VRIC, Virtual Reality International Conference, Laval Virtual 2001, May 16-18

# Graphics Performance Benchmarking Based on VRML Browsers

Jiri Zara, Jaroslav Krivanek
Department of Computer Science and Engineering,
Czech Technical University, Prague, Czech Republic
{zara, xkrivanj}@fel.cvut.cz

**Keywords: VRML, benchmark, browser, graphics performance**

*Abstract: The set of benchmarks for graphic performance has been designed and implemented taking practical users' needs into consideration. As 3D technology begins to be a very common part of Internet presentations and applications, high interest of appropriate hardware and software combination for virtual reality can be expected. This paper introduces benchmarks based on usage of different VRML browsers. In contrast with testing of basic but low-level tasks (e.g. number of triangles per second), our approach is oriented to a real throughput of practically used computers. Measurements on various computer configurations are presented and discussed.*

## 1. Introduction

The measurement of graphics performance for certain computer and its configuration is important not only for hardware producers and software developers but also for ordinary users.

Benchmarks can be divided into two main categories. The first class contains tests for basic units, either hardware or software. This is the case of well-known benchmarks like SPECint2000 and SPECfp2000 for the measurement of CPU performance or SPECglperf [4] for the OpenGL library. The second category of benchmarks is oriented to real applications thus it seems to be more appropriate for users' needs. Examples are benchmarks like AWadvs for Alias|Wavefront users or ProCDRS for CAD designers working with Pro/Designer etc.

The idea and thereafter an application presented in this paper belong to the second class of benchmarks. Based on the observation that VRML is going to be widely accepted for presentation of 3D data on the Internet, we have decided to develop a set of benchmarks that uses VRML browsers as tools for the testing of overall graphics performance. While this high-level benchmarking gives direct benefit to ordinary users, the comparison of different VRML browsers is additional but desirable side effect. These benchmarks can thus help users to check the performance of their computers as well as to select appropriate plug-in for viewing VRML content.

It should be stressed that presented benchmarks do not deal with the conformance of browsers to VRML specification. This task is already handled and very well maintained by VRML Conformance Working Group [2, 3].

The structure of this paper is as follows. Section 2 introduces a basic methodology used for benchmarking and lists various tests specific to different purposes. Results of measurements of selected VRML browsers are presented in section 3. Discussion on browsers and different configurations is placed in section 4. Section 5 concludes the paper and informs about intended extensions to presented set of benchmarks.

## 2. Design of Benchmarks

We have concentrated on two basic issues when testing VRML browsers and their performance – *memory* requirements and a *speed* of rendering. Since VRML browsers do not provide any means of how to measure a memory allocated for rendering of certain VRML world, no regular measurement of memory requirements has been implemented yet. Moreover, the access to memory manager subsystem depends on operating system and there is no simple way of getting information about memory allocated by plug-in running within WWW browser. That is why we have finally decided to measure the memory requirements individually, typically by reading values provided by system programs like Task Manager in MS Windows.

Our recommended practice is to read the absolute value of the memory used by all processes in the system after WWW browser has been started but before a VRML plug-in has been invoked. Then a user should read the second value of memory usage when specific VRML test scene is loaded and rendered. The difference of these two values gives

a global overview on the memory allocated both by VRML plug-in itself and its additional data structures for certain 3D scene. Although a detailed scientific analysis could further separate and discuss the size of those two chunks of memory, we consider the one common value fully sufficient for benchmark purposes.

Four specific VRML scenes have been selected from the whole set of more than forty tests for checking the memory requirements. These tests are described in section 2.2.

embedded into a HTML document. This fact has leaded to unified design of all benchmarks consisting of two separate parts as shown on Fig. 1 – VRML browser window and measuring applet.

Rendering area has a constant size of 500x500 pixels for most testing scenes. The choice of a VRML browser plug-in depends on the user. The only requirement is that a browser must have implemented the EAI interface. Currently we support the former EAI specification from January 1997 and the last one will be supported as soon as

