A Survey of Level of Detail Support in Current Virtual Reality Solutions[†]

Martin Reddy

Dept. of Computer Science
University of Edinburgh
Mayfield Road, EH9 3JZ

e-mail: M.Reddy@ed.ac.uk

Keywords*: level of detail, virtual reality, interactivity, performance.*

Abstract: The technique of Level of Detail (LOD) offers a powerful method of reducing the computational burden of a virtual reality (VR) system. As a result, it represents a valuable and important facility which one would expect to find in all serious VR graphics systems. This survey aims to present many of the rendering engines commonly employed in state-of-the-art VR solutions and details the degree of support which these systems provide for LOD. The investigation reveals a significant lack of support for this facility over the range of packages reviewed. Consequently, a call is made for improved LOD support in future VR products.

1 Introduction

Virtual reality (VR) is an extremely computationally demanding paradigm: requiring the simulation and display of a virtual world at real-time update rates, generally assumed to be at least 15Hz. Level of Detail (LOD) offers a means of improving the performance of a VR system under certain circumstances by trading visual detail for speed (Astheimer & Pöche, 1994). This is done by storing a number of representations of an object, each varying in complexity (e.g. polygon count), and then selecting an appropriate model to use at each frame of the simulation: a more complex representation will appear more detailed, but it will consume more compute-time.

The most common selection criterion used to modulate an object's LOD is its distance from the viewpoint (distance LOD), i.e. lower detail models are employed as the object becomes more distant. Indeed, it is this case which most people associate with the term LOD. It should be noted however that other modulation schemes are possible: such as relating LOD to an object's motion (Hitchner & McGreevy, 1993) or peripheral location (Reddy, 1995). Also, another application of LOD is to automatically balance the system's computational load in an attempt to maintain a fixed frame rate (Funkhouser & Séquin, 1993).

Due to the performance benefits which can be accrued through the use of LOD in the time-critical domain of VR, one might conclude that it offers an invaluable asset which should be incorporated into every serious VR graphics system. The following survey of VR packages was embarked upon to investigate the reality of this expectation, and to inform the VR community of the support which exists for LOD at the current juncture.

[†] This article was first published in *Virtual Reality: Research, Development and Application* (1995) Vol. 1, No. 2, pp. 85–88; a publication of the Virtual Reality Society. (c) Virtual Press, 147a High Street, Waltham Cross, Herts, EN8 7LN.

2 Discussion

Throughout the following discussion, the term 'manual' will refer to an action which must be explicitly programmed at the application level: whereas 'automatic' refers to an action which is implemented and transparently controlled by the VR system proper. For each package which is reviewed, the product's capability to manually select LOD is assessed, along with any support for automatic distance LOD. If the package offers any other relevant features, then these will be reported too.

1. **2Morrow V1.4** from 2Morrow Inc.: 2Morrow is a PC-based set of tools for building VR applications. It features a 3D Modeler, World Builder and Runtime (walkthrough) Module; as well as a Function Library of C-callable routines. Manual LOD switching is supported and can be performed by the *setrep* command within the 2Morrow scripting language. The system also provides support for automatic distance LOD, based upon the pixel area occupied by the object on-screen. This information is defined in the object's .PLG file.

2. **AVRIL V2.0** by Bernie Roehl: AVRIL is a freely-available VR graphics library derived from REND386 and which runs on a PC (80386 or higher). The Application Program Interface (API) supports the manual selection of an object's representation via two mechanisms. These are the *vrl_ObjectSetShape()* and *vrl_ObjectSetRep()* functions which can be used to select the desired object shape and the LOD representation of that shape, respectively. AVRIL also supports an automatic distance LOD feature which is governed by the pixel area of an object's bounding sphere in screen space. This is defined within the object's .PLG file by specifying the on-screen pixel size at which each LOD should be chosen (as with the 2Morrow system described above). N.B. Comparable LOD facilities exist in the related REND386 and VR386 packages.

