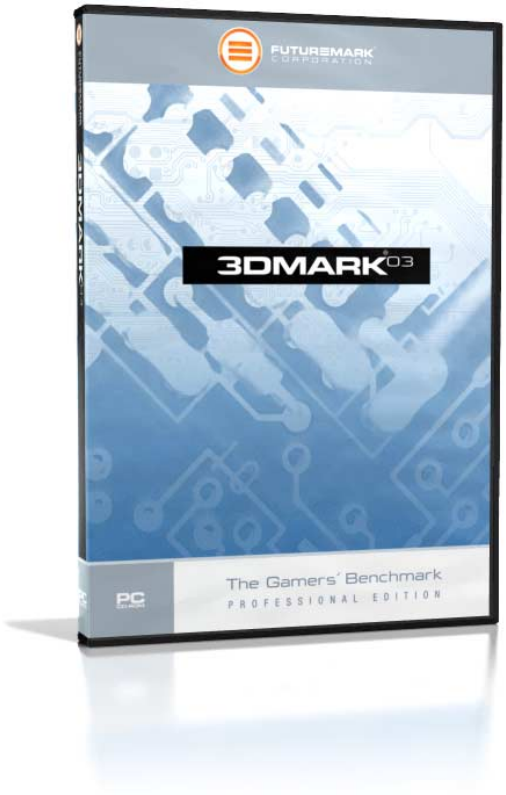
FUTUREMARK CORPORATION

WHITE PAPER

3DMARK®03

Next Generation 3D Benchmarking



Maneesh Dhagat
Futuremark Corporation

12930 Saratoga Avenue,
Suite B-2, Saratoga, CA 95070 USA

February 11, 2003

Table of Contents

Overview

This paper introduces 3DMark®03, the latest in the 3DMark benchmark series built by Futuremark® Corporation (formerly known as MadOnion.com). The name 3DMark has become a standard in 3D graphics benchmarking; it has grown to be used by virtually all on-line and paper publications. It has proven itself as an impartial and highly accurate tool for benchmarking 3D graphics performance of the latest PC hardware. 3DMark has a very large following worldwide among individual PC owners. More than 5 million benchmark results have been submitted to Futuremark's Online ResultBrowser database. It has become a point of great prestige to be the holder of the highest 3DMark score. A compelling, easy-to-use interface has made 3DMark very popular among game enthusiasts.

Futuremark's latest benchmark, 3DMark03, continues this tradition by providing a state-of-the-art Microsoft® DirectX® 9 benchmark.

Figure 1: 3DMark03 is Futuremark's DirectX 9 Benchmark

The introduction of DirectX 9 and new hardware shader technologies put a lot of power in the hands of game developers. Increasingly realistic 3D games will be available over the next year and a half. The use of 3D graphics will become more accessible to other applications areas and even operating systems. In this new environment, 3DMark03 will serve as a dependable tool for benchmarking 3D graphics.

In this paper we discuss our approach to 3D graphics benchmarking and the development of 3DMark03. We cover our multi-step development process that is key to developing a dependable benchmark. The technologies used in each of the benchmark tests are described in detail. We also discuss our much improved image quality tools. We present our score calculation approach and the formulae used to generate the results. Lastly, the Online ResultBrowser, our popular online comparison tool, is described.

3D Graphics Benchmarking

3D graphics benchmarking allows users to accurately evaluate the performance of the newest 3D graphics software technology on the latest 3D hardware. 3D hardware currently comes in two forms – consoles and PCs. The PC market experiences more upgrades and revisions yearly by far than consoles. PCs also come in a tremendous variety. Each PC component – motherboard, CPU, system memory, graphics card, etc. – has multiple possible manufacturers, making 3D graphics benchmarking very complex. The 3DMark series makes measuring 3D graphics performance simple.

3D game benchmarking, besides measuring 3D graphics performance, also evaluates 3D sound, real-time physics overhead, etc. In 3DMark03, we have also added several new features in the area of 3D game benchmarking.

3D graphics continues to be influenced by many factors, including the graphics card and CPU industries, 3D development platforms, and 3D gaming trends.

Graphics Cards and CPUs. Graphics chipsets and cards are the primary influencers for 3D graphics performance. 3D games also need the CPU for real-time physics, artificial intelligence (AI), visibility optimization, as well as many other things. Over the years, more and more functionality has been transferred to the graphics card. Starting with the hardware rasterizer in the early 1990s, the hardware transformation and lighting functionality in the late 1990s, and finally hardware vertex shading in 2001, many tasks are now offloaded from the CPU to the graphics hardware giving better 3D, as well as overall PC performance. We expect this trend to continue. Thus, 3DMark03results are very dependent on the graphics card and scale less based on the CPU speed. For this reason we have added a CPU test to allow the user to measure CPU performance for 3D graphics usage.

3D Platforms. The dominant 3D platform on PCs is Microsoft DirectX®. 3DMark releases have coincided with major DirectX versions - 3DMark®2000 with DirectX® 7, 3DMark®2001 with DirectX® 8, and 3DMark®2001 SE with DirectX® 8.1. Accordingly, 3DMark03 targets DirectX® 9 features and continues the tradition being a forward-looking benchmark. We hope to give the user a view into state-of-the-art 3D graphics not only today, but also up to one and half years into the future. OpenGL is another popular 3D platform, especially in CAD and scientific applications. As Direct3D® has higher usage in games and more uniform driver support, we only support Direct3D. However, we actively monitor the progress of OpenGL adoption.

3D Gaming. 3D games currently represent the primary use of 3D graphics. Various 3D game genres have naturally evolved as result of historically popular games such as Wolfenstein3D by id Software. The table below presents a high-level categorization of 3D game genres.

Table 1: 3D Game Genres at a High-Level

Game Genre	***Description***
Simulation	Recreates real-life situations requiring execution of specific activities.
Action (First Person Shooter)	First person view of player in a virtual world; confronts enemies without running out of energy.
Adventure (Role Playing Game)	Assuming a role, player follows a plot, interacting with environment and other players.
Strategy	Player assumes god-like position to control a simulated world; goal can be expansion of territory, accumulation of things, etc.

Although the boundaries between genres are often blurry, we can successfully use them to design the workload for a benchmark. For 3DMark03, we have chosen simulations, action, and adventure games to represent a typical workload. It is not our intention, nor is it possible, to include all game genres. Instead the goal is to include a good representative sample to represent 3D game usage.

There have been several efforts in 3D graphics benchmarking. Tools like 3D WinBench 2000, an earlier benchmark, evaluated DirectX 7.0 feature by feature. On the other end of the spectrum, games like Quake3 and Unreal Tournament have benchmark utilities that measure frame-rates of their demo scenarios. These utilities give the user a good idea of that specific game's performance on their PC. With 3DMark it is not our intention to evaluate every DirectX feature, but to measure typical 3D graphics usage. Nor is it our goal to measure the technology embedded in a particular game; our goal is to benchmark 3D graphics usage in general. Specific games are often optimized for certain hardware; their workload can be significantly different on different hardware. In 3DMark, it has been a central goal to keep the workload the same on all hardware. For these reasons, 3DMark remains the neutral standard in 3D graphics benchmarking. Over 30 million copies of 3DMark are in hands of users. Ninety-five percent of online publications use 3DMark in their reviews of display cards[*]; almost all print media also regularly utilize 3DMark. Over 2 million users a month visit our website to share their benchmark results. We have collected over 5 million benchmark results in our databases.

Our Development Methodology

Futuremark approaches all of the benchmarks it creates with a standard development methodology. We believe that the process we follow is central to the development of a successful and dependable benchmark.

The key part of the development process is cooperation with all major manufacturers. For 3DMark, we have found that groups with the most interest in this process are: graphics chipset makers, CPU manufacturers, and PC manufacturers. These companies are willing to cooperate with us because they share the vision that strong, objective benchmarks are in everyone's interest. We have been running a formalized beta program that allows these vendors to participate in designing leading benchmarking standards in the PC industry. The cornerstones of our design process are *transparency* and *neutrality*. We make a strong effort to document all processes that make up the benchmark; we continuously strive to make these documents better. Also, we always try to maintain the highest standards of neutrality, neither favoring nor ignoring any party.

The figure below depicts a high-level view of our benchmark development methodology.