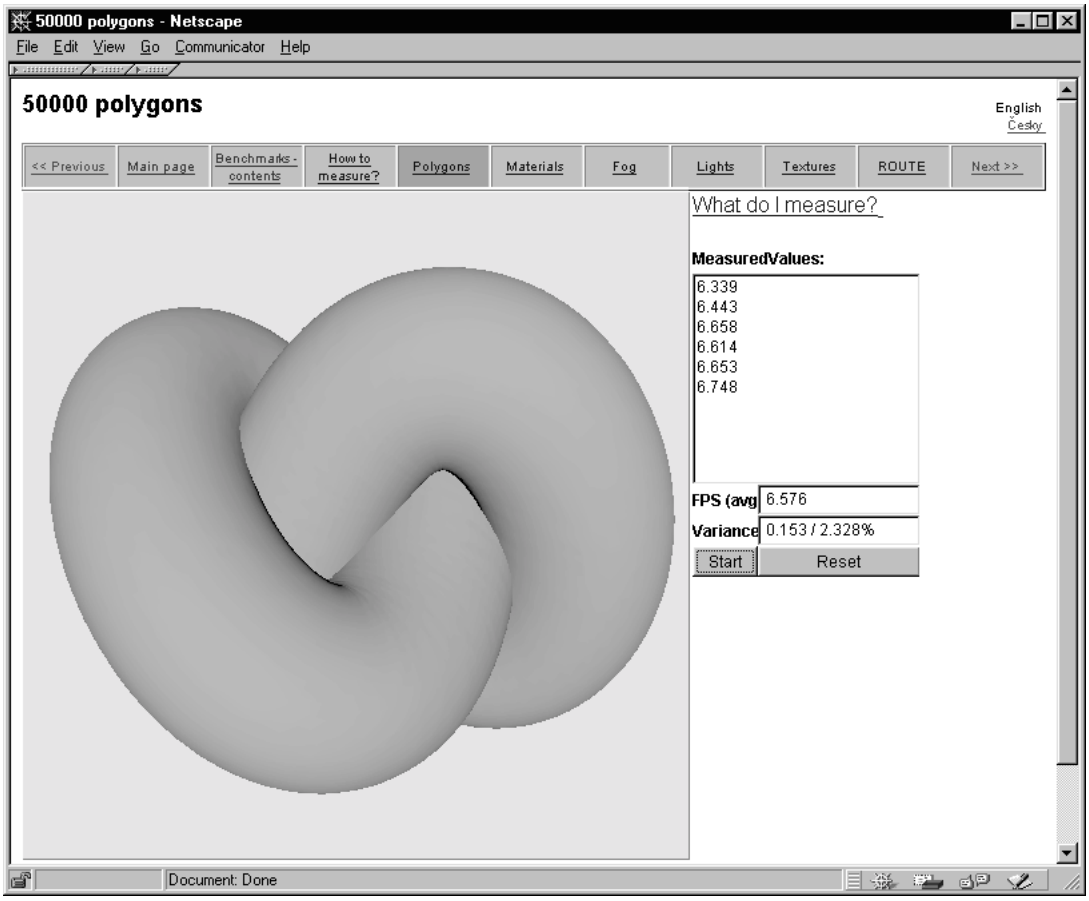



**Fig. 1.** Benchmark environment – a window containing VRML browser rendering area on the left, measuring applet on the right

*Speed measurement* is much easier than maintaining memory requirements. Here the commonly accepted parameter is *fps* (frames per second). All VRML browsers provide current value of fps in Browser Script Interface both for ECMAScript and Java language. Unfortunately, repeatedly performed checking of fps via ECMAScript called from Script node slows down the rendering speed considerably. Our observations show that fps can be decreased even by 10% when a simple Script node is added to a VRML scene and Browser.getCurrentSpeed() method is invoked in a loop.

When a Java applet is connected to a VRML browser using EAI [1], rendering speed is influenced minimally and it has almost the same value as for a standalone VRML browser window

browsers will widely implement it.

Each VRML scene used for testing contains simple animation construction that rotates a scene in an infinite loop. Scenes differ in many characteristics, either in quantity (number of polygons, lights, textures, etc.) or in specific features (materials, light sources, texture mapping). All scenes used for benchmarking are listed in section 2.2.

Measuring Java applet is constructed quite simply. A VRML browser is periodically (each 0.5 s) queried for current fps. A small window displays last 20 values and arithmetic mean is updated continuously together with sigma (standard error). To avoid the unstable situation when new VRML scene is loaded into a browser and internal data structures are built, old fps values are disposed of.

The user has always a possibility to reset the measuring process and start it from the beginning.

## 2.1 Configuration Issues

In contrast of benchmarks for single objects like CPU or OpenGL library, the measurement of overall graphics performance depends on configuration of various components. Here is the list of the most important ones:

- CPU, FPU and bus architecture
- Graphics card (3D accelerator)
- Memory (RAM, graphics card)
- Operating system (windowing system)
- Rendering library (OpenGL/Direct 3D)
- WWW browser and VRML browser

Since the hardware configuration is mostly given and unchangeable for a certain benchmarked computer we have concentrated on last two components only – rendering library and VRML browser.

Two widely used rendering libraries are OpenGL and Direct 3D (D3D). They are often implemented into hardware of graphics cards. While OpenGL is multi-platform, D3D has been primarily designed for MS Windows only. Some VRML browsers support both these libraries, some utilize just one of them and some browsers have their own special rendering library. That is why the users should be aware of configuration of a browser and 3D graphics card. Our measurements show that the differences among rendering libraries are noticeable and differ for various benchmarks.

## 2.2 Measured Characteristics

The rendering process is affected by a number of factors. The speed of rendering depends on number of polygons, material used, number and characteristics of light sources, number and size of textures, number of transformations etc. Since there is no way how to construct a universal, all-purpose VRML scene, a set of simplified single-purpose tests has been developed. These benchmarks are subdivided into the following categories:

- Polygons
- Materials and Fog
- Lights
- Textures
- Event processing

Each category contains several tests graduated from simple to complex content. Sample color images at the end of this paper show typical scenes for each category.

### 2.2.1 Polygons

This group of benchmarks has been designed for ”pure” polygon rendering without any special features like textures, specular reflections etc. Eight different meshes from 1.000 triangles up to 50.000 triangles represent the same polygonal shape. Only diffuse reflection and the avatar’s headlight are taken into the consideration. Example of such a scene is on Figure 1. For standard situation, the rendering speed should be either constant or decreased in correspondence with number of polygons. A sudden reduction of fps means mostly a lack of memory. The scene with 50.000 triangles is recommended for checking of memory requirements.

### 2.2.2 Material and Fog

A specific test for each material parameter (diffuseColor, specularColor, shininess, emissiveColor, and transparency) is performed on the same polygonal object (with 10.000 triangles) like in previous group of benchmarks. Finally a scene with all material parameters set to non-zero values is measured. This set of material benchmark is then repeated for the same scenes enhanced by eight directional lights (see also color images C.1 – C.3).

Processing of fog is tested by only one scene consisting of several 3D objects placed in various distances from the viewer (image C.4).

### 2.2.3 Light Sources

Three sets of benchmarks (each for one type of VRML light node – DirectionalLight, PointLight, SpotLight) have been prepared (see image C.5). In each set the same object is lighted gradually by 4, 8, and finally 12 light sources of the same type. Here the restriction to maximum of 8 sources in many OpenGL implementations often causes wrong visible results when all 12 lights are switched on. Small picture of expected and properly rendered image is thus provided in corresponding benchmark page to help a user to recognize this kind of error.

