Note on 3D Graphics Technology

A piece of paper is two-dimensional. Historically, one used a camera to create an accurate two-dimensional representation of a real three-dimensional object or scene. To create an image of an imaginary three-dimensional object or scene required an artist or draftsman. The challenge of 3D computer graphics was to convert a mathematical representation of a 3D object into a real image, or picture, of that object. If this could be done, then a person could use a computer to take a virtual journey in a foreign city, a proposed building, a complex machine, or an imaginary adventure.

The most straightforward approach to 3D computer graphics was ray tracing. Given a mathematical representation of a 3D scene, including a complete specification of the sources of illumination, a computer could examine the paths of thousands, or millions, of light rays as they moved from the light sources, passed through, reflected off, and diffracted around objects, and passed into the viewing plane. Such techniques had been used to construct stunning images, but required many hours of computer time to create a single image.

Real-time 3D graphics required that motion also be captured, not just a static image. As objects moved, or as the user acted to change the point of view, the image need to change in real time. Smooth flicker-free movement required the creation of 30-60 frames per second, giving 17-35 milliseconds for the construction of an image, a far cry from the hours needed for ray tracing.

3D computer graphics technology was a set of techniques, or "tricks," for quickly building a convincing image of a 3D scene. A mixture of science and art, it drew on analytic geometry, the study of how light reflected off various surfaces, perceptual psychology, and computer algorithms. David Kirk, NVIDIA's Chief Scientist, explained:

> 3D computer graphics is difficult because it requires simultaneous excellence in a number of different disciplines. There are Ph.D. dissertations being written in how skin reflects light. To make use of that knowledge, someone has to take some very deep math, some empirical results, some knowledge of computer processing, and mix them to create algorithms. I would guess that there are fewer than 200 people in the world who could program our Dawn demonstration.

Intellectual Roots

The intellectual roots of modern computer graphics can be traced to Ivan Sutherland's work in the early 1960s. A student at MIT, Sutherland created a program (Sketchpad) that allowed the user to draw on a computer screen with a light pen. Sketchpad introduced the idea of interactive computer graphics, pop-up menus, objects (e.g., circles, rectangles), and the ability of deform and transform objects. Sutherland's film depicting the operation of Sketchpad stimulated a number of other researchers to focus on computer graphics.

Evans & Sutherland

In 1967, Sutherland joined David Evans at the University of Utah department of computer science. Together, they formed a firm (Evans & Sutherland) that developed image display technologies enabling flight simulators and computer-aided industrial design.

This note was prepared by Professor Richard P. Rumelt with the research assistance of Gregory Rose, UCLA Anderson M.B.A., 2004. This note was prepared using public sources.

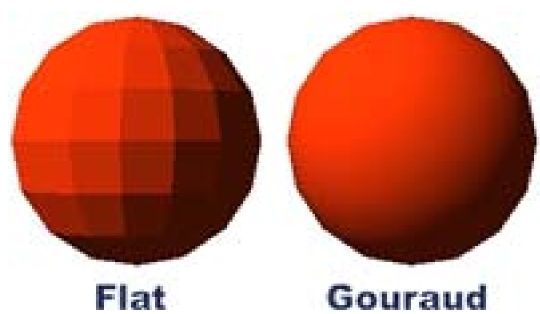
NASA Flight Simulation

To construct a flight simulator, one could not simply supply an image on a screen. To provide a realistic simulation, the image had to change instantly and realistically in response to the (simulated) position of the viewer. It was soon determined that this could only be accomplished by (1) creating mathematical descriptions of the three-dimensional objects to be viewed, (2) creating a mathematical method for projecting an image of these objects onto a (mathematical) flat surface, much as a camera projects an image of the world onto film, (3) continually updating this projection as the positions of objects and/or the viewer change, and (4) converting this projected image into a screen display. This general approach to creating images became known as 3D graphics. The final images were normal 2D displays but the logic used to create these images required the construction and analysis of a 3D world.

The activities at the University of Utah and at E&S attracted talent from around the United States. Evans and Sutherland's students were key innovators in the growing field of computer graphics. Among those passing through Utah as students were:

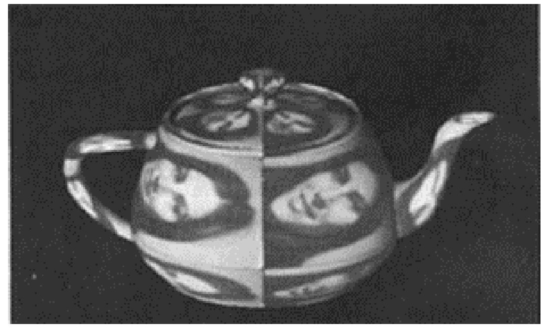
- John Warnock (1969), developer of the hidden-surface removal algorithm, inventor of Postscript, and founder of Adobe Systems. When a 3D object is viewed, only some of its "sides" are visible. Hidden-surface removal algorithms are methods for quickly determining which elements of an object do not need to be rendered to an image.