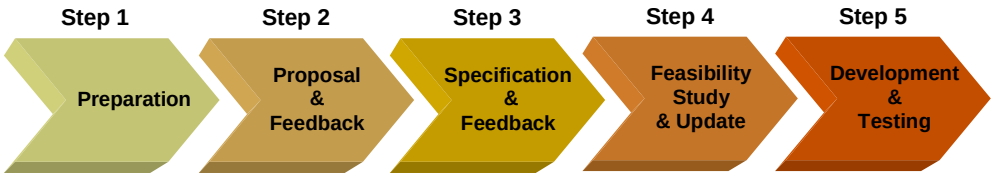


Figure 2: Benchmark Development Methodology

Step One. In this first step, we use a variety of resources to enumerate a first set of possible features for the benchmark. We draw upon our own experience with previous benchmarks and the feedback we have subsequently received. Communications with the gamer and game developer communities also provide valuable input. The beta member group is another source of insights. From these, we develop high-level ideas of features we may include in the benchmark. At this early stage, features and implementation options are intentionally kept open. For example, for 3DMark03, we had decided to implement dynamic shadows. The implementation options

[*] Of 104 reviews covered, 99 reviews used 3DMark. *MadOnion Benchmark Media Usage Report – Q1 2002.*

– rendering to textures, using stencil buffers, using z-shadow buffer, etc. – were documented, but a specific method was not chosen.

Step Two. The document produced in the previous step is a proposal. It is designed to present features and implementation options in a format amenable for getting constructive feedback. This proposal is circulated to our beta members. The feedback received aids us in choosing features and implementation methods.

Step Three. In the third step we create a written benchmark specification. Each of the workload tests is specified in detail with exact versions of the technologies used. For example, we may detail the action occurring in a first person shooter scenario using DirectX 8 hardware, 1.1 vertex shaders, and 1.1 pixel shaders. The specification is circulated to the beta members. The feedback is analyzed and incorporated at our discretion.

Step Four. In the next step we implement prototype code to see if the available technology will support our plans. We may discover the certain tests are not possible, or may be surprised to discover that more can be achieved. The results are incorporated into the specification and again circulated.

Step Five. The fifth step consists of implementing the workloads or tests. Periodic releases are made to the beta members and their feedback may again be taken into account.

3DMark03 Overview

Benchmark Structure. 3DMark03 is a collection of 3D tests. These include a set of four *game tests*; these are the only tests used to calculate the overall 3DMark03 score. The benchmark also includes a set of *CPU*, *feature*, *image* quality, and *sound* tests. Each of these tests measures specific 3D-related functionality, but their result is not included in the overall score. They do not fall into the target usage, but are included to allow the user to evaluate these features. The CPU test is a convenient way to measure the performance of the CPU for typical 3D usage. The feature tests isolate the performance of key 3D features primarily relating to shader technologies. The next set of tests is an exciting new addition to 3DMark: the 3D sound tests. These evaluate the impact of 3D sound sources on 3D graphics performance. The software also includes a set of much-improved image quality tools. These provide a powerful way to ensure integrity of the graphics hardware and drivers.

Real-Time Rendering. Each 3DMark03 game test is a real-time rendering of a 3D scenario. It is important to note that these renderings are not merely animations or a set of recorded events; they are designed to function like 3D games work. As with 3D games, all computations are performed in real time. This is a critical part of our philosophy of 3D graphics benchmarking.

DirectX. All tests have been compiled and linked with DirectX 9.0 libraries. However, only a small part of the benchmark requires DirectX 9 hardware. As DirectX 8 hardware is now commonly available, a major portion of tests requires DirectX 8 hardware support. A small part of the benchmark can be run with only DirectX 7 compatible hardware. Note that 3DMark03 is not appropriate for solely evaluating DirectX 7. 3DMark2001 SE should be used to measure DirectX 7 and early generation DirectX 8 hardware.

Frame-based Rendering. For the first time, 3DMark provides two different real-time rendering mechanisms: *time-based* rendering and *frame-based* rendering. Each game scene has a timeline or natural pace of action. Time-based rendering adjusts the frame-rate to maintain this timeline. For simple scenes, the hardware may be able keep up a high frame-rate; for complex ones, it may need to lower the frame-rate to maintain the timeline. For very low-end hardware, the user may even see a “slide-show” effect, as the hardware struggles to keep up the natural pace of action. This is the typical mechanism used by 3D games and hence 3DMark03 scores are only generated with time-based rendering. Earlier 3DMark versions only included time-based rendering. This time we also provide a frame-based rendering mechanism that renders a fixed number of frames for each second of the timeline. The number of frames is user configurable. Frame-based rendering forces each run to

generate exactly the same number of total frames regardless of the PC used. Of course, the hardware would render these frames at the fastest pace that it can handle, so a fluctuating frame-rate is still observed. Frame-based rendering is a great mechanism for comparing performance of two PCs running exactly the same workload.

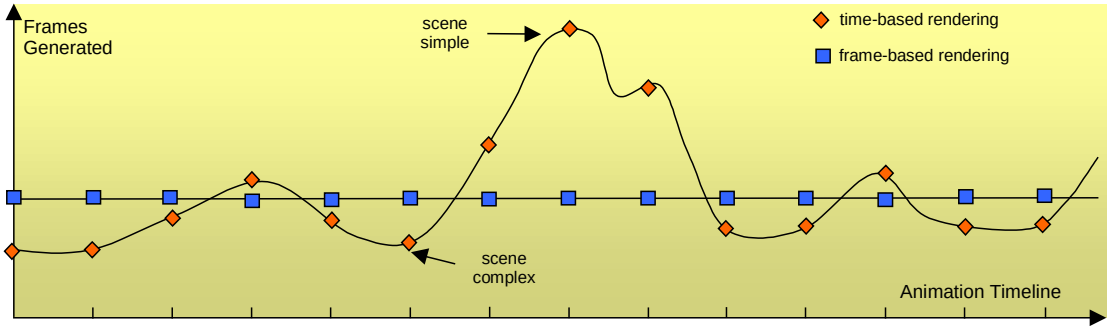


Figure 3: Time-based versus Frame-based Rendering

Shader Technologies. We believe future 3D products will move to using *vertex shaders* and *pixel shaders*. 3DMark03 focuses on these. Both are programmatic models that allow the 3D graphics developer to bypass fixed vertex and pixel processing in the Direct3D rendering pipeline. The developer can execute small programs directly on the graphics hardware. With this freedom, the developer can generate the stunning graphical effects found in 3DMark03.

Vertex shaders allow you to bypass the fixed transformation and lighting stage of the Direct3D pipeline. Vertex shader programs can be executed directly on the graphics hardware, if present, or emulated in software. Pixel shaders, introduced with DirectX 8, replaced the DirectX® 6/7 fixed multi-texture unit. Pixel shader programs, unlike their vertex counterparts, require hardware support. Vertex and pixel shaders have become an important part of 3D graphics and are accordingly featured prominently in 3DMark03. One of the game tests, the DirectX 9 showcase, uses 2.0 vertex shaders and 2.0 pixel shaders. All other games tests use 1.1 vertex shaders. The DirectX 8 game tests use 1.4 pixel shaders if available; otherwise they default to 1.1 pixel shaders.

3D Engine. Previous versions of 3DMark used the MAX-FX 3D engine. The trend in 3D engines is moving towards very lightweight DirectX wrappers as more and more work is transitioned to the API and graphics card. For 3DMark03, we have based the benchmark directly on top of the DirectX 9.0 3D platform. This allows the benchmark to be independent of specific technologies embedded in a single 3D engine implementation. We have developed a set of lightweight wrapper routines on top of DirectX to aid code re-use.

Real-time Physics. Besides 3D rendering, typical 3D games include CPU workloads. In 3DMark03, we use physics calculations to represent this workload. It is important to note that these computations are being performed in real-time to accurately represent real game usage. Natural movements such as falling characters and crashes in 3DMark03 use real-time physics. Havok's Game Dynamics SDK provides the real-time physics functionality.

Game Test 1: Wings of Fury

Axis fighter planes rush an allied bomber armada headed towards the target of its bombing mission. Allied fighter planes supporting the bombers respond swiftly, and deadly dogfights ensue. Machine gunners in the bomber gun turrets also join the fight. Fighter planes are lost on both sides. Bombers are hit by fighter gunfire and by flak from anti-aircraft guns below. They collide with one another and go down in exploding pieces. This is the scenario for the first game test.