### 2.2.4 Textures

Two sets of benchmarks test two different parameters of texture rendering – a size and a number of textures. One texture in various resolutions is mapped on one large polygonal object (image C.6). The dimension of rectangular texture images increases from 1 pixel, through 512 and 2048 up to 4096 pixels. The last case is memory exhaustive, since the raw texture data represents about 50 MB.

Another scene has been designed for the second set of texture benchmarks. It consists of 1024 objects and each object is modeled by eight triangles. Some objects rotate in an infinite loop. Then 1, 512, or 1024 different textures are applied to those objects (see color images C.7 and C.8). Although the size of individual textures is reasonably small (128x128 pixels), the big amount of textures requires a lot of texture memory.

During the design of benchmarks we were interested how VRML browsers maintain the case when the same image is applied as a texture for

many objects but without using a DEF/USE statement. We have prepared a scene for this special purpose. The same texture is mapped on 1024 objects without any USE. Some browsers are able to recognize this ”trick” and to load the image just once, some not.
Three selected texture benchmarks are suggested for testing memory consumption – a scene with 1024 objects without textures, with 512 different textures and with 1 texture individually placed on every object (without USE statement).

### 2.2.5 Processing Events

The last set of benchmarks has been constructed with the aim to check how the number of events processed in VRML browser influences the rendering speed. Two scenes with large event cascade have been designed. One TimeSensor sends events to 2.000 receiving nodes or to 4.000 nodes respectively. Events are not further processed, since receiving nodes simply ignore their value.

## 3. Measurements

The whole package of benchmarks has been currently used in MS Windows environment only, either with Netscape or MSIE web browsers. The following four VRML plug-ins were tested:

- blaxxun Contact [6]
- Cortona [7]
- CosmoPlayer [8]
- WorldView [9]

Table 1 contains a subset of various configurations tested. The aim was not to test the best PC available on the market, but rather to show how various components influence the graphic performance.
Tables 2 – 5 are organized by benchmark specific tests i.e. each table represents data related to different graphic performance issue. Again, we have selected only some of data for this paper – polygons, materials, lights, and textures. Configuration numbers are the same as in the Table 1. All speeds bigger than 100 fps are trimmed to 100 fps. The reason for that is to set the fair conditions for comparison. Some VRML browsers and 3D libraries do this restriction automatically although the real speed could be much higher.

**Table 1.** Configurations used for benchmarks

| Cfg. | Computer + Graphic card + OS | WWW browser | VRML browser | Renderer |
|---|---|---|---|---|
| | Celeron 400Mhz/256MB + Riva TNT2 Ultra 32 MB + Win NT | | | |
| **1** | | MSIE 5.0 | Cortona 3 | SW |
| **2** | | MSIE 5.0 | Cortona 3 | OpenGL |
| **3** | | Netscape 4.6 | Cosmo 2.1.1 | SW |
| **4** | | Netscape 4.6 | Cosmo 2.1.1 | OpenGL |
| **5** | | MSIE 5.0 | Cosmo 2.1.1 | SW |
| **6** | | MSIE 5.0 | Cosmo 2.1.1 | OpenGL |
| | Athlon 900Mhz/256MB + GeForce 2 GTS 64 MB + Win NT | | | |
| **7** | | MSIE 5.5 | WorldView 2.1 | Direct3D |
| **8** | | Netscape 4.7 | Cortona 3 | SW |
| **9** | | Netscape 4.7 | Cortona 3 | OpenGL |
| **10** | | Netscape 4.7 | Cosmo 2.1.1 | SW |
| **11** | | Netscape 4.7 | Cosmo 2.1.1 | OpenGL |
| **12** | | MSIE 5.5 | Cosmo 2.1.1 | SW |
| **13** | | MSIE 5.5 | Cosmo 2.1.1 | OpenGL |
| **14** | | Netscape 4.7 | Contact 4.4 | Direct3D |
| **15** | | Netscape 4.7 | Contact 4.4 | OpenGL |

**Table 2.** Speed (in fps) in polygon tests

| Configuration | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 000 faces | 17 | 45 | 18 | 32 | 19 | 100 | 47 | 31 | 32 | 36 | 71 | 38 | 100 | 50 | 100 |
| 2 000 faces | 15 | 45 | 17 | 23 | 17 | 100 | 46 | 30 | 31 | 34 | 67 | 36 | 100 | 50 | 100 |
| 5 000 faces | 12 | 30 | 14 | 18 | 14 | 55 | 36 | 22 | 25 | 28 | 61 | 29 | 100 | 33 | 100 |
| 10 000 faces | 8,6 | 16 | 10 | 13 | 11 | 29 | 29 | 16 | 19 | 22 | 52 | 22 | 100 | 33 | 100 |
| 20 000 faces | 5,9 | 8,8 | 7 | 9 | 7,1 | 15 | 25 | 11 | 14 | 15 | 40 | 16 | 77 | 23 | 50 |
| 30 000 faces | 4,5 | 6,1 | 5,3 | 7 | 5,4 | 10 | 21 | 8,2 | 10 | 12 | 33 | 12 | 50 | 19 | 50 |
| 40 000 faces | 3,6 | 4,6 | 4,3 | 5,4 | 4,3 | 7,6 | 19 | 6,7 | 7,9 | 10 | 27 | 10 | 40 | 17 | 33 |
| 50 000 faces | 3 | 3,7 | 3,7 | 4,6 | 3,7 | 6,1 | 16 | 5,6 | 6,9 | 8,4 | 24 | 8,5 | 33 | 14 | 33 |