Shading



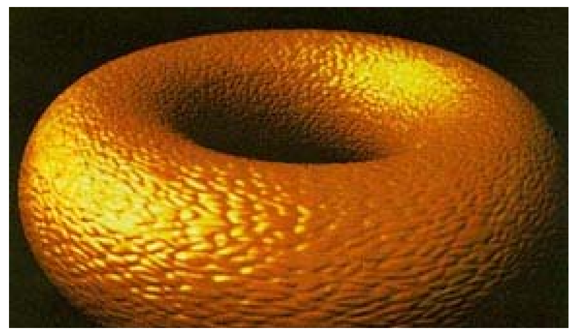
- Henri Gouraud (1971), developer of Gouraud Shading. In 3D graphics, objects are normally constructed out of many adjoining polygons. Where the polygons join, lines appear if the shades of the adjoining areas differ. Gouraud developed a very efficient method of shading polygons to create the appearance of a smooth curved surface.
- Nolan Bushnell (1972), developer of *Pong,* founder of Atari
- Bui-Tuong Phong (1973), developer of Phong Shading. Phong shading was an advance on Gouraud shading, though computationally more demanding.

Texture Mapping from Catmull's Thesis



- Edwin Catmull (1974), developer of texture-mapping and of the z-buffer algorithm, co-founder of Pixar. A texture is a repeated pattern, like bricks in a wall or the weave of a carpet. Catmull developed methods for assigning patterns to polygons so that apparently complex scenes could be described and rendered efficiently. The z-buffer algorithm is a method for coding 3D information so that hidden surface removal is very efficient.
- Frank Crow (1975), developer of anti-aliasing methods. When a line, or the edge of a polygon, is rendered into dots or pixels on a screen, there are small jumps in the position of the edge. Particularly distracting is that the intensity of these jumps varies greatly with angle of the line or edge changes. Anti-aliasing methods minimize the visual artifacts introduced by raster imaging.

Bump Mapping by Blinn



- James Blinn (1976), developer of bump mapping. Blinn extended Catmull's textures to include 3D textures, in which the shading of the "bumps" within the texture varied with the position on the object, giving a much more realistic look under lighting.
- Henry Fuchs (1980), developer of BSP-trees. Binary Space Partitioning (BSP) trees are databases that store the elements (polygons) of an object or scene. The BSP stores the polygons so that the special relationships among them can be quickly ascertained.
- Jim Clark (1982), developer of the first geometry engine, founder of Silicon Graphics, Inc. A geometry engine is a special purpose VLSI processor for computer graphics that performs basic mathematical operations central to 3D computer graphics.

Computer graphics technology, at each stage of its development, stood in a state of tension between the goals of rapid real-time 3D response and representational accuracy. A flight simulator, for example, had to depict an aircraft carrier with enough realism to train the pilot, but the aesthetics of light and shade or of photo-realistic metal and water were much less important than real-time response. A movie special-effect, by contrast, needed to look very real. But because movies were not interactive, a special effects laboratory could spend hours of computer time to create a few seconds of special effects. Given this tension, two separate computer graphics traditions developed. The graphics technologies developed at the University of Utah were chiefly aimed at the efficient rendering of 3D models. By contrast, a second tradition developed in New York that focused on creating high-quality computer-aided film animations.

In 1975 Alexander Schure, founder of the New York Institute of Technology (NYIT) decided to boost the Institute's work in animation by building on the ideas generated at the University of Utah. He hired Ed Catmull (above) soon after his graduation from UU and a number of other

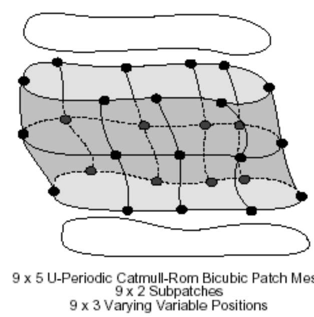
UU students and researchers, including Jim Blinn and Jim Clark. The team at NYIT developed key computer animation technologies that later evolved into Disney's CAPS (Computer Animated Production) system.