3. **BRender V1.1** from Argonaut Software: BRender (Blazing Render) is a real-time 3D graphics library which is available on PC, Mac and various game console platforms. The BRender system allows the manual switching of an object's LOD by attaching a custom callback function to the object. In this situation, instead of rendering the object, control is always passed to a user-defined function. This can be used to perform the desired LOD operations and then call the *ZbModelRender()* function. There is no intrinsic support provided for automatic distance LOD. The user would therefore have to implement this manually through the use of a callback function.

4. **Cyberspace Developer Kit V2.0** from Autodesk, Inc.: The CDK package is an object-oriented 3D graphics library for creating interactive visualisations and VR simulations on a PC (80386 or higher). The CDK allows the manual switching of an object's visual representation under the programmer's control. The API also offers functions to support automatic distance LOD (this is implemented by monitoring the displacement of each object from the viewpoint).

5. **DIVE V3.0** by the Swedish Institute of Computer Science. DIVE (Distributed Interactive Virtual Environment) is a fully distributed heterogeneous VR system which is freely-available (for non commercial purposes) on SGI, RS6000 and Sun platforms. In DIVE, an object's visual representation can be manually switched by removing and adding a view to an object. However, these changes are then distributed to all other peers with an additional function call. This therefore does not provide true support for

manual LOD because it means that any object must be presented to all participants at the same detail level; whereas LOD should be modulated independently for each peer. There is no support for automatic distance LOD in the current version of DIVE. Having said this, it is only fair to note that the DIVE system was developed with the primary aim of producing a VR system to support distributed interactions. As such, graphics optimisation techniques like LOD have not been a major concern in the initial DIVE development.

6. **dVS V3.0** from Division Ltd.: dVS is a software environment for developing VR applications based upon a distributed, multi-user architecture. It consists of a runtime module as well as an optional C-callable library. dVS provides support for both manually selected LOD and automatic distance LOD. The latter is computed based upon the distance between the viewer and a definable reference point within the object. The *VCVisual_SetLod()* function can be used to force the renderer to display a particular LOD, overriding the automatic selection. The Division system also comes with a number of utilities to automatically generate various LOD models in the native .BGF format. These include the *bgfreduce* program which reduces the polygonal complexity by regenerating facets, and the *bgfblast* utility which reduces the complexity of circular and rectangular sections.

7. **Open Inventor V2.0** from Silicon Graphics Inc.: Inventor is an object-oriented 3D toolkit which is available on a number of platforms including the SGI, Sun and PC. Manual LOD selection under the Inventor system can be performed using the *SoSwitch* group node which traverses a single child node depending upon the value of a selection field. Automatic distance LOD is also possible and is implemented via the *SoLevelOfDetail* group node. This selects a particular child node depending upon the rectangular area occupied by the object's bounding box after it has been projected into screen space (Wernecke, 1993). A further interesting feature of the Inventor toolkit is the *SoComplexity* node, which in effect offers a method of generating various LOD representations for certain internal primitives (e.g. circles, cones etc.). It enables the programmer to specify the visual complexity of subsequent primitives, offering control over the number of polygons and the degree of texture mapping used.

8. **MR Toolkit V1.4** by the University of Alberta. The MR (Minimal Reality) Toolkit offers a set of tools for producing VR applications on numerous platforms including the SGI, DEC, HP and RS6000. It is freely-available to non-profit organisations and consists of a collection of subroutine libraries, device drivers and other utilities. There is no direct support for automatic distance LOD in MR Toolkit; however, users on SGI workstations can use the package in conjunction with IRIS Performer, and so they can access all of the LOD features of Performer (see below). The MR Toolkit authors are currently working on adding real-time extensions for version 2.0. Therefore it is not expected that any new features will be implemented (i.e. improved LOD support) until this next release has been completed.

9. **IRIS Performer V1.2** from Silicon Graphics Inc.: Performer is a suite of highly optimised 3D graphics libraries which can be used to implement real-time, interactive graphics applications on an SGI. The Performer libraries provide support for LOD operations; with the ability to smoothly fade between the different representations (on RealityEngine^{TM} machines). Automatic distance LOD is implemented by supplying a value for the centre of the object (to be used for the distance calculation) and a range of distance values defining the selection thresholds for each LOD. One extremely powerful

feature of Performer is that it can automatically modulate the LOD of each object in order to maintain a desired fixed frame rate.