**Table 3.** Speed (in fps) in tests of materials lighted by 8 light sources. Dash marks are used in the places where scenes were rendered incorrectly

| Configuration | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Diffuse | 9,5 | 16 | 8,6 | 11 | 8,9 | 15 | 31 | 17 | 19 | 20 | 54 | 20 | 100 | 33 | 50 |
| Emissive | 9,5 | 16 | 8,6 | 10 | 8,9 | 15 | 32 | 18 | 19 | 20 | 54 | 20 | 100 | 33 | 50 |
| Transparency | 8,3 | 16 | 5,6 | 10 | 5,8 | 15 | - | 16 | 19 | 13 | 53 | 14 | 100 | - | - |
| Specular - low shininess | 9,3 | 16 | 8,1 | 8 | 8,4 | 12 | - | 17 | 19 | 19 | 52 | 19 | 100 | 33 | 50 |
| Specular - high shininess | 9,3 | 16 | 9 | 9,7 | 9,3 | 16 | - | 17 | 19 | 20 | 53 | 21 | 100 | 33 | 50 |
| All properties | 8,1 | 16 | 5,8 | 9,5 | 6 | 16 | 32 | 16 | 19 | 14 | 50 | 14 | 100 | 33 | - |

**Table 4.** Speed (in fps) in tests of light sources. Configurations No. 7 and 14 gave completely wrong results. Almost all configurations with OpenGL was able to properly process maximally 8 light sources and they ignored additional lights. Correct results for 12 light sources were obtained by Cortona browser only

| Configuration | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Headlight | 8,1 | 16 | 9 | 14 | 9,2 | 28 | - | 15 | 19 | 19 | 55 | 19 | 100 | - | 100 |
| 4 dir. lights | 8,1 | 16 | 8,3 | 11 | 8,6 | 19 | - | 15 | 19 | 18 | 54 | 19 | 100 | - | 100 |
| 8 dir. lights | 8 | 16 | 7,4 | 8,7 | 7,7 | 14 | - | 15 | 19 | 17 | 53 | 17 | 100 | - | 50 |
| 12 dir. lights | 7,9 | 14 | 7,4 | 8,5 | 7,7 | 14 | - | 14 | 18 | 17 | 53 | 17 | 100 | - | 50 |
| 4 point lights | 7 | 13 | 5,5 | 8,8 | 5,7 | 11 | - | 14 | 16 | 13 | 53 | 13 | 100 | - | 100 |
| 8 point lights | 6,4 | 11 | 4 | 6 | 4,1 | 7,2 | - | 12 | 16 | 10 | 53 | 10 | 100 | - | 50 |
| 12 point lights | 5,9 | 10 | 4 | 6 | 4,1 | 7,2 | - | 12 | 14 | 10 | 52 | 10 | 100 | - | 50 |
| 4 spot lights | 7 | 12 | 5,1 | 8 | 5,3 | 11 | - | 14 | 16 | 12 | 52 | 13 | 100 | - | 100 |
| 8 spot lights | 6,3 | 11 | 3,6 | 5,5 | 3,7 | 6,7 | - | 12 | 15 | 9,2 | 51 | 9,3 | 52 | - | 50 |
| 12 spot lights | 5,7 | 9 | 3,6 | 5,5 | 3,7 | 6,7 | - | 12 | 14 | 9,2 | 51 | 9,3 | 50 | - | 50 |

**Table 5.** Speed (in fps) in texture tests

| Configuration | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **1 object** | | | | | | | | | | | | | | | |
| without texture | 8,2 | 16 | 9,8 | 17 | 10 | 30 | 26 | 17 | 19 | 21 | 56 | 22 | 100 | 30 | 100 |
| texture 1x1 | 4,6 | 12 | 4,2 | 10 | 4,3 | 14 | 21 | 9,1 | 16 | 9,6 | 38 | 9,7 | 67 | 25 | 100 |
| texture 512x512 | 3,9 | 11 | 4,2 | 10 | 4,3 | 14 | 20 | 7,7 | 16 | 9,6 | 37 | 9,7 | 67 | 25 | 100 |
| texture 2048x2048 | 2,8 | 7,5 | 4,1 | 9,7 | 4,2 | 14 | 20 | 3,9 | 16 | 9,3 | 35 | 9,5 | 67 | 20 | 100 |
| texture 4096x4096 | 2,5 | 7,5 | 4,1 | 9,5 | 4,2 | 14 | 23 | 2,7 | 16 | 9,2 | 34 | 9,5 | 67 | 20 | 100 |
| **1024 objects** | | | | | | | | | | | | | | | |
| without textures | 6,5 | 9,8 | 5,8 | 10 | 5,9 | 14 | 12 | 11 | 13 | 14 | 33 | 14 | 55 | 14 | 33 |
| 1 texture (DEF/USE) | 3,5 | 6,4 | 1,8 | 5,8 | 1,8 | 6,7 | 8,9 | 6,6 | 10 | 5,3 | 24 | 5,4 | 33 | 13 | 31 |
| 512 textures | 2,6 | 1,6 | 1,6 | 5,2 | 1,6 | 6,9 | 6,7 | 4,5 | 9,3 | 4,4 | 20 | 4,3 | 31 | 7,8 | 25 |
| 1024 textures | 2,5 | 0,7 | 1,6 | 0,7 | 1,6 | 0,7 | 7,1 | 4,2 | 8,5 | 4,4 | 18 | 4,3 | 25 | 9 | 17 |
| 1 texture (no DEF/USE) | 3,5 | 6,4 | 1,6 | 0,7 | 1,6 | 0,7 | 9,7 | 6,5 | 9,7 | 4,4 | 18 | 4,4 | 25 | 13 | 25 |