The NYIT technology moved west when George Lucas, looking ahead from his success with the first *Star Wars* movie, sought help in computer animation and was guided to NYIT. Five key people left NYIT, including Ed Catmull, to form the special effects department within Lucasfilm. In 1979, Lucas formed a computer graphics division, making Catmull the president. In 1986, the computer graphics division (then a part of Industrial Light and Magic), split off to become Pixar. Lucasfilm would continue to have access to Pixar's rendering technology. Ed Catmull became president of Pixar.

Over the next 15 years, Pixar became the leading motion picture animation and computer generated graphics company in the world. Its technology was used on films including Finding Nemo, Toy Story, Star Wars Episode II, Matrix, the Lord of the Rings, and many others.

At the core of Pixar's technology strategy was the development of RenderMan in 1988, a software standard for describing 3D graphical scenes and for generating data files containing scene descriptions. Just as Postscript was a "language" for describing an image on a 2-D page, RenderMan was a language for describing a 3D scene. The Renderman "standard" permitted the separation of the artistic job of creating the specifications for a scene and the computational job of turning it into an image (rendering).

Excerpt from RenderMan Manual



The Cornell University Program of Computer Graphics (PCG), under the direction of Donald Greenberg, also pioneered key technologies. With a focus on lighting and color, PCG's program has been to use computers to solve the complex wavefront equations describing how reflected light behaves.

Silicon Graphics

Silicon Graphics Inc. (SGI) was formed by Jim Clark, a graduate of the University of Utah and a Stanford faculty member, in 1982. SGI's product focus was straight forward: the company was devoted to supplying high-performance graphics workstations to industrial and scientific users. From the beginning, SGI specialized in 3D graphics and it continually tried to have the highest performing graphics workstations on the planet.

SGI's influence on the graphics industry was immense. It not only provided the highest performing graphics hardware, it also developed a special graphics language (GL) that became the standard for many engineers in the industry. Eventually, a form of GL achieved wide distribution as the freely available OpenGL. Other firms competed with SGI in the making and selling of high-end graphics workstations (e.g., Apollo and HP), but SGI's graphics language, its intellectual leadership, and its continuing efforts to push performance frontiers kept it in a leadership position. Together, SGI's hardware and GL framed the problem of 3D graphics in a way that became the dominant logic in the industry. At the heart of this logic was the concept of the *graphics pipeline*:

Application

- At the start of the graphics pipeline was the application program. This was the drawing program, flight simulator, game, or industrial design program that created the 3D scene. It created the scene by specifying three key elements: (1) objects, (2) surface textures, and (3) light sources.
- The scene is fed into the graphics pipeline as a set of standardized commands (for example, OpenGL calls defining surfaces).Normally, the application will define the surfaces of objects in terms of polygons. If curved surfaces have been defines, they are tessellated—broken into polygons. Any polygons with more than three sides are further tessellated into triangles. Each triangle is described as a set of three verticies, each vertex defined by three coordinates.

Transformation & Lighting

- In the transformation step, the coordinates of each vertex of each polygon are transformed to be relative to the (moving) viewer, or camera. Thus, each triangle is projected, or transformed, into the triangle it becomes on the viewing plane.
- Triangles that "face backwards" are culled.
- Each triangle's illumination is calculated from its position and the position of the light(s).

Setup

- Each triangle facing the viewer must be identified with the set of pixels on the viewing screen it affects. This requires a projection of each triangle onto a view plane, followed by identification of the pixels constituting its edges and interior. The viewing plane has edges. Some triangles may be completely outside the viewing window and can be eliminated (culled). Others may intersect the viewing window and need to be clipped.
- The projection of an arbitrarily shaped object onto a viewing plane can be a very complex operation. However, a triangle always projects as another triangle (or a line).

Rasterization

- Triangles at different depths from the viewing plane may potentially affect the same pixels. Perspective corrections and issues of overlap are sorted out in the raster operations stage of the pipeline.

Texture & Shading

- Texture and shading operations determine the color of each fragment depending upon the lighting specifications and the texture assignments. Fog and haze corrections are applied.

Raster Operations

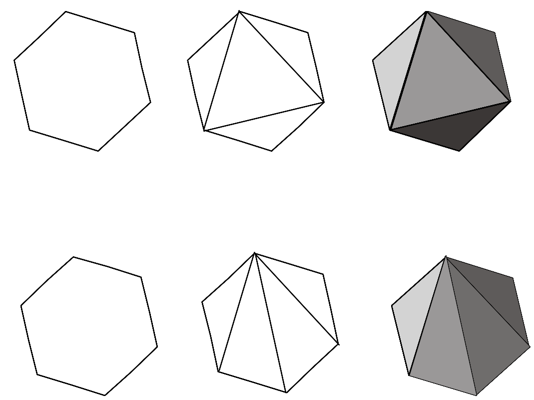
- Raster operations include blending and z-buffering. The "z-buffer" is the measured the distance from the view plane to each fragment's contribution to a pixel. This step makes sure that scene elements "in front" cover those "behind," with blending computations for cases of transparency.
- The final step in the pipeline is to build the frame buffer (actual screen data).