10. **Reality Lab V2.0** developed by RenderMorphics Ltd. and subsequently acquired by Microsoft Corp.: Reality Lab (RL) is a 3D graphics library offering fully textured, Gouraud and Phong shaded rendering across a number of platforms including the PC, Sun, SGI, Mac and Sony PSX, among others. Manual LOD can be implemented by removing and replacing visual objects in the current view (e.g. using the RLFrameAddVisual() function and its counterpart). RL does not provide any automatic mechanism to modulate LOD. Any such support must therefore be implemented by the programmer using callback functions.

11. **RenderWare V1.4** from Criterion Software Ltd.: RenderWare is a real-time 3D graphics library which is available on Sun, PC and Mac platforms. The graphics library allows the manual selection of different representations of an object by switching between various 'clumps'. However, there is no support for automatic distance LOD; consequently this is left as the responsibility of the application. The author has been informed by Criterion Software Ltd. that automatic distance LOD is being considered for a future release, although it will not be included in V2.0 of the package.

12. **Superscape VRT V4.0** from Superscape Ltd.: The VRT (Virtual Reality Toolkit) is an integrated solution for building VR applications on a PC. It features a number of modules including a Shape Editor, World Designer, Visualiser, Texture Editor and a Sound Editor. VRT supports the manual selection of an object's LOD through the use of its C-like scripting language, SCL (Superscape Control Language). It also supports an automatic distance LOD feature which can be accessed through the menu-driven interface. This is instigated by supplying the threshold distance values and the names of the LOD models to use via the built-in Distance Editor.

13. **VREAM V1.1** from VREAM, Inc. (also shipping as **VRCreator V2.0**): VREAM is a graphically-driven VR authoring system for the PC. It contains a world editor, runtime walkthrough system and an extensive scripting language. There are a number of ways of manually selecting an object's LOD in VREAM. These can be implemented using the VREAMScript programming language and/or external programs. Although full automatic distance LOD is not strictly supported, VREAM does allow the user to switch texture mapping off when objects exceed a certain distance (implemented with a VREAM world attribute).

14. **VRML V1.0**: VRML (Virtual Reality Modeling Language) is a platform-independent, open file format for transmitting and displaying virtual environments over the Internet. The VRML specification includes an *LOD* group node which provides an automatic distance LOD facility. This is realised by specifying the centre point of the object (which will be used for the distance calculations) and providing a list of distances in increasing order. The distance values are used to select the appropriate child node (LOD model) to use depending upon the object's distance from the viewpoint (Bell *et al.*, 1995). Manual LOD selection can be performed with the *Switch* node, similar in operation to the *SoSwitch* facility of Open Inventor (see above). However, because VRML is a file format and is not programmable, the scope for manual LOD is limited.

15. **WorldToolKit V2.0** from Sense8: WorldToolKit is a VR graphics library which runs on SGI, Sun, PC, DEC and HP platforms. The package supports manual switching

of object representations via the *WTobject_remove()* and *WTobject_add()* functions. It does not at this time support automatic distance LOD, but Sense8 have informed the author that the next release of WorldToolKit will include scene hierarchies with full automatic distance LOD support. In addition, a particularly interesting feature of the WorldToolKit package is the inclusion of the *WTobject_levelofdetail()* function which can be used to generate various LOD models (off-line) from an original model using a proprietary polygon reduction algorithm. This works by accepting a distance value and the current viewing angle. It then returns a simplified model which is indistinguishable from the original object under those viewing conditions.