Early in the design of 3DMark03, we knew from information in our benchmark databases that a significant portion of PCs had DirectX 7 graphics hardware. We needed one game test that could run on these mid-range PCs. For these, a lighter game test was required. We concluded that a flight simulator type scenario would be a reasonable choice, as simpler background objects would occupy the majority of the screen.

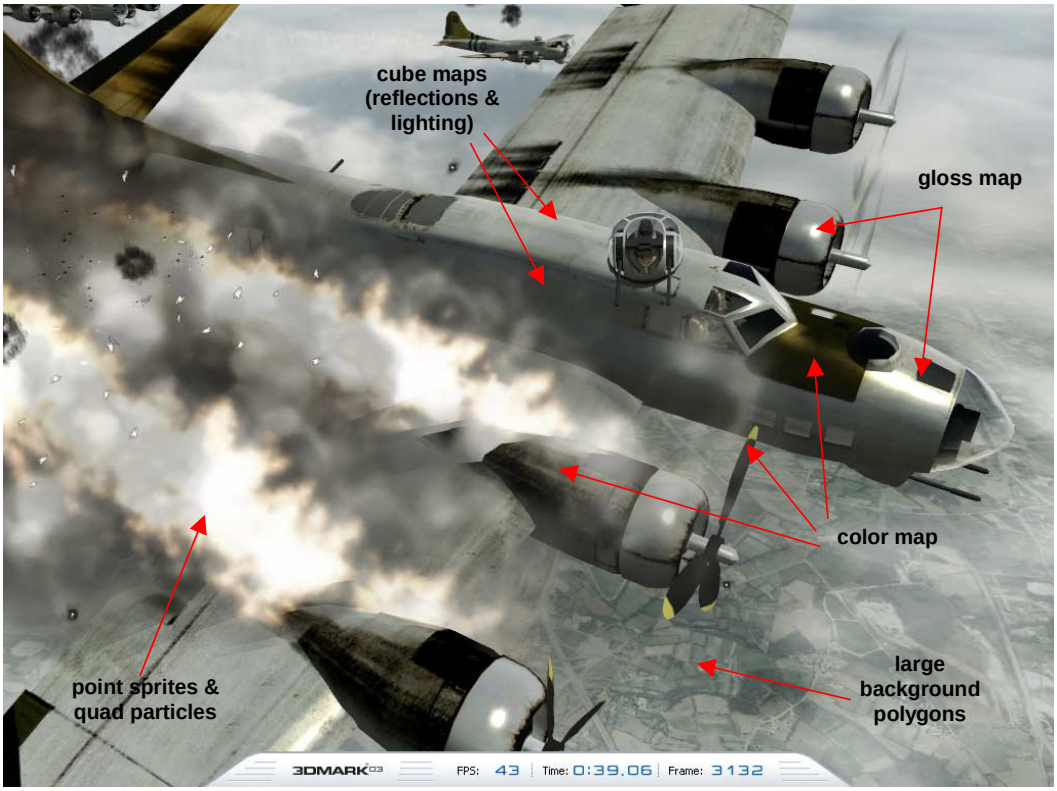


Figure 4: Bomber Falls showing features of Game Test 1

Textures. This game test requires lower fill rate, as the large background polygons are single textured. The airplanes have a higher number of polygons and can be textured more interestingly than applying just a single color map. Cube maps are used for reflections and lighting; this produces a nice per-pixel lighting effect. A color map is used for the airplane color and insignia. A gloss map is used to dictate which parts of the airplane are more reflective. Together these maps form the quad-textured material of the airplanes. Note that we used only fixed function pixel processing, but require cube map and MODULATE2X texture blending hardware support.

Particles. This game test also has heavy use of particle systems. In such systems, an object consists of a cloud of elementary particles, each of which is born, evolves and dies over time. They are used for smoke-trails, condense-trails, and explosions. DirectX hardware accelerated point sprites, a specific type of particles, are used where possible such as condense-trails of airplanes. However, point sprites have limitations. They cannot be independently scaled, rotated, or animated as they specified with only a single point. Where such functionality is needed, such as for explosions, we use traditionally rendered quad particles. These are specified with the coordinates for two triangles.

In flight simulator games today, most of the screen is taken up with gauges and controls. Through the rest of the screen, the user looks at the environment and other airplanes. In this game test we focused on this active part of the screen. We have added several cinematic camera and higher details to stress this significant part of the workload.

This test is not meant as a definitive evaluation of DirectX 7. It is not designed to give the average performance of DirectX 7 3D graphics usage. Typical DirectX 7 games use fixed vertex processing, whereas this game test uses 1.1 vertex shaders. We believe this is the future of vertex processing on both graphics cards and CPUs. The overall goal of game test 1 is to complete the collection of the four game tests as a test that can run on DirectX 7 hardware and one that requires a lower fill-rate. To fully evaluate DirectX 7 performance, the previous version of the benchmark, 3DMark2001 SE, is more appropriate.

Technical Summary. DirectX 7 graphics hardware with cube map and modulate2x support is required for this test. 32 MB of graphics memory is needed to run this test without texture uploads during the test execution. All vertex processing uses 1.1 vertex shaders. The planes have 4 textures as described above. The plane material can be rendered in a single pass on graphics hardware capable of four texture layers in a single pass. Particles use point sprites as far as possible, but due to their limitations, quad particles are used for some effects. Approximately 32,000 polygons are rendered per frame. 16MB video memory is used for textures, 6MB for vertex buffers and 1MB for index buffers.

Game Test 2: Battle of Proxycon

A space ship is under attack. The enemy is trying to breach the ship's hull using special pods. Defenders bravely take up positions to repel the intruders. They carefully move into the ships corridors. Intruders waiting behind a hydraulic door are swiftly destroyed. Soldiers fall on both sides. The final fight leads to the ship's landing bay. With a final push, the enemy is finally repelled. This is the exciting scenario of our second game test.

As DirectX 8 hardware is now commonly available, a major part of 3DMark03– game tests 2 and 3 – requires DirectX 8 hardware support. For game test 2, we chose an action or first person shooter (FPS) scenario. A FPS game test was first included in 3DMark99 and it proved to be quite popular. Since then, users have consistently requested a new FPS game test using the latest 3D graphics technology for this game genre. Often first person shooters are the games that push 3D graphics quality the furthest.

The most significant feature of this game is the use of normal mapping and stencil shadowing.

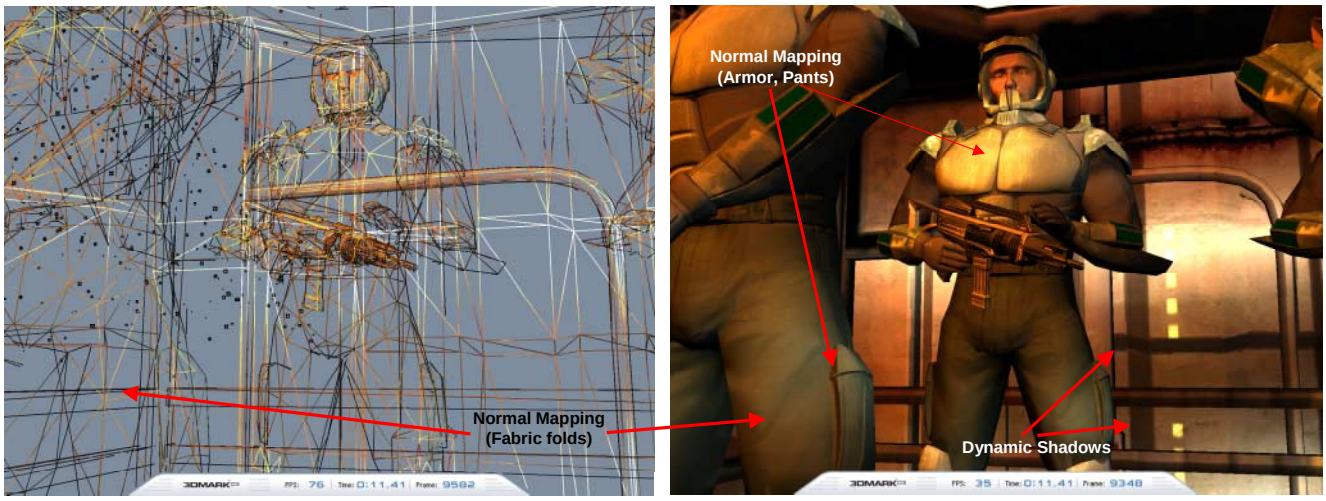


Figure 5: Normal Mapping and Dynamic Shadows in Game Test 2

Normal Mapping. Normal mapping is an advanced texturing technique that provides a great amount of geometric detail. Even though the objects in this game test have a relatively low polygon count, normal mapping adds the appearance of a great amount of geometric detail. Normal maps can themselves be generated from

high-polygon versions of the same objects; this is the technique used in this test. Normal mapping allows the complexity of objects to be low – decreasing memory requirements and improving 3D graphics performance – while still providing compelling visuals.