The graphics pipeline and the GL language did not specify how these operations were performed. They could be carried out by the computer's CPU, or they could be carried out by special-purpose graphics hardware. As the capabilities of integrated circuits grew over time, more and more of the graphics pipeline was carried out in specially designed processors. SGI's first generation machines (IRIS 1400) performed polygon transformations. Using a parallel architecture, it could transform many triangles at once from world coordinates to their screen projections. However, all other calculations, including lighting, shading, and surface elimination, were done by the application.

SGI's second generation hardware added the capability to rasterize Gouraud shaded triangles and handle z-buffering operations.

SGI's third generation hardware, the Reality Engine, was introduced in 1992, added hardware texture mapping and full anti-aliasing. The Reality Engine was realized as 3-to-6 boards plugged into a MIPS RISC workstation. The geometry board held geometry engines—processors capable of 100 million floating point operations per second. These engines tessellated polygons into triangles and projected triangles onto the viewing plane. The particulars of the transformed triangles were then broadcast to the rest of the system on a "triangle bus." Each raster board held five fragment generators—processors that grabbed triangles off of the triangle bus and generated the potential pixel information they implied. Also on each raster board were 80 image engines—processors that constructed the actual image information from the relevant triangle fragments.

3D Objects Constructed from Shaded Triangles



In a three dimensional space, three points determine a plane. Consequently, the vertices of a polygon with four or more sides do not necessarily lie in the same plane. Here, the vertices of two hexagons lie on different planes, implying two different 3D objects. As the viewing angle changes, one or even two of the vertices may be hidden. By contrast, changes in point of view never hide the vertices of a triangle. Breaking the 3D object's surface into triangles provides a simple way of transforming the point of view—each triangle can be transformed independently.

Rendering performance of computer graphics systems can be measured in a variety of ways. Initially priced at about $1 million, and standing 4 feet high, the Reality Engine was capable of processing 1 million triangles per second. In 1996, SGI introduced the InfiniteReality Engine, capable of processing 7 million triangles per second. Again realized on multiple boards, the InfiniteReality engine applied improvements in integrated circuits performance and cost to increase the capabilities of the original Reality Engine. Also again, the target customer was very high-end engineering, scientific, and entertainment applications.

The Technology of PC Graphics

Early personal computers, like the Apple II (1978), incorporated simple 2-D color graphics capabilities, as did certain versions of the IBM PC (1981). Before the mid-1990s, however, there was little consumer demand for hardware accelerated 3D graphics on the PC platform. The explosion of interest in 3D graphics on the PC grew out of the simultaneous convergence of innovation in 3D game design, falling prices of memory and integrated circuits, and innovation in 3D hardware acceleration techniques.

Crawling Up the Pipeline

Before the mid-1990s, there was no acceptance of a standard way of representing a 3D object or scene. Consequently, first generation 3D "accelerator" cards focused on basic raster operations—building the final buffer of colored pixels to be sent to the video monitor. The first product to break with this approach was NVIDIA's initial product, the NV1, introduced in 1965. The NV1 assumed that a 3D object's surface was described by 4-sided non-planar polygons—quadratic curves in three-space. It thus opened the door to the accurate representation of complex curved surfaces. But the NV1 was a flop—it did not gain acceptance from game programmers and was part of the reason a potentially lucrative deal with Sega, a console maker, fizzled.

The first chip-set that provided a substantial and instantly noticeable improvement in real-time 3D graphics was *Voodoo*, introduced by 3Dfx in 1996. The Voodoo hardware provided much faster handling of texture-mapping and full-image anti-aliasing. In particular, Voodoo supported "mipmapping," a technique for smoothly scaling textures with distance. Absent mipmapping, the patterns in textures would suddenly shift and "jump" as the viewer's distance to the surface changed.

The 3Dfx Voodoo chipset supported triangle processing. The popularity of Voodoo in 1997-98 pushed the game industry to converge on the standard of representing surfaces with triangles. Because a triangle is always planer (three points determine a plane), computation of the location of points inside the triangle is very simple. The triangle standard was supported by OpenGL and was also the basis of Microsoft's graphics API, DirectX, released in 1998. NVIDIA also adopted the triangle approach in its first successful product, the Riva128 (4/97).