**Table 6.** Memory requirements (in Mbytes)

| Configuration | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 50 000 faces | 22 | 34 | 12 | 32 | 17 | 31 | 34 | 22 | 7 | 13 | 4 | 1 | 23 | 7 | 6 |
| 1024 objects without textures | 22 | 31 | 12 | 32 | 18 | 33 | 16 | 3 | 10 | 17 | 5 | 1 | 6 | 1 | 5 |
| 1024 objects + 512 textures | 58 | 53 | 39 | 52 | 47 | 60 | 58 | 37 | 57 | 42 | 44 | 31 | 48 | 14 | 31 |
| 1024 objects + 1 texture (no DEF/USE) | 23 | 32 | 67 | 69 | 71 | 77 | 21 | 17 | 34 | 68 | 81 | 53 | 117 | 8 | 6 |

Finally, Table 6 shows memory requirements for certain configurations. VRML scenes used for memory allocation tests are taken from Table 2 and 5.

Presented results are only a subset of executed measurements. Others are available on our VRML Benchmark pages [5], where people can also upload their results, thus the content of the pages will be gradually updated.

## 4. Discussion

Although our intention when designing a new benchmark set was to test the overall graphics performance, several specific questions can appear, for instance:

1. How big is a difference between software and hardware rendering? (What is more important – a speed of CPU or a quality of a graphic card?)
2. What is the best VRML browser?
3. Is there any difference between OpenGL and Direct3D?
4. What is the bottleneck in VRML scenes for nowadays computers?
5. Etc.

To seriously answer all such questions, an extensive set of measurements should be performed and a deep analysis should be done. Due to the space limitations in this paper, we will concentrate only on some of issues mentioned above. Our main interest was the comparison of different VRML browsers using hardware-accelerated graphics. Thus the following sections will discuss results from configurations No. 7 (WorldView), No. 9 (Cortona), No. 11, 13 (CosmoPlayer), No. 14, and 15 (Contact).

(see the last group of columns in Fig. 2). This behavior has a practical reason – a WWW server could provide various images on the same *url*. Web cameras are proper examples of that. On the other hand, images used for benchmarking have been placed on local disk and not distributed by WWW server.

Taking the three "normal" test scenes into a consideration, it is clear that Contact VRML browser consumed the lowest amount of memory, both in version utilizing Direct3D (No. 14) and OpenGL (No. 15). The second in a row is CosmoPlayer embedded in MSIE (No. 11) and the third one is Cortona (No. 9). It is surprising that CosmoPlayer behaves differently in Netscape and MSIE. It always consumed more memory when running within Netscape environment.

When caring about the amount of memory required by VRML browser, we have also made an observation that rendering of several *successive* VRML scenes caused gradual and extremely high grow of allocated memory, sometimes up to all physical/logical memory available. Since we were not able to figure out whether this strange de-allocation strategy is intentional kind of caching or simply wrong memory management, we did not take it into a consideration when designing our

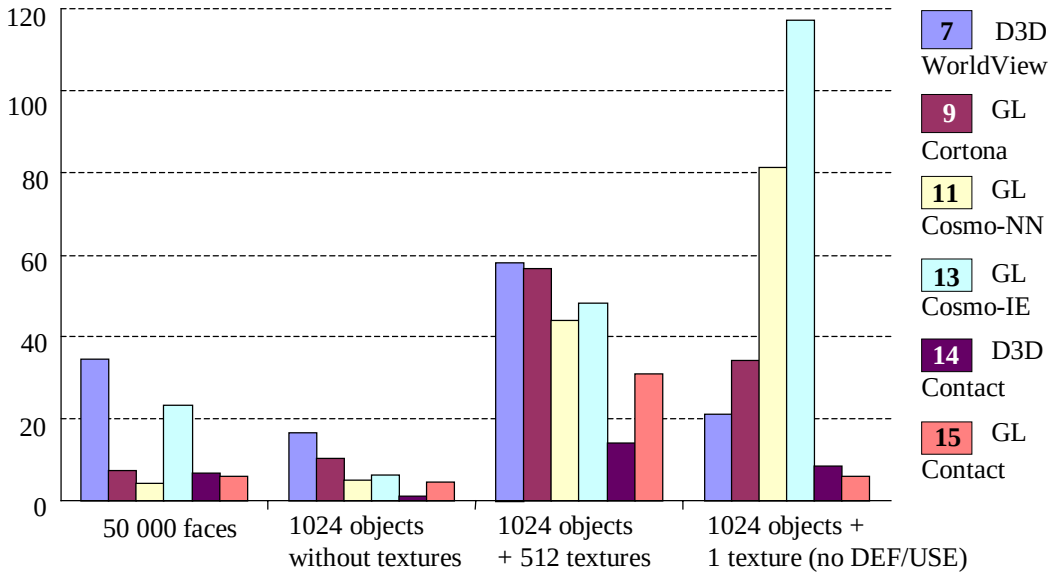



**Fig. 2.** Comparison of memory requirements (in MBytes) for different VRML browsers

### 4.1 Memory Requirements

We have found big differences in memory requirements. The amount of allocated memory varied in almost 50% when comparing the greediest browser (WorldView) and the most modest one (blaxxun Contact) – see Fig. 2. Let us also highlight the fact that CosmoPlayer was probably the only one browser that allocates incredibly big memory for our "tricky" scene with 1024 identical textures with the same url, but without any USE statement benchmarks. This undesirable behavior has been observed, when CosmoPlayer and Cortona browsers utilized hardware accelerated OpenGL library. A memory was managed well when software renderers were used.