Dynamic Shadows. Real-time shadows for this game test are generated using the stencil buffer. Stencil shadowing is a powerful mechanism in which shadow volumes are extracted from each lit object. Every object falling into the shadow volume acquires a shadow. The stencil shadows are even self-enabling, i.e. they can cast shadows on the originating object itself. For example, a character's head, lit from a certain angle, can cast the shadow of the nose on the head itself.

We believe that this combination of normal mapping and stencil shadowing will be widely used in future 3D games.

When using this kind of stencil shadowing, the developer is left with some options on the implementation. 3DMark03 does as much work as possible in the vertex shaders, since the goal of 3DMark03 is to measure vertex and pixel shader performance in 3D games. Also, it is expected that many games with similar technology will have a heavy workload for the CPU doing physics (including collision detection), artificial intelligence and visibility optimizations for example. It is therefore desirable to perform as much as possible on the graphics card in order to offload the CPU.

An alternative implementation would be to give some of the graphics tasks to the CPU, and thereby offloading the graphics card. The skinning could be done on the CPU, which would reduce the amount of vertex shader tasks. Also, when pre-skinning on the CPU, the characters would not need to be re-skinned for each rendering pass. Then again, skinning is a fairly light vertex shader operation, and with as few characters as in this game test, there should not be much benefit. Also, if there are many characters on screen, more pre-skinned characters would need to be transferred over the AGP bus. In addition to the skinning, the shadow silhouette calculation could also be done on the CPU, which could be an optimization for low end DX8 hardware, with lower vertex shader performance.

1.4 Pixel Shaders. This game test benefits from 1.4 pixel shader hardware support. Available in DirectX 8.1 and DirectX 9 hardware, 1.4 pixel shaders allow rendering to be performed with fewer passes as compared to 1.1 pixel shaders. With 1.1 pixel shaders, most objects in this game test need one rendering pass for depth buffer initialization and three passes for each light source. The three passes are: stencil pass, light fall-off to alpha buffer, and diffuse and specular reflection using look up tables. If 1.4 pixel shader support is available, the same calculations can be completed with one depth buffer initialization pass and a single pass for each light source.

Figure 6: Post Processing Effects in Game Test 2

Post Processing. Some cameras for this test use full frame post processing effects using 1.1 pixel shaders. The two effects used are *Depth of Field* and *Bloom*. Depth of field is a cinematic effect where only objects at a certain distance from the camera are displayed sharply focused. Objects closer or further appear blurred. This effect is achieved by blending the original frame and a blurred version. Bloom is another cinematic effect that makes bright surfaces appear over-bright. Such a surface appears all white obscuring any details too small to be visible in the intense reflection. Also, light spills over to the surrounding darker areas. Bloom is achieved by additively blending blurred highlights to the original frame. Note that, post-processing effects are disabled by default for this test. They are also automatically turned off when anti-aliasing is selected.

Technical Summary. DirectX® 8 graphics hardware with 1.1 pixel shader support is required for this test. 128 MB of graphics memory is needed to run this test without texture uploads during the test execution. All vertex processing uses 1.1 vertex shaders; all pixel processing uses 1.1 pixel shaders or 1.4 pixel shaders if supported. Most objects have a color map, a normal map, and a lookup table for specular and diffuse reflection. Characters are vertex shader skinned. Dynamic shadows are generated using the stencil buffer. Havok real-time physics is used for falling characters and crashes. Approximately 250,000 polygons are rendered per frame using 1.1 pixel shaders. With 1.4 pixel shaders this number is lowered to approximately 150,000 due to reduced number of rendering passes. 80MB video memory used for textures, 6MB for vertex buffers and 1MB for index buffers.

Game Test 3: Troll's Lair

A lone adventurer wielding a powerful weapon explores an old study. She moves to a bookcase. There she discovers a secret door that leads down to a hidden cellar. Two hostile trolls reside within. At first they are oblivious to her presence, concentrating on their game. They soon notice her and fierce combat ensues. The adventurer uses her magic sword to defeat them quickly. This is the adventure or role playing game (RPG) scenario of our third game test.

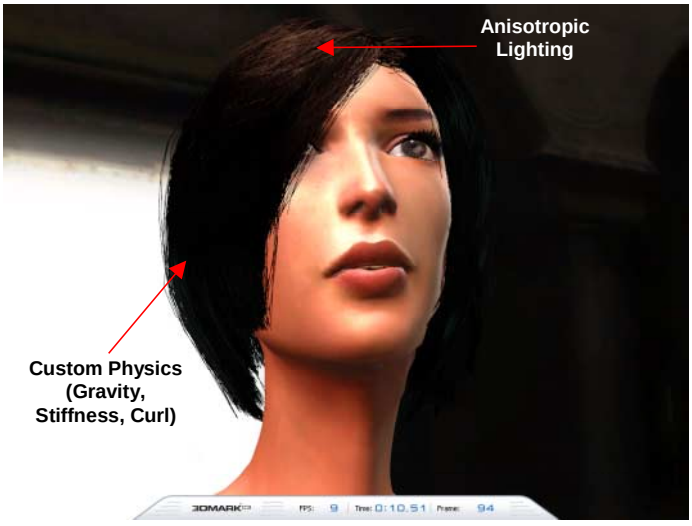


Figure 7: Custom Hair Model and Anisotropic Lighting in Game Test 3

Hair. This test has a much higher polygon count than game test 2. In fact, for the main character each hair strand is individually modeled. To render the hair we use *Anisotropic Lighting*. This technique is typically used to simulate the effects of lighting on grooved surfaces, such as some fabrics, vinyl records, brushed metal, or, as in this game test, human hair. Hair movements follow a custom physics model; three forces affect the vertices of the hairs: gravity, hair stiffness, and hair curl.

Other Effects. As in game test 2, normal mapping is used to generate the effect of higher geometric detail. Stencil shadowing is used to generate real-time shadows. If available, 1.4 pixel shaders are used to reduce the number of rendering passes. Depth of field and bloom post processing effects are used to generate more realistic looking frames. Note that, post-processing effects are disabled by default for this test. They are also automatically turned off when anti-aliasing is selected. Collapsing of the troll characters uses Havok real-time physics.