Though complex in its details, further innovation in 3D graphics on the PC platform had a fairly straightforward outline through 2000. The mainline program was to reproduce the SGI pipeline, and, by implication, the *RealityEngine*, as a mass-produced consumer product. This was first achieved by NVIDIA's *GeForce 256* chip in late 1999.

The GeForce 256 GPU contained 23 million transistors and supported 64 megabytes of onboard frame buffer memory. The triangles it handled were fed to 4 parallel on-chip pixel processing pipelines, providing a very high level of performance—15 million triangles per second and 480 million pixels per second. ATI, pressing to catch up with NVIDIA, introduced its first GPU in 2000. Both firms added features to improve performance over the next two years.

Lighting Equations

Working with first-generation GPUs, graphics designers had found that a wide variety of visual effects could be achieved by clever use of the GPU's features. In general, the possibilities for creativity depended upon the *lighting equations* implemented on a GPU. For example, in 2000 GPUs used lighting equations[1] like (1).

$$Light = (Ambient + Diffuse \times Texture + Specular] \times (1 - Fog) + Fog \times Fog\_Color. \qquad (1)$$

[1] These lighting equations, or lighting models, were essentially properties of the APIs. Equation (1) is a simplification of technology in use by DirectX 8 (2000).

Here, the final color of a rendered point is the product a computed color and the amount of "fog," together with the color of the fog. The computed color, in turn, depends upon the color of the *Ambient* light, the color of the Diffuse lighting of the material, the *Texture* map being applied, and the *Specular* light. The specular light determines indicates the shininess of the object—its intensity varies sharply with the angle between the surface and the viewer.

As a simple example, a fine mesh texture applied to a diffusely lit brown floor might look like a carpet. Increasing the intensity of the specular component of the light might make it look instead like metal grating.

Research showed that this approach could create the illusion of many different materials. For example, full use of the specular component made an object look like a mirror or like chrome. However, if the specular component were tinted yellow, the material looked like shiny gold. If tinted green it looked like reflective copper.

GPUs that provided multiple textures let designers create even more striking effects. By mixing two or more textures, novel and intriguing visual effects could be generated.

Programmable GPUs

Despite the variety of effects achievable with first-generation GPU technology and lighting equation implementations, there were effects graphics designers could not easily achieve. No matter how the lighting equation was used, skin tended to look like plastic. Complex entities like hair, foamy water, smoke, tree bark, and cloth resisted realistic depiction. Techniques for each of these problems could be developed, but which should be implemented in hardware? And, whose solution should be implemented? The approach adopted by the industry was to leave the question to the designer, "exposing" (opening to the user) the lighting functions of the GPU.

Vertex and Pixel Shaders

Once a surface has been broken into triangles, the GPU pipeline next operated on the vertices of each triangle. Each vertex was assigned a transformed position together with lighting and texture properties. The color and intensity of pixels between the vertices were obtained by interpolation of the vertex lighting and texture mapping. A programmable vertex shader gave the user the option of bypassing the GPU's fixed vertex processing functions and passing a set of commands—a program—to the GPU. The first consumer vertex-shader product was NVIDIA's GeForce 3, introduced in February of 2001.

Vertix "shaders" actually allowed the position, as well as the lighting, of a vertex to be altered under program control. The vertex shader was implemented as a small CPU that could store up to 128 instructions. It operated on input registers that stored the position, texture, and illumination conditions of a vertex, and wrote to an output register that held the resultant computed properties of the vertex.

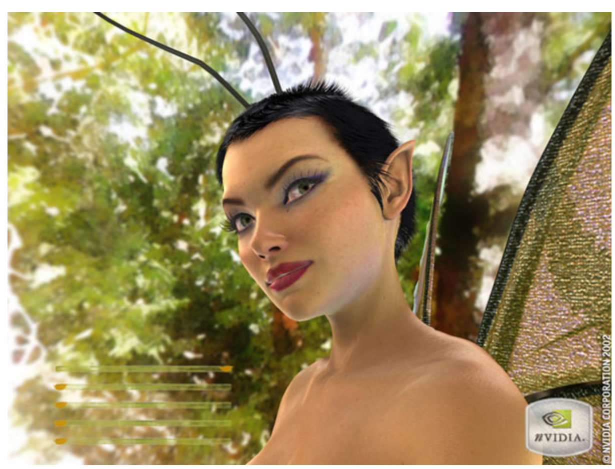
Some of the more intriguing uses of vertex shaders involved animation rather than lighting. Vertex shaders were used to "skin" characters, allowing the triangles defining the surface to move and shift as some underlying structures ("the skeleton") shifted. Vertex shaders also allowed water waves to be created, as vertices were shifted in cyclic patterns.