### 4.2 Speed

The most important measurements have dealt with the speed (see Fig. 3). We have compared best configurations for each VRML browser. The

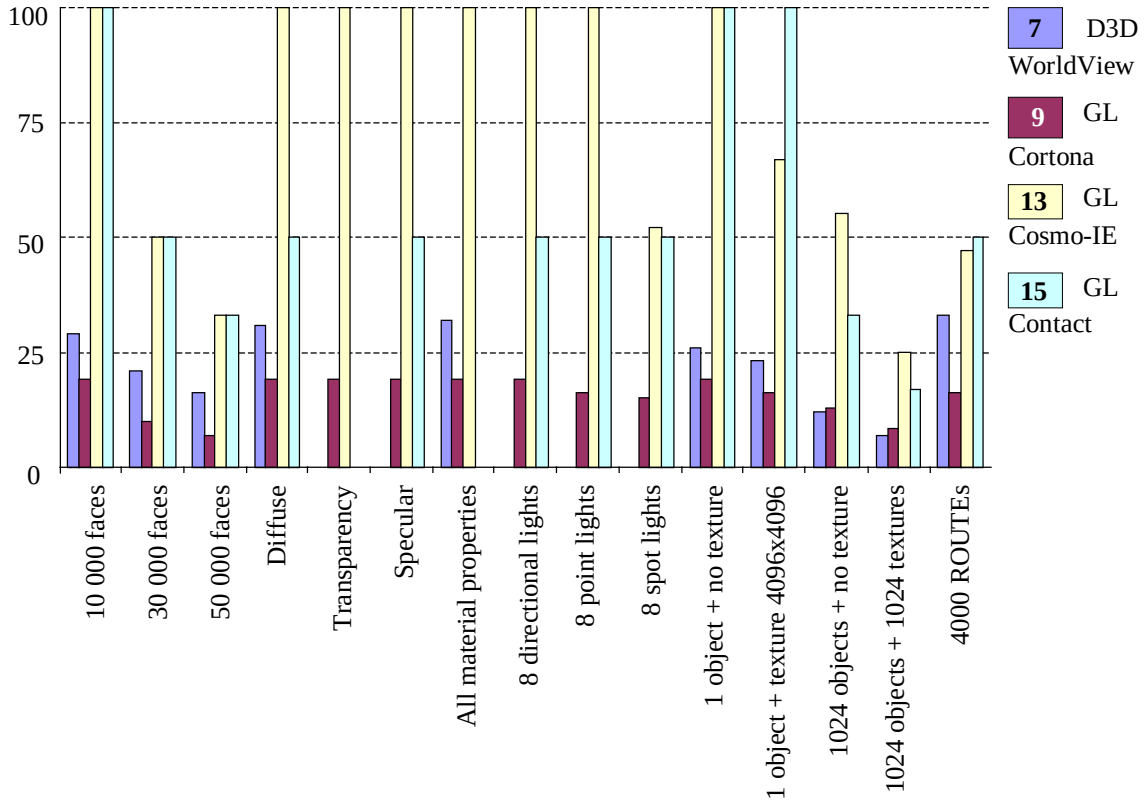



**Fig. 3.** Comparison of speed (in fps) for selected tests and four different VRML browsers

OpenGL hardware renderer was the best choice in all cases. The only exception was the WorldView browser that supports Direct 3D only.

Instead of comparing all measured values, only the most important ones are presented in the graph in Fig. 3. Representative speeds taken from tests of polygons, materials, lights, textures and routes give a good overview of performance achieved by tested browsers.

Two specific levels of the speed are of high importance. Speed 25 fps is commonly accepted as a low limit for smooth animations (movie quality). Speed 50 fps is required when resulted images are presented with stereoscopic devices, where different images for each eye must be rendered. Taking these two speed levels as the main criteria for the performance of VRML browsers, we can be satisfied with CosmoPlayer and blaxxun Contact. They rendered almost all scenes with the speed higher than or equal to 50 fps.

The other two VRML browsers gave markedly worse results. They were not able to achieve the speed of 50 fps in key tests. The speed of 25 fps was also unattainable task in many cases. Although WorldView seems to be a little bit better than Cortona, it is actually worse, because it presented incorrect results in light and material tests, while Cortona was always perfect in these particular issues.

### 4.3 Other Observations and Final Evaluation of VRML Browsers

When analyzing all speed results (not only those shown in Fig. 3), we have noticed several phenomena that were common for all tested browsers:

- Different Material parameters influence the rendering speed minimally.
- Processing PointLights and SpotLights is more complicated comparing with DirectionalLights. This is caused by additional computations of attenuation.
- OpenGL implementations often do not process more than 8 light sources. Many VRML browsers ignore this limitation and present wrong result, although switching to software evaluation of lighting could probably solve this. This problem is actually VRML Conformance issue, but it should be mentioned here too.
- Variances in size and number of textures do not influence the speed very much. Of course, all scenes with textures were always rendered much slowly than without textures.
- Direct3D libraries are not only slower than OpenGL, but they also have serious problems with transparent 3D objects.
- Event processing does not influence the speed too much. Big cascade consisting of 4.000 events decreased the speed only by 20% in average. This high number of events is quite rare in practical VRML scenes.

**Table 7.** VRML browsers arranged from the best to the worst

| No. | Browser | Pros | Cons |
|---|---|---|---|
| **1** | CosmoPlayer | Best speed with OpenGL | High memory requirements<br>Max. 8 lights rendered correctly |
| **2** | Contact | Very fast with OpenGL | Problems with transparent objects<br>Max. 8 lights rendered correctly |
| **3** | Cortona | Perfect processing of lights | Speed is not big enough |
| **4** | WorldView | --- | Slow and inaccurate with lights and materials |

These observations show that light sources have a big impact on rendering, both in terms of speed and correctness.
Table 7 summarizes our opinion of tested VRML browsers.