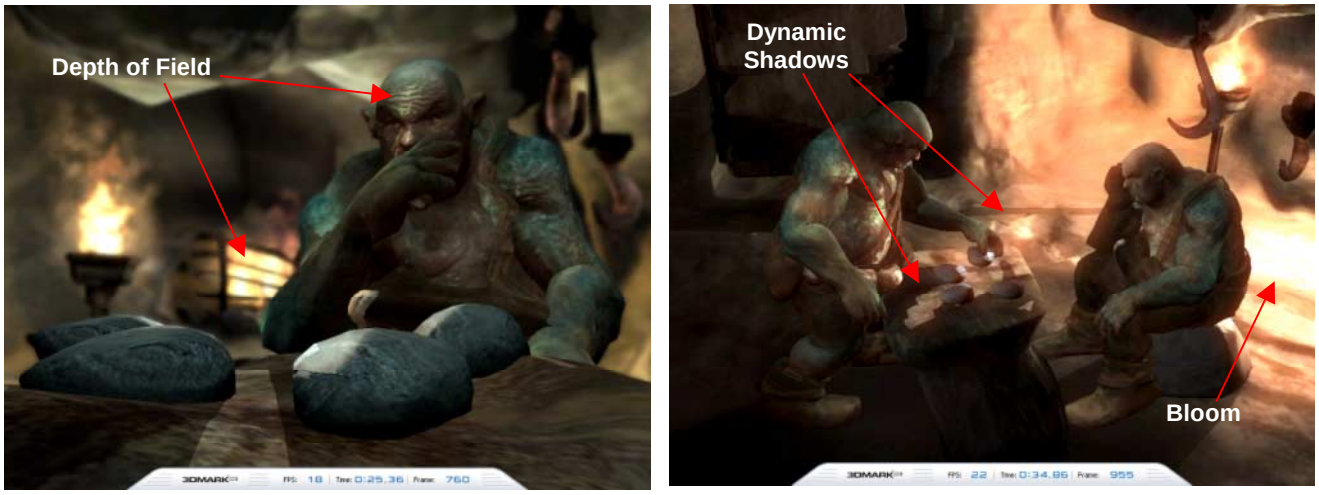


Figure 8: Other Effects in Game Test 3

Technical Summary. DirectX 8 graphics hardware with 1.1 pixel shader support is required for this test. 128 MB of graphics memory is needed to run this test without texture uploads during the test execution. All vertex processing uses 1.1 vertex shaders; all pixel processing uses 1.1 pixel shaders or 1.4 pixel shaders if supported. Most objects have a color map, a normal map, and a lookup table for specular and diffuse reflection. Characters are vertex shader skinned. Dynamic shadows are generated using the stencil buffer. Havok real-time physics is used for falling characters. A custom physics model is used for the main character's hair. Approximately 560,000 polygons are rendered per frame using 1.1 pixel shaders. With 1.4 pixel shaders this number is lowered to approximately 280,000 due to reduced number of rendering passes. 64MB video memory is used for textures, 19MB for vertex buffers and 2MB for index buffers.

Game Test 4: Mother Nature

A slowly moving stream runs gently along swaying trees and moving blades of grass. Clouds overhead cast shadows on the rocks and ground. A turtle slowly crawls up the bank. A butterfly can be seen fluttering freely. The sun shines brightly in this beautiful day scene of game test 4.

This game test is our DirectX 9 showcase. Previous 3DMark versions have included a test that displays the next-generation 3D graphics technology. This time it is the incredible photo-realistic rendering possible with 2.0 vertex and pixel shaders. A nature scene was used in 3DMark2001 to showcase DirectX 8 and it was highly appreciated. We decided to once again select this theme. This game test shows some amazing effects and provides an excellent workload for determining PC performance for highly demanding 3D applications of the future.

Leaves and Grass. The 3DMark2001 nature test included an interesting effect showing leaves swaying in the wind. These were implemented with quad objects; each object contained a number of leaves and the object waved perpendicular to the camera. For 3DMark03, each leaf is a separate object and moves independently. The same is the case with each blade of grass, each of which moves with the wind separately. Leaf movements use 2.0 vertex shaders. The leaf vertex shader uses the new "sincos" instruction, which is well suited for modeling natural movements. The movement of grass blades uses 1.1 vertex shaders.

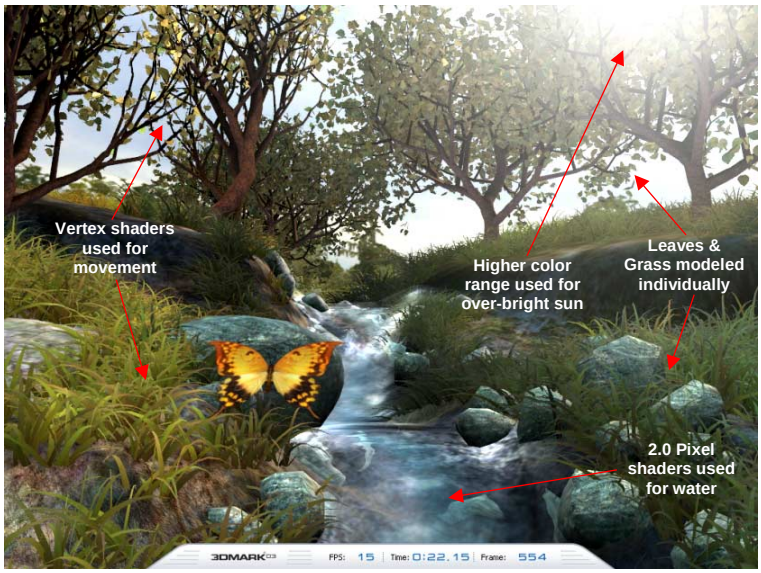


Figure 9: Properties of Nature Objects in Game Test 4

Lake. 2.0 pixel shaders are used to render the lake. Multiple texture stages are used. It uses a ripple map, two reads of a normal map, a reflection map, a refraction map, and a reflection cube map for reflection from more distant objects. It also uses a transparency map and calculates per-pixel Fresnel.

Sky and Ground. The sky uses a higher color range – a new feature in DirectX 9. This allows us to go beyond the limitation of using approximately 16 million colors. This is especially valuable for calculating over-bright lighting like the sun in this test. The ground material becomes darker as clouds pass overhead. Other parts of the ground appear over-brightly lit by the sun. This is achieved by using 1.4 pixel shaders, with a color map, a detail map, a light map, a bump map, and a normalization cube map.

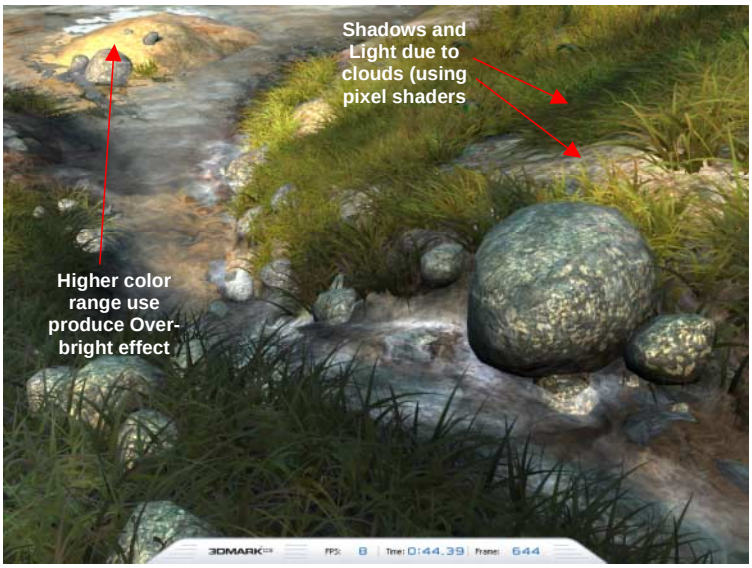


Figure 10: Other Effects in Game Test 4

Technical Summary. DirectX 9 graphics hardware with 2.0 pixel shaders is required to run this test. 2.0 vertex shaders are used for modeling the swaying of individual leaves. 2.0 pixel shaders are used for the lake surface. It uses multiple texture stages as described above. The sky uses 2.0 pixel shaders. The sun is calculated using higher dynamic color range. The ground uses 1.4 pixel shaders, with a color map, a detail map, a light map, a bump map, and a normalization cube map. It also includes a cloud layer to simulate moving cloud shadows and brighter lit areas. Approximately 780,000 polygons are rendered per frame. 50MB video memory is used for textures, 54MB for vertex buffers and 9MB for index buffers.

CPU Test

The 3DMark03CPU test allows users to evaluate CPU performance for 3D graphics workloads. It is important to note that this test is intended to only measure CPU performance for 3D graphics usage and not for general PC usage. For the latter, a benchmark such as PCMark™, available also from Futuremark, is more appropriate.

This test runs game test 1 at a resolution of 640x480 using software vertex shaders. Game test 1 is very amenable to CPU measurement as the materials used are simple and most of the screen consists of single-textured, low polygon background objects. We also run game test 3 at a resolution of 640x480 with the use of dynamic shadows disabled. The optimized code path for 1.4 pixel shaders is disabled, since that feature should not be credited in this test. Note that this test still depends somewhat on the graphics hardware, but these settings reduce the dependency.

The CPU test result can also be used as good measure of the software vertex shader speed. Vertex shading is an important part of 3D game performance. Similar calculations as vertex shading are needed in other parts of 3D game software, such as real-time physics.

Note that the CPU test by itself is not multithreaded, but multithreading in DirectX should give some benefit to the CPU score, when run on a dual CPU system or on a CPU with hyper threading support. 3DMark03 has been programmed and compiled as a 32-bit application. Therefore, 64bit support in the CPU should not improve the scores.

The CPU test reports a numerical CPU Score.

Feature Performance

3DMark03 includes a set of feature tests. These do not affect the overall score. However, their results provide valuable information for the benchmarking professional, as they often cannot be obtained from any other source. 3DMark03 feature tests measure the performance some important 3D graphics features primarily relating to vertex and pixel shader technologies.