Pixel shaders were created by exposing parts of the rasterization pipeline to the user—creating programmable pixel engines. The GeForce 3 also contained pixel shaders, though later GPUs had much more developed pixel shaders. These shaders gave developers that ability to create per-pixel lighting effects that dramatically increased the realism of the rendered scene.

Pixel shaders allowed the implementation of bump mapping, first developed by Blinn in 1976, on a consumer device. A complex example of the use of pixel shaders was the Dawn Demo, created by NVIDIA in 2003. Dawn was an imaginary fairy creature animated in real time. Her skin texture and tone were created by careful attention to both how the skin surface looked and how it moved as the character moved:[2]

> Skin is unlike most surfaces that we model in computer graphics because it is composed of layers of semi-translucent epidermis, dermis, and subcutaneous tissue. *Subsurface scattering* is the term we give to the phenomenon whereby light enters the surface, scatters through the medium, and exits at another point. This effect can commonly be seen as an orange glow of light passing through the skin when you hold your hand up in front of the Sun. This scattering under the skin is important to skin's appearance at all angles and helps give it a soft, highly distinctive character. Unfortunately, this reality defies a common assumption in graphics APIs and architectures: namely, light at one point on an object doesn't affect reflection at other points. In the past, some groups have tried to emulate skin's complexity using multiple, layered texture maps. In general this approach has proven to be difficult to manage and difficult for the texture artists to work with as they previsualize the final blended color. Instead, we used a single color map, with color variations added through the shading….. Furthermore, skin has extremely fine variations that affect its reflective properties. These have subtle effects on skin's appearance, particularly when the light is directly opposite the camera position—that is, edge and rim lighting. Real skin has tiny features such as vellus hairs and pores that catch the light. These details were too small for us to model explicitly, but we wanted a surface that still gave us an appropriate overall look. Adding a bump map provided some additional detail when seen in close-up—particularly for small wrinkles—but we wanted a soft appearance, not shiny, stippled plastic, and we wanted the effects to be visible regardless of the size on screen (bump maps are usually visible only when seen up close). We approximated both of these shading attributes by recognizing that we could model them as simple formulas based on the surface normal and the lighting or viewing vectors. In particular, along silhouette edges we sampled the lighting from *behind* Dawn, as indexed by the view vector—mixing the light coming "through" Dawn

NVIDIA Dawn Demo



[2] Curtis Beeson and Kevin Bjorke, *GPU Gems*, Chapter 3, "Skin in the 'Dawn' Demo," developer.nvidia.com /docs/IO/11801/Chapter_3.pd

> with her base skin tones to create the illusion of subsurface and edge-effect lighting, particularly for very bright areas in the background map.

In another demo, NVIDIA engineers created a "time machine" effect on the image of an old pick-up truck. The truck was constructed from painted metal, wood, chrome, and glass. Under user control, the truck appeared to age—colors faded, the paint oxidized and developed random surface bubbles, and rust advanced. The wood faded and cracked, chrome welted and corroded, and glass fogged. The paint effects were accomplished with 60 instructions to the pixel shaders, mixing 11 different textures.

Shader Languages

Demonstrations by NVIDIA and ATI showed what vertex and pixel programming could accomplish. Nevertheless, developers were slow to plunge headlong into this new technology. The key impediments were lack of backward compatibility and the difficulty of developing shader programs. The backward compatibility problem arose because the shader programs would not be executed unless the user's PC had the most recent GPUs installed. Most users did not, and would not therefore see the advanced results. The Dawn demo, for example, would not even run unless the user had an NVIDIA card of recent vintage.

The second problem—the difficulty of writing shader programs—was tackled in a several ways. NVIDIA developed a high level language, Cg, designed to make it easier to work with pixel and vertex shaders. Microsoft's DirectX 9 included the definition of a comparable language, called HLSL (high level shader language). In addition, a number of firms developed tools for the development of shaders. In particular, NVIDIA offered the GUI-based *FX Composer* and ATI offered a suite of development tools called *RenderMonkey*.

Glossary

The following glossary of 3D graphics terms is adapted from the 3D Glossary published on NVIDIA's website.

3D

Having or appearing to have width, height, and depth (three-dimensional).

Accelerated Graphics Port (AGP)

An industry-standard expansion bus found in most modern PCs, specifically designed for graphics cards. It is a faster alternative to the PCI bus and allows graphics programs to store large amounts of data temporarily in the computer's system memory. The speed at which a GPU can access and use the information across the AGP bus has a large impact on graphics performance. NVIDIA's GPUs are fully optimized to take advantage the fastest AGP speed currently available.