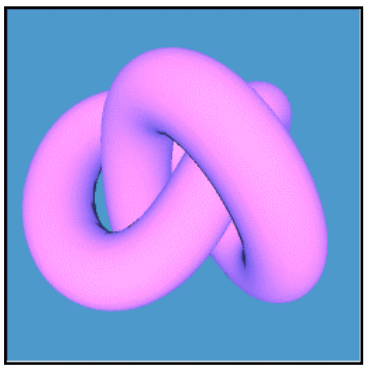
## 5. Conclusion and Future Work

A new set of graphics benchmarks has been introduced in this paper. It exploits the fact that VRML browsers perform complex tasks when presenting large, lighted, textured and animated 3D scenes. All benchmarks are publicly available on the web [5] as well as results measured on four well-known VRML browsers.
We are going to extend benchmarks for testing VRML browsers implemented as pure Java applets, i.e. browsers that do not require installation as a plug-in to a WWW browser. A transition to latest EAI specification is another task for the near future.
Finally, we are going to implement automatic receiving and updating new measurements sent us by independent testers. The first step in this effort is an on-line comparison of user's measurements with already performed (reference) benchmarks on our server. Visualization of such graphs is done in 3D using VRML as shown on color images 9 and 10. We would like to receive also measurements performed on other than MS Windows operating systems, e.g. Linux and Apple MacIntosh OS.
The authors believe that newly developed graphics benchmarks will help users to objectively check a graphics performance of their computers. Developers of VRML browsers are invited to use these benchmarks for evaluation of their products. Any other feedback is highly welcome.

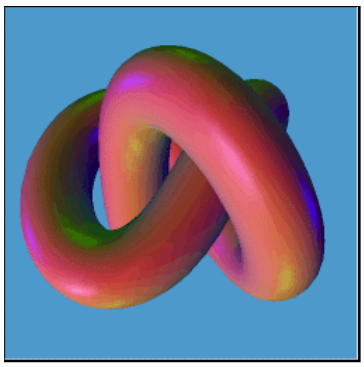
## References


1. The Virtual Reality Modeling Language External Authoring Interface. *Committee Draft* ISO/IEC 14772-2, http://www.vrml.org/WorkingGroups/vrml-eai/Specification/
2. Lynne Rosenthal, Mark Skall, Mary Brady, Michael Kass, Carmelo Montanez-Rivera: *Web-Based Conformance Testing for VRML*, Standard View, Volume 5, Number 3, September 1997, Association for Computing Machinery Inc.
3. VRML Conformance Working Group, http://xsun.sdct.itl.nist.gov/~mkass/vrml-conf/
4. SPECopc Group Develops Benchmarks for Systems Based on OpenGL API, http://www.spec.org/gpc/opc.static/overview.htm
5. VRML Browser Benchmark Pages, http://www.cgg.cvut.cz /VRML/Benchmarks/ http://www.cgg.cvut.cz ~zara/bench/
6. blaxxun Contact VRML browser download page, http://www.blaxxun.com/download/
7. Cortona VRML browser download page, http://www.parallelgraphics.com/downloads/
8. CosmoPlayer VRML browser download page, http://www.cai.com/cosmo/
9. WorldView VRML browser download page, http://www.cai.com/cosmo/

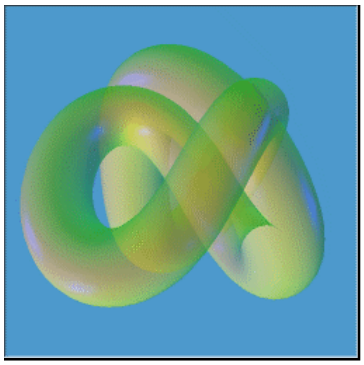
## Appendix - Color Images



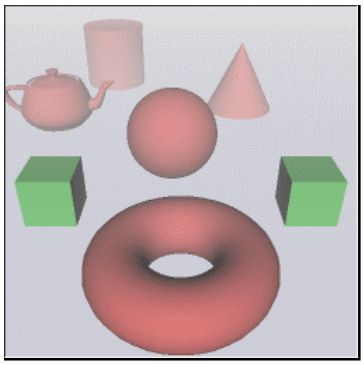
**C.1.** A model for test of emissive color



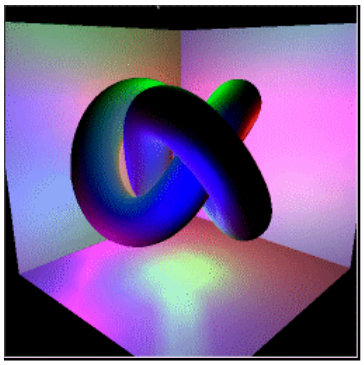
**C.2.** A model for test of specular reflection



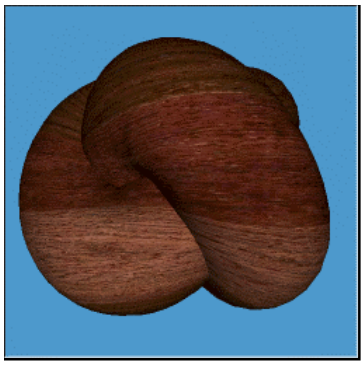
**C.3.** A model for test of transparency