Fill Rate

These two tests isolate the performance of fixed function pixel processing of your system – specifically the *Fill Rate*. The fill rate is the speed at which your graphics hardware is capable of drawing textures onto 3D objects. The tests report the result in million *texels* drawn per second (MTexels/s). Texels or texture elements are the pixels in the source texture. We draw a number of large surfaces covering the entire screen and apply large textures to these surfaces using fixed function pixel processing. There are two fill rate tests.

Single-Texturing. The first test measures single-texturing – how fast the graphics card is capable of drawing single textures onto 3D objects. A single texture is applied to each of 64 objects. This means that the graphics hardware must fill each of these objects separately, irrespective of how many texture layers the card is capable of drawing in a single pass.

Multi-Texturing. This test measures multi-texturing performance – how fast the graphics card is capable of drawing multiple textures onto 3D objects. Multiple textures are applied to an object in a single pass until a total 64 textures have been applied. This test benefits from the use of modern graphics cards, which are capable of applying several textures in a single pass. For example, if a graphics card is capable of applying 6 texture layers in a single pass, 6 textures will be applied to an object in 10 passes and the remaining 4 textures will be applied in the 11th pass. Fixed pixel texturing has an upper limit of 8 textures per pass. So, even modern DirectX 9 hardware, capable of up to applying 16 textures per pass, will not be able to apply more than 8.

Vertex Shader

This test isolates the vertex shader performance of your system – specifically how fast characters can be skinned. In the previous version of 3DMark, the vertex shader test had a lobby scene with multiple gunmen. For this version, a set of angry trolls from game test 3 run around in a plaza attempting to club each other.

This time we are able to handle more complex characters. There are 30 trolls in total; each is made up of approximately 5500 triangles. The trolls are rendered in four passes with 1.1 pixel shaders. They use a color map, a diffuse cube map, and diffuse and specular bump maps. Using 1.1 vertex shaders, each troll is also skinned 4 times. Hence approximately 30*5500*4 = 660,000 triangles are skinned in each frame.

The result of this test is given in frames per second. If 1.1 pixel shaders are not supported, this test is skipped.

Figure 11: Vertex Shader Feature Test

Pixel Shader 2.0

This test measures the performance 2.0 pixel shaders on your system – specifically by doing *procedural texturing*. The developer can use procedural techniques to generate high detail texturing at runtime. This means that fewer high-resolution textures need to be delivered with the application, and stored in the system and/or graphics memory at runtime. Procedural texturing can be used for generating textures that reflect the natural patterns found in solids such at wood or marble. It can also be used for generating height maps for large landscapes. As these textures are calculated at runtime, they have theoretically an infinite resolution; the camera can be moved closer and closer, while the object maintains texture quality. Only the calculation precision of the graphics hardware is a limit for the resulting texture resolution.

This test contains three 3D objects – an elephant, a rhinoceros, and the platform they stand on. A marble texture is applied to the elephant and the rhinoceros, and a wooden texture is applied to the platform. The camera moves around them to show the texture effects.

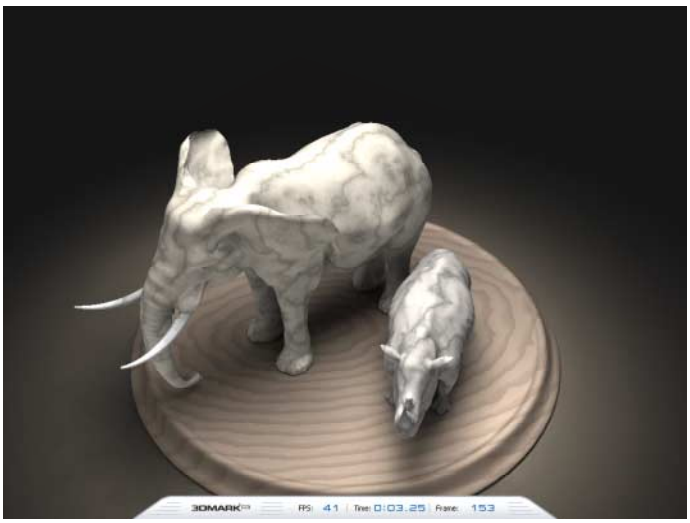


Figure 12: Pixel Shader 2.0 Feature Test

Marble. The marble pattern applied on the elephant and the rhinoceros is produced using a 3D turbulence texture map. This is generated during loading of the test. Pixel shaders are used to sample this marble map at runtime and then apply the mathematical function below to adjust the final color of each pixel. Note the use of the 2.0 pixel shader "SinCos" function:

Color = *lerp*(DarkMarble,
LightMarble,
sin((*turbulence*(Position + Frequency)+ Position.x * Scale) * PI)

Wood. The platform is texture mapped using a wood texture effect. A 3D noise map is generated during loading of the test. Pixel shaders are used to sample the noise map twice at runtime to produce the turbulence values. These are employed in the mathematical function shown below to adjust the pixel color. This generates the tree rings on the platform that vary in both frequency and width.

Color = *smoothpulsetrain*(LightWood,
DarkWood,
turbulence(RingOffset))

The result of this test is given in frames per second. If 2.0 pixel shaders are not supported, this test is skipped.

Ragtroll

The *Ragtroll* test evaluates the ability of the hardware to balance CPU and graphics workloads. It combines two typical 3D game-related workloads into a single test: CPU workload of real-time rag doll physics and a graphics workload of vertex processing. The test demonstrates the advantages of using hardware accelerated vertex shaders while the CPU is loaded with other 3D game-related tasks. It is interesting to switch between software and hardware vertex processing for this test. Hardware vertex shading can give dramatic increases in the frame-rate. The test scales both with more powerful graphics hardware and with more powerful CPUs.

In this test, several trolls take swan dives from an altitude. As they plummet to the ground, 1.1 vertex shaders are being used to skin them. The movements of their limbs, as they fall, collapse, or tumble down a staircase, are being modeled with Havok real-time physics. The troll material, as in the previous tests, uses four rendering passes using 1.1 pixel shaders. The lighting model is the same as in game tests 2 and 3 and benefits from 1.4 pixel shaders.

Figure 13: Ragtroll Feature Test

The result of this test is given as frames per second. If 1.1 pixel shaders are not supported, this test is skipped.

Sound Test

3DMark03 features an exciting new addition – a set of 3D sound tests. Besides being an interesting demonstration of the use of 3D sounds, it's the first test to allow the user to evaluate the performance impact of using 3D sound sources.

The scenario is similar to the air combat scene used in game test 1. A static chase camera follows a fighter plane as it moves through a bomber squadron. This scenario is particularly well suited for demonstrating moving sound sources with the *Doppler effect*, where the noise pitch of an airplane engine changes as it rapidly passes by. Each sound source accurately reflects its position in the three-dimensional world. Sound also experiences *attenuation*, where the intensity of distant explosions is affected by scattering and absorption.

Using the FMOD sound library by Firelight Technologies, sound sources are attached to every airplane, gun, hit, and explosion. The test is first run without any sound sources; it is then run with 24 and 60 sound sources for frame-rate comparison. Above all, this test is meant to show the performance impact of having a large number of 3D sound sources on the PC. The user will most likely observe a frame-rate drop in going from no sound sources to 24 or 60 sound sources. A sound card optimized for 3D game usage can reduce the frame-rate decrease by offloading the CPU of sound related tasks.

The test result is shown in frames per second. A sound test run will be skipped if the hardware cannot support the required amount of sound sources.