Alpha Blending

A graphics processing technique that simulates transparency or translucency for objects in a 3D scene to create visual effects like smoke, glass or water. Pixels in the frame buffer of a graphics system include three color components (red, green and blue) and sometimes an alpha channel component as well. The alpha channel data stores the degree of transparency, ranging from opaque to completely clear.

Antialiasing

Any technique for reducing the visual impact of aliasing, or the "jaggies," on a computer graphics system.

Anisotropic Filtering

An advanced texture-filtering technique that improves image quality for scenes with objects that extend from the foreground deep into the background. For example, a road that extends to the horizon will look better with anisotropic filtering.

Application Programming Interface (API)

A standardized programming interface that enables developers to write their applications to a standard and without specific knowledge of hardware implementations. The benefit is that a single application can run on a wide range of hardware platforms instead of needing to be rewritten for each of those hardware platforms. The software driver for the hardware intercepts the API instructions and translates them into specific instructions tailored to specific hardware. APIs such as DirectX and OpenGL can also emulate hardware functions in software to ensure that the application will run even if the hardware platform is missing a desired feature.

Aspect Ratio

The ratio of the width of the image to its height, expressed as width:height. A standard U.S. television screen or computer monitor has a 4:3 (pronounced "four by three") aspect ratio. Some high-definition television (HDTV) broadcasts are formatted in a 16:9 (1.78:1) aspect ratio. Most feature films have a 1.85:1 aspect ratio.

Buffer

Memory dedicated to a specific function or set of functions. For example: the graphics memory functions as a frame buffer, but can also be used as a Z buffer or a video buffer. Smaller buffers exist many different places inside a GPU as well and serve as temporary storage areas for data and instructions.

Bump Mapping

A shading technique using multiple textures and lighting effects to simulate wrinkled or bumped surfaces. Bump mapping is useful because it gives a 3D surface the appearance of roughness and other surface detail, such as dimples on a golf ball, without increasing the geometric complexity. Some common types of bump mapping are Emboss Bump Mapping, Dot3 Bump Mapping, Environment Mapped Bump Mapping (EMBM) and True, Reflective Bump Mapping. Dot3 bump mapping is the most effective technique of the three.

Dynamic Adaptive Speculative Preprocessor (DASP)

DASP applies a patent-pending, intelligent, pre-processing technology that stores application instructions and data before they are needed. This reduces the bottlenecks that occur between memory and the CPU, and graphics and audio subsystems, thereby boosting overall system performance.

Digital Vibrance Control (DVC)

Allows the user to adjust color controls digitally to compensate for the lighting conditions of their workspace, in order to achieve accurate, bright colors in all conditions. Currently this feature is not available on Mac systems.

DualDDR Memory Architecture

Second-generation NVIDIA nForce platforms include this revolutionary memory architecture consisting of dual-independent 64-bit memory controllers for increased memory bandwidth and lower latency.

Direct 3D® (D3D)

The 3D graphics portion of the Microsoft ® DirectX® API. Many application and game developers use this API to write their software. The developers of the software (usually a game) write instructions to the Direct3D and the graphics driver (a piece of software) translates them to the instructions to the Direct3D and the graphics driver (a piece of software) translates them to the GPU so they can be rendered on your monitor. Microsoft continues to revise Direct3D to make it an industry leading API. All of NVIDIA GPUs products support Direct3D.

DirectX®

A hardware abstraction layer API from Microsoft that is integral to the Windows® operating system. The DirectX standard includes Direct3D, DirectSound, DirectDraw, DirectVideo, DirectPlay, and DirectInput. Microsoft continues to revise DirectX to make it the industry standard consumer graphics API. Microsoft even licensed NVIDIA technology for the latest version, DirectX 8, in order to add programmability to the API. NVIDIA's GPUs support DirectX, and the GeForce3™ GPU is the most complete hardware implementation of DirectX 8.

Double Buffering

A programming technique that uses two frame buffers so the GPU can be working on one frame while the previous frame is being sent to the computer display. This prevents conflicts between the display refresh function and the graphics rendering function. See Frame Buffer. Drivers Software that enables communication between the graphics processor and the rest of the PC. Software drivers are frequently updated to improve performance, quality, and enable new features. Be sure to download the latest drivers as they come become available. NVIDIA is the only graphics company to offer a Unified Driver Architecture (UDA) that works with all NVIDIA GPUs.