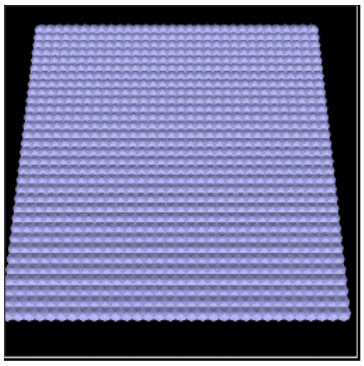
**C.4.** A scene for fog test



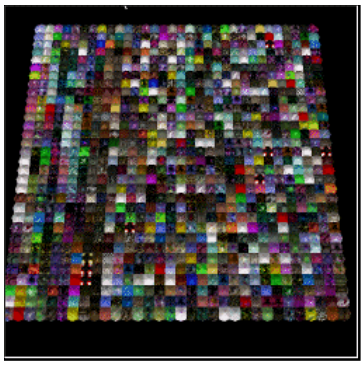
**C.5.** A scene for test of point lights



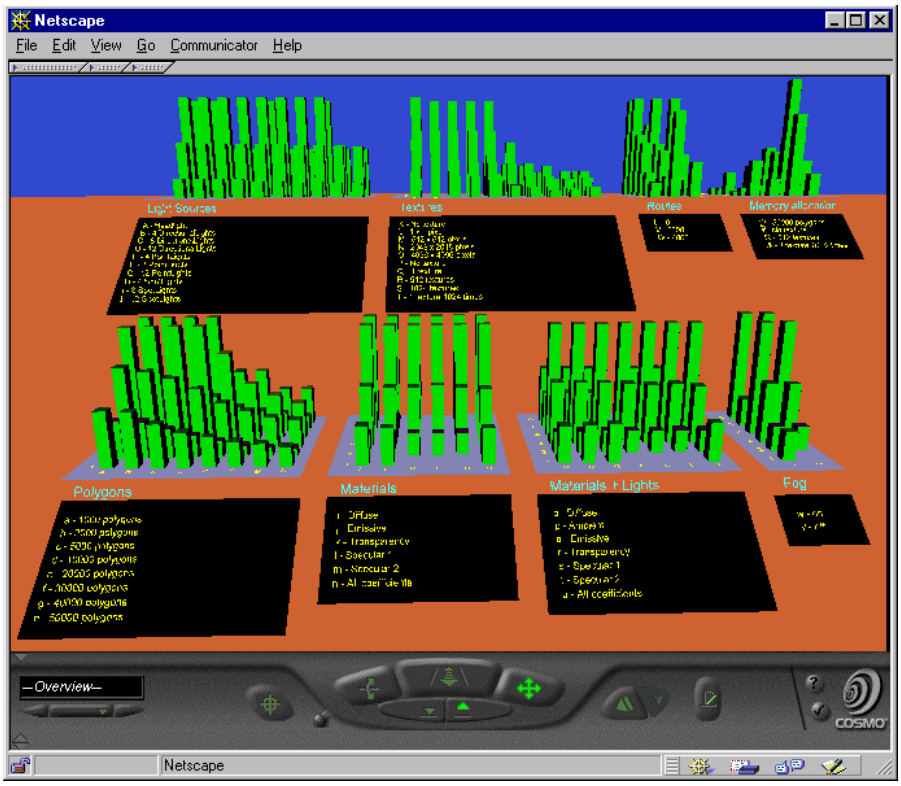
**C.6.** A model for large texture size test



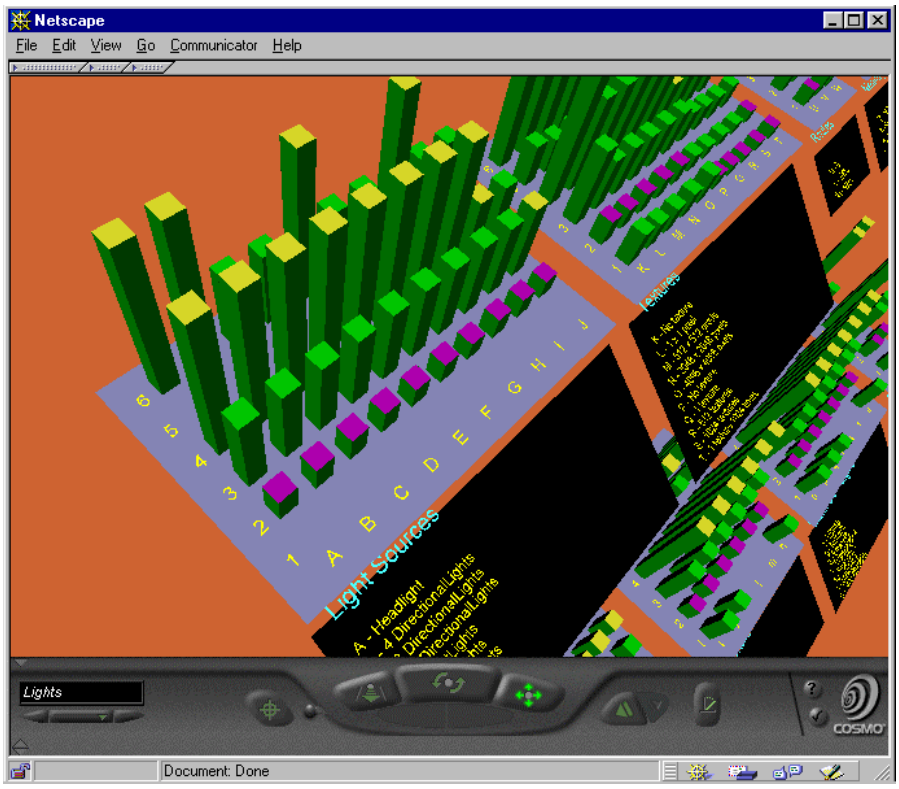
**C.7.** A scene with one texture on 1024 objects



**C.8.** 1024 textures on 1024 objects




**C.9.** Experimental visualization of results of VRML benchmarks – global view




**C.10.** Detailed view on graph representing results of light tests. Best values for each test are highlighted by yellow top on green columns, while worst values have pink tops.