant scenery, e.g. a mountain range on the horizon

This is effectively at infinity and can be represented as a back cloth hung in a cylinder around the viewer. It could be implemented using bitmap images (e.g. in GIF or JPEG formats). One issue is how to make the appearance change according to the weather/ time of day.

- Weather and Sky

Outdoor scenes wouldn't be complete without a range of different weather types! Objects should gradually lose their color and contrast as their distance increases. Haze is useful for washing out details as the browser can then ignore objects beyond a certain distance. The opacity of the haze will vary according to the weather and time of day. Fractal techniques can be used to synthesize cloud formations. The color of the sky should vary as a function of the angle from the sun and the angle above the horizon. For VRML, the weather would be characterized as a set of predetermined weather types.

- Distancing

The illusion will be more complete if you can see progressively more detail the closer you get. Unfortunately, it is impractical to explicitly specify VR models in arbitrary detail. Another approach is to let individual models to reference more detailed models in a chain of progressively finer detail, e.g. a model that defines a lawn as a green texture can reference a model that specifies how to draw individual blades of grass. The latter is only needed when the user zooms in on the lawn. The browser then runs the more detailed model to generate a forest of grass blade.

5.3 Actions and Scripts

Simple primitive actions are part of the VRML model, for example to ability of the user to change position/orientation and to pick up/put down or "press" objects. Other behaviour is the responsibility of the various objects and lies outside the scope of VRML. Thus a virtual calculator would allow users to press keys and carry out calculations just like the real thing. This rich behaviour is specified as part of the model for the calculator object class along with details of its appearence. A scripting language is needed for this, but it will be independent of VRML, and indeed there could be a variety of different languages. The format negotiation mechanism in HTTP seems appropriate to this, as it would allow browsers to indicate which representations are supported when sending requests to servers.

5.4 Achieving Realism

Another issue, is how to provide realism without excessive computional demands. To date the computer graphics community has focussed on mathematical models for realism, e.g. ray tracing with detailed models for how objects scatter or transmit light. An alternative approach could draw upon artistic metaphors for rendering scenes. Paintings are not like photographs, and artists don't try to capture all details, rather they aim to distill the essentials with a much smaller number of brush strokes. This is akin to symbolic representations of scenes. We may be able to apply this to VR. As an example consider the difficulty in modelling the folds of cloth on your shirt as you move your arm around. Modelling this computationally is going to be very expensive, perhaps a few rules can be used to draw in folds when you fold your arms.

6 Virtual presence Teleconferencing

The price performance of computer systems currently doubles about every 15 months. This has happened for the last five years and industry pundits see no end in sight. It therefore makes sense to consider approaches which today are impractical, but will soon come within reach.

A world without people would be a dull place indeed! The markup language described above allows us to define shared models of VR environments, so the next step is to work out how to allow people to meet in these environments. This comes down to two parts:

- The protocols needed to ensure that each user sees an up to date view of all the other people in the same virtual location, whether this is a room or somewhere outdoors.
- A way of visualising people in the virtual environment, this in turn begs the question of how to sense each user - their expressions, speech and movements.

For people to communicate effectively, the latency for synchronizing models must of order 100 milliseconds or less. You can get by with longer delays, but it gets increasingly difficult. Adopting a formal system for turn taking helps, but you lose the ability for non-verbal communication. In meetings, it is common to exchange glances with a colleague to see how he or she is reacting to what is being said. The rapid feedback involved in such exchanges calls for high resolution views of people's faces together with very low latency.

A powerful technique will be to use video cameras to build real-time 3D models of people's faces. As the skull shape is fixed, the changes are limited to the orientation of the skull and the relative position of the jaw. The fine details in facial expressions can be captured by wrapping video images onto the 3D model. This approach greatly reduces the bandwidth needed to project lifelike figures into the VR environment. The view of the back of the head and the ears etc. are essentially unchanging and can be filled in from earlier shots, or if necessary synthesized from scratch to match visible cues.

In theory, the approach needs a smaller bandwidth than conventional video images, as head movements can be compressed into a simple change of coordinates. Further gains in bandwidth could be achieved at a cost in accuracy by characterizing facial gestures in terms of a composition of "identikit" stereotypes, e.g. shots of mouths which are open or closed, smiling or frowning. The face is then built up by blending the static model of the user's face and jaw with the stereotypes for the mouth, cheeks, eyes, and forehead.

Although head mounted displays offer total immersion, they also make it difficult to sense the user's facial expressions. They are also uncomfortable to wear. Virtual presence teleconferencing is therefore more likely to use conventional displays together with video cameras mounted around the user's workspace. Lightweight headsets are likely to be used in preference to stereo or quadraphonic loudspeaker systems, as they offer greater auditory realism as well as avoiding trouble when sound spills over into neighboring work areas.

The cameras also offer the opportunity for hands free control of the user’s position in the VR environment. Tracking of hands and fingers could be used for gesture control without the need for 3D mice or spaceballs etc. Another idea is to take cues from head movements, e.g. moving your head from side to side could be exaggerated in the VR environment to allow users to look from side to side without needing to look away from the display being used to visualize that environment.

7 Where Next?

For workstations running the X11 windowing system, the PEX library for 3D graphics is now available on most platforms. This makes it practical to start developing proof of concept platform independent VR. The proposed VRML interchange format could be used within the World Wide Web or for email messages. All users would need to do is to download a public domain VRML browser and add it to their mailcaps file. The author is interested in getting in touch with people willing to collaborate in turning this vision into a reality.

References

[1] “*Hypertext Markup Language (HTML)*”,
Tim Berners-Lee, January 1993.
URL=ftp://info.cern.ch/pub/www/doc/html-spec.ps
or http://info.cern.ch/hypertext/WWW/MarkUp/MarkUp.html

[2] “*Uniform Resource Locators*”, Tim Berners-Lee, January 1992.
URL=ftp://info.cern.ch/pub/www/doc/url7a.ps
or http://info.cern.ch/hypertext/WWW/Addressing/Addressing.html

[3] “*Protocol for the Retrieval and Manipulation of Texual and Hypermedia Information*”,
Tim Berners-Lee, 1993.
URL=ftp://info.cern.ch/pub/www/doc/http-spec.ps
or http://info.cern.ch/hypertext/WWW/Protocols/HTTP/HTTP2.html

[4] “*The SGML Handbook*”, Charles F. GoldFarb, pub. 1990 by the Clarendon Press, Oxford.