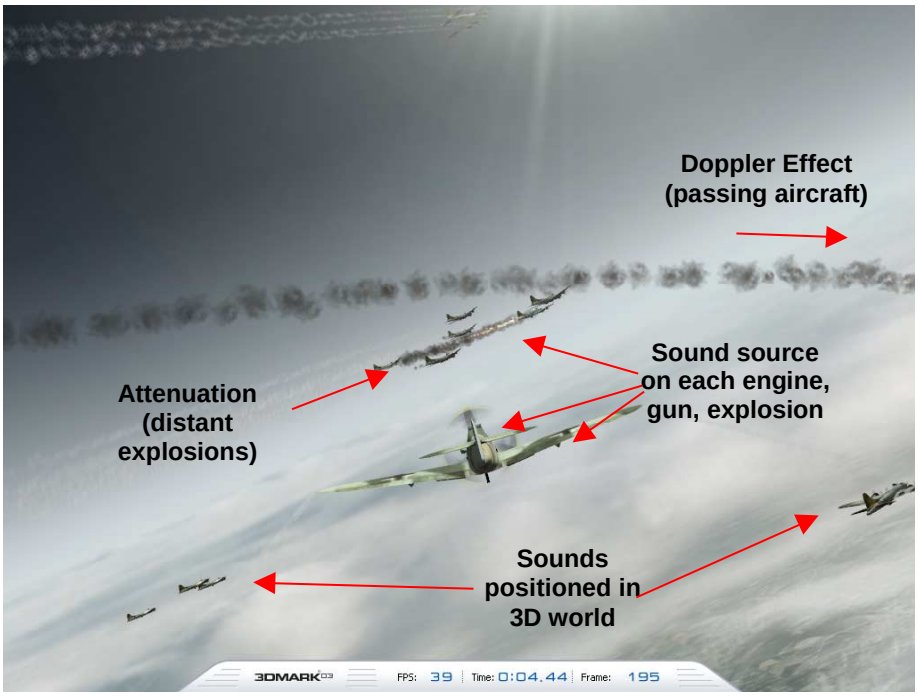


Figure 14: Effects Experienced in the 3D Sound Test

Image Quality Tools

3DMark03 includes an improved set of image quality tools for the professional tester. There are two tools included: *Image Quality* tool and *Texture Filtering* tool.

Image Quality Tool. There was an image quality tool included with the previous 3DMark. This time we have made it much more powerful. This new version allows the user to choose any set of frames from any of the tests for quality inspection. The selected frames are rendered using the graphics hardware as well as using the DirectX reference rasterizer. Alternatively the same set of frames can be rendered on two different PC systems. The user's favorite imaging software can then be used to visually compare the frames for rendering quality.

Various options are available to allow fine-grain control of this tool. A resolution can, of course, be selected. The user can choose the use of a post processing mode (for depth of field and bloom effects) or various anti-aliasing modes for pixel processing. Several Texture Filtering options are available: anisotropic, bi-linear, tri-linear, or an optimal mixture of bi- and tri-linear determined by the software. The user can force the use of software vertex shaders.

The user selects the first and last frame to be rendered. When generating the frames, the tool uses frame-based rendering at 30 frames per second. For frames to be rendered using the DirectX reference rasterizer, the DirectX®9 SDK must be installed on the PC.

This tool and the frame-based rendering option mentioned earlier combine to form an excellent mechanism to ensure integrity of the graphics hardware and drivers.

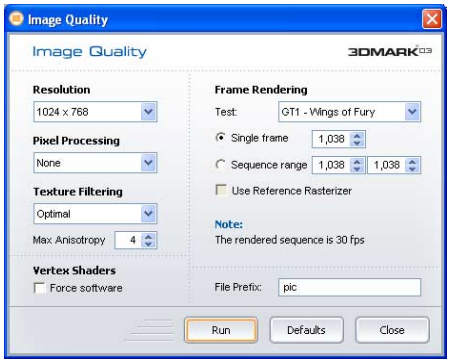


Figure 15: Image Quality Tool

Texture Filtering Tool. This is a synthetic image quality tool for visually verifying texture filtering quality. The tool highlights for the user any imperfections or artifacts that might occur on particular hardware.

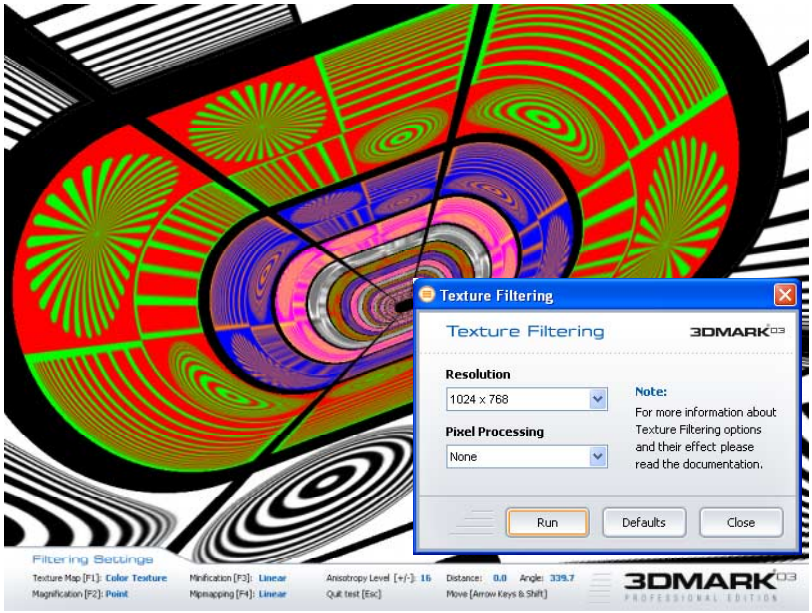


Figure 16: Texture Filtering Tool

The camera can be controlled using the keyboard for better image inspection; it can be moved back and forth as well as rotated. Keyboard options also allow the user to switch between color, and black and white textures. Filters to handle Magnification and Minification artifacts can also be chosen; nearest point, linear, and anisotropic are available. The Mipmapping approach used can also be chosen from: none, point or linear. Lastly, a resolution and an anti-aliasing mode can be specified as options in this tool.

The filtering modes typical for 3D applications are:

- Bilinear filtering - linear magnification, linear minification, point mipmapping.
- Trilinear filtering - linear magnification, linear minification, linear mipmapping.
- Bilinear Anisotropic filtering - anisotropic magnification, anisotropic minification, point mipmapping.
- Trilinear Anisotropic filtering - anisotropic magnification, anisotropic minification, linear mipmapping.

3DMark03 Demo

The 3DMark demo has always been an important part of the product. We have always used it as an opportunity to share our excitement about the latest 3D graphics technology. The goal of the demo has always been to simply have some fun and to showcase some very compelling scenes. Getting feedback on the demo has always been exciting; it is not uncommon to see the 3DMark demo running in loop mode at tradeshow exhibits.

The 3DMark03 demo contains all four game tests, including a significantly longer version of fighter combat scene of game test 1. All tests include the use of rich sounds. All post-processing effects are now enabled.
Note that depth of field and bloom effects were disabled by default in the game tests. Additional pixel shader effects can also be seen. The “scratchy film” post processing effect simulates the natural wear on old movie reels. A cross fade effect is used in some places to transition between frames. If the graphics card supports the higher dynamic color range of DX9, the end credits background will show a beautiful HDR sunset. Sit back and enjoy!

Score Calculation

As with previous versions, 3DMark03 scores will initially range between a 1000 and 5000 3DMarks. They are scaled such that an entry-level system will score approximately 1000 and a high-end system, at the time of 3DMark03 product release, will score approximately 5000. An entry-level system is approximately one with a DirectX 8 graphics card and a CPU corresponding to 1 GHz. A high-end system roughly has a DirectX 9 graphics card and a CPU corresponding to 3 GHz. Of course, with time, the high scores are expected to keep increasing. For example, for 3DMark2001 the best score immediately after the launch was close 5000, and today’s high scores are over 20,000 3DMarks.

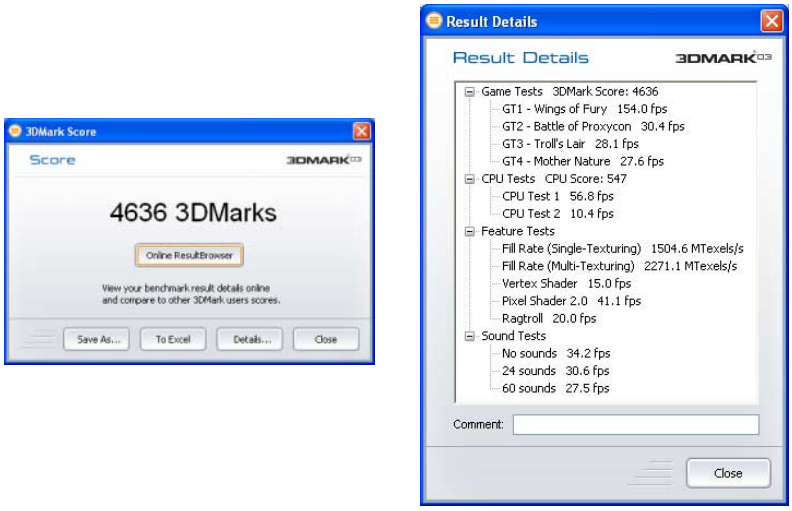


Figure 17: Example 3DMark03 Scores

Each of four game tests generates a frame-rate (in frames per second) that is used to calculate the overall score. The formula for calculating the overall 3DMark03 score is:

3DMark03 score = (Game Test 1 frame-rate x 7.3) +
(Game Test 2 frame-rate x 37) +
(Game Test 3 frame-rate x 47.1) +
(Game Test 4 frame-rate x 38.7)

This formula has been obtained by running the benchmark on a number of high-end systems. The results for each system are weighted such that game test 4 gives 20% of the total score and game tests 1, 2 and 3 are equally weighted to give the remaining 80%. Hence, game test 1, 2 and 3 each give 26.6% of the total score. The weights are averaged across all the systems used to generate the weights in the formula. Note that this weighting can be very different on low-end systems.