3 Results

Table 1 below collates the results of the survey into a tabular representation. This provides a concise summary of the findings, enabling a rapid inspection and comparison of all LOD related features. Within the table, the first column presents the title of the VR package. The second column (Manual LOD) records whether the VR system allows manual switching of an object's visual representation. Column Three (Distance LOD) records whether the product incorporates an automatic distance LOD feature. The fourth column (Load Balancing) records whether the system can automatically modulate the LOD of all objects in an attempt to maintain a fixed frame rate. And finally, the fifth column (LOD Generation) states whether any provision is made to automatically create different LOD models from an original polygon object (N.B. some VR packages provide a 3D Modeler which may allow the manual removal of polygons from a model. This is not taken as supported for automatic LOD generation).

Package Title	**Manual LOD**	**Distance LOD**	**Load Balancing**	**LOD Generation**
2Morrow V1.4	Yes	Yes	No	No
AVRIL V2.0	Yes	Yes	No	No
BRender V1.1	Yes	No	No	No
CDK V2.0	Yes	Yes	No	No
DIVE V3.0	Yes	No	No	No
dVS V3.0	Yes	Yes	No	Yes
Open Inventor V2.0	Yes	Yes	No	Limited
MR Toolkit V1.4	Yes	No	No	No
IRIS Performer V1.2	Yes	Yes	Yes	No
Reality Lab V2.0	Yes	No	No	No
REND386/VR386	Yes	No	No	No
RenderWare V1.4	Yes	No	No	No
Superscape VRT V4.0	Yes	Yes	No	No
VRML V1.0	Limited	Yes	No	No
VREAM V1.1	Yes	Limited	No	No
WorldToolKit V2.0	Yes	No	No	Yes

Table 1: *Summary of support for LOD in common VR packages.*

4 Conclusion

From the above presentation, we can see that there are two principal ways of implementing distance LOD: either by calculating the distance of the object from the viewpoint, or by determining the area of the object's bounding volume on the screen. It is also apparent that most VR packages enable the manual switching of object representations, although only around half (56.3%) of the packages reviewed support a means of automatically modulating an object's LOD based upon its displacement from the viewpoint.

Another point which is made clear in the above discussion is the marked lack of VR products which provide a facility to automatically generate different LOD models (12.5% of those reviewed). That is, a program which takes an original polygon description of an object and creates another description of that object, retaining the same general form but containing less polygons. This deficiency implies that the user must create these representations by hand—probably with a separately purchased modeling/CAD package—which can prove to be an approximate and time-consuming task. Many people may advocate that LOD generation is the task of the modeling/CAD package anyway; however, it would appear that support for this facility is poor in most commercial modelers too (e.g. it is not included as standard in AutoCAD, 3D Studio, 2Morrow, Ez3d etc.).

In summary, LOD is an essential and powerful tool for VR. In spite this, many VR packages do not fully support this facility yet. It is therefore hoped that improved support for LOD will become apparent in future software releases, and that a number of commercially available LOD generation programs will appear to support the growth of LOD in VR.

References

Astheimer, Peter, & Pöche, Mariea-Luise. 1994 (August 23–26). Level-Of-Detail Generation and its Application in Virtual Reality. *Pages 299–309 of: Proceedings of the VRST'94 conference.*

Bell, Gavin, Parisi, Anthony, & Pesce, Mark. 1995 (26 May). *The Virtual Reality Modeling Language: Version 1.0 Specification.* (Available on-line from VRML Repository WWW Page).

Funkhouser, Thomas A., & Séquin, Carlo H. 1993. Adaptive Display Algorithm for Interactive Frame Rates During Visualization of Complex Virtual Environments. *Computer Graphics Proceedings, Annual Conference Series*, 247–254.

Hitchner, Lewis E., & McGreevy, Michael W. 1993. Methods for user-based reduction of model complexity for Virtual Planetary Exploration. *Pages 622–36 of: Proceedings of the SPIE - The International Society for Optical Engineering*, vol. 1913.

Reddy, Martin. 1995. Musings on Volumetric Level of Detail for Virtual Environments. *Virtual Reality: Research, Development, Applications*, **1**, 49–56.

Wernecke, Josie. 1993. *The Inventor Mentor: Programming Object-Oriented 3D Graphics with Open Inventor(TM), Release 2.* Reading, MA: Addison-Wesley.