In 3DMark03, we have separated the measuring of CPU performance into a separate test and score. This score has a different range than the 3DMark score, in order to prevent confusion between these two total scores. It is balanced to give a CPU Score of 500 to high-end systems at launch time. The formula for calculating the CPU score is:

CPU score = (CPU Test 1 frame-rate x 4.6) +
(CPU Test 2 frame-rate x 27.5)

This formula is obtained using the same approach, with the two game tests weighted so that each contributes 50% to the final score.

The table below shows sample scores on three PC configurations. Two systems have the same graphics card, and two have the same CPU. Note that the CPU score rises significantly for the more powerful CPU. The overall 3DMark score is less affected by the CPU, but depends more on the graphics card.

Table 2: 3DMark Score and CPU Score scaling

System	CPU	AMD® Athlon™ XP 1700+	AMD® Athlon™ XP 2700+	AMD® Athlon™ XP 2700+
	Internal Clock	1469 MHz	2170 MHz	2170 MHz
	External Clock	133 MHz	166 MHz	166 MHz
	Graphics Card	ATI® Radeon™ 9700 Pro	ATI® Radeon™ 9700 Pro	ATI® Radeon™ 9500 Pro
3DMark Score		**4423**	**4577**	**3486**
CPU Score		**424**	**517**	**515**

Online ResultBrowser

The utility of a benchmark result can be rather limited in isolation. Without a mechanism to compare your system to others, it is difficult to say if your system is a high-end state-of-art PC, a mid-end system, or even a low-end beige box. To allow benchmark users to come together to compare and analyze results, we provide a web service called the Online ResultBrowser or ORB.

The ORB has become Futuremark's most popular online service. It provides the users a web application to manage and compare benchmark results. The ORB database contains over 5 million results. After running the benchmark, the user can choose to *upload* the results to the ORB. We enforce data privacy, so no one except the user will be able to see the individual results. Futuremark also verifies the uploaded results for accuracy. The ORB helps the user by adding meaning to the benchmark data; it allows the user to compare the results with those from other PCs. The user is able evaluate the PC's relative performance - determine the PC's weaknesses and its strengths.

Of course, users can decide to share their results by explicitly *publishing* them. This allows them to show their results to the rest of the world. For many users, their 3DMark performance results are a point of pride. For some, their position in our top performing PC rankings is the source of a bit of fame. Many proud PC owners include ORB URL links to their published benchmark result in their email signatures.

The ORB is the user's virtual benchmark laboratory. Users can experiment with different system configurations such as increasing the RAM or upgrading the CPU. They can get the new PC performance by searching through results published by others. Users can also maintain their own performance track record by submitting multiple projects to the ORB. They can assemble custom multi-compare sets to compare their PC to multiple other configurations.

As the ORB is an online service, Futuremark continues to improve it by adding new functionality. The benchmark data collected by Futuremark is used for generating statistics and recommendations for the user community. This means that every result submitted helps all users to select reliable hardware upgrades. The ORB aids in making buying decisions; before spending money, the user can validate expectations of different hardware options.

System Information

Associated with each benchmark result is the complete profile of the target PC; we call this the *System Information*. The system information not only provides detailed configuration information (CPU speed, RAM, graphics chipset, etc.) to go along with the performance data, but also gives the system state (open applications, AGP mode, free system memory, etc.). The complete system information consists of over 300 fields. These include:

- **CPU** Information – clock speeds, internal and external caches
- **Direct X** Information – display drivers, direct draw/show attributes, texture formats, 3D capabilities
- **Memory** Information – memory arrays and modules
- **Motherboard** Information – bios, card slots, system devices
- **Monitor** Information – monitor attributes
- **Power Supply** Information – batteries
- **Operating System** Information – version, installed service packs
- **Open Processes** Information – applications, processes
- **Logical Drives** Information – local and network logical drives
- **Hard Disk** Information – disk drive attributes

Note that no private information is ever collected.

To ORB uses the system information to enable the search and compare functionality. System information also allows Futuremark to verify the accuracy of published benchmark results. These increasing numbers of system information records are used to provide information back to the users in the form of lists of most popular and powerful hardware components shown on our web site. We also use system information data to build other tools such as the Futuremark Performance Analyzer – an online tool used for product comparisons.

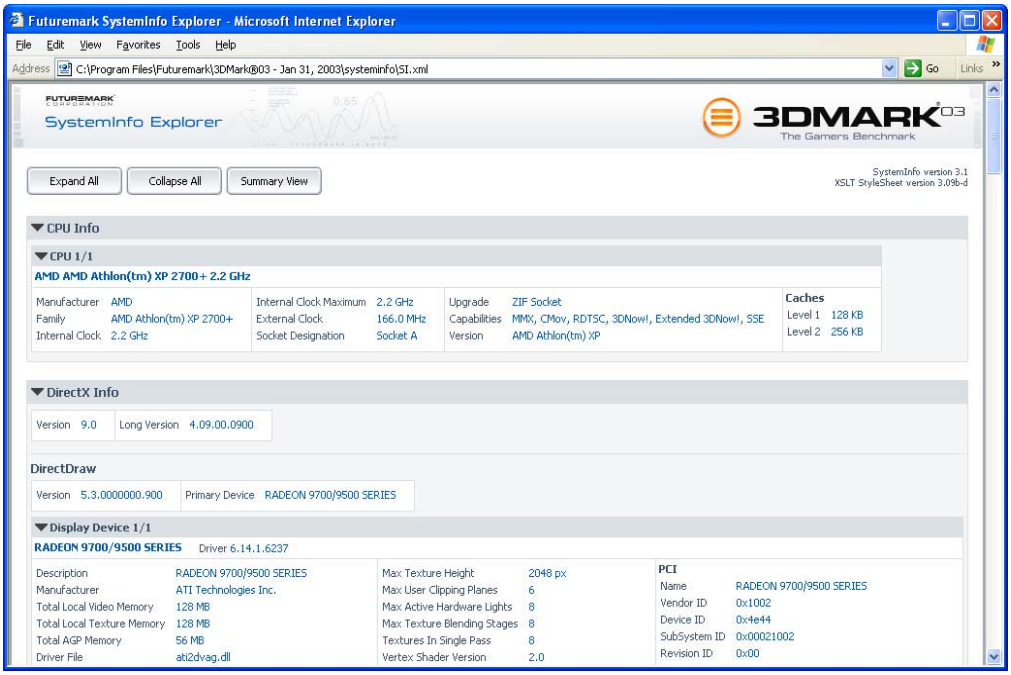


Figure 18: Example 3DMark03 System Information

Conclusion

Over the last four years, 3DMark has become a trusted standard in 3D graphics benchmarking. Futuremark's latest version, 3DMark03, continues the tradition of providing an easy-to-use tool for benchmarking next-generation 3D graphics technology. This time, powered by the latest DirectX 9 features and supporting graphics hardware, we showcase 3D graphics likely to be seen over the next year and a half. More importantly we provide a neutral and transparent benchmarking tool to allow the user to evaluate the capabilities of the latest hardware. We have added new functionality, such as the 3D sound test, a set of new feature tests and much-improved image quality tools, to allow the user to isolate performance of key parts of 3D graphics usage. As graphics hardware becomes more powerful, especially with new sophisticated shader support, stunning 3D graphics will become accessible to more applications. We believe 3DMark03 will serve as a highly dependable tool for the benchmarking professional in this new environment.