DualNet

Part of the nForce2 Digital Media Gateway. DualNet is integrated support for an NVIDIA Ethernet Mac and for a 3Com® Ethernet Mac—allowing a PC to serve as a home gateway, managing traffic between two separate networks and ensuring rapid transfer of data from WAN to LAN without any added arbitration or latency.

Fill Rate

The speed at which your graphics card can render pixels–usually measured in millions of pixels per second (Megapixels/sec). GPUs with higher fill rates can display higher resolutions and more colors at higher frame rates than other chips with lower fill rates. NVIDIA GPUs have the highest fill rates of all GPU available on the market. See Frames Per Second.

Flat Panel Display

A display device that uses liquid crystal display (LCD) technology instead of a cathode ray tube (CRT) like most monitors. Flat panels are much thinner than CRT monitors and offer more flexibility for placement on desks or attaching to walls. NVIDIA GPUs support a variety of flat panel displays.

Fog

A graphics function that simulates the behavior of actual fog and/or mist. GeForce3™ GPUs support a variety of advanced fog calculations including layered fog, patchy fog and more.

Frame Buffer

Memory that is dedicated to the graphics processor and used to store rendered pixels before they are displayed on the monitor.

Frames Per Second (FPS)

The rate at which the graphics processor renders new frames, or full screens of pixels. Benchmarks and games use this metric as a measurement of a GPU's performance. A faster GPU will render more frames per second, making the application more fluid and responsive to user input.

Frequency

Specifically, the number of times per second that a specific event occurs. As applied to semiconductor devices, frequency most often applies to the clocks that control how fast the device can operate. Frequency is measured in Hertz (Hz), which means cycles per second. Semiconductor devices today run with clock speeds of megahertz (MHz), meaning one million cycles per second, or even gigahertz(GHz).

Graphics Processing Unit (GPU)

A high-performance 3D processor that integrates the entire 3D pipeline (transformation, lighting, setup, and rendering). A GPU offloads all 3D calculations from the CPU, freeing the CPU for other functions such as physics and artificial intelligence.

Graphics Pipeline

The series of functions, in logical order, that must be performed to compute and display computer graphics.

High-Definition Video Processor (HDVP)

Turns your PC into a fully functional DVD player, and an HDTV player with the purchase of an additional third-party decoder.

High-Resolution Antialiasing (HRAA)

Delivers fluid frame rates of 60 frames per second or more at high resolutions, such as 1024x768x32 or higher, with full-scene antialiasing (FSAA) turned on. Featuring the Quincunx AA mode, HRAA delivers a high level of detail and performance for all applications.

HyperTransport Technology

A state of the art I/O bus interface, delivering the highest continuous throughput—800MB/s between the nForce platform processors. Ensures data and information are relayed through the system as quickly as possible.

High-Definition Television (HDTV)

A high-definition TV (HDTV) signal that offers higher resolutions and a wider aspect ratio than a traditional TV signal.

I/O

Input/Output. This is a general term to describe any bi-directional interface on a semiconductor device. The higher the I/O speed, the faster information can be exchanged.

Isochronous

Time-dependent. Processes where data must be delivered within certain time constraints. For example, multimedia streams require an isochronous transport mechanism to ensure that data is delivered as fast as it is displayed and to ensure that the audio is synchronized with the video. delivered as fast as it is displayed and to ensure that the audio is synchronized with the video.

Integrated Graphics Processor (IGP)

An IGP replaces the "Northbridge" of traditional motherboard architectures. NVIDIA IGPs feature GeForce(TM) MX -powered graphics with innovative system and memory enhancements including DASP to help boost CPU performance, and HyperTransport(TM) technology—a high-performance I/O bus interface.

Jaggies

A slang term used to describe the stair-step effect you see along curves and edges in text or bit mapped graphics. Antialiasing can smooth out jaggies.

Lightspeed Memory Architecture

NVIDIA memory bandwidth optimizations designed to make complex scenes faster. These optimizations make full-scene antialiasing (FSAA) practical for the first time, enabling users to enjoy high-resolution antialiasing (HRAA).

Lightspeed Memory Architecture II

LMA II boosts effective memory bandwidth by up to 300%. Radical new technologies including Z occlusion culling, fast Z-clear, and auto pre-charge effectively multiply the memory bandwidth to ensure fluid frame rates for the latest 3D and 2D games and applications.

Lighting

In 3D graphics, lighting is used to calculate the degree to which an object is affected by a source. Lighting is used in games to create realistic looking scenes with greater depth instead of flat looking or cartoonish environments. NVIDIA's GeForce3 GPUs are capable of custom lighting effects through the vertex shader capabilities of the nfiniteFX engine.

Lossless Z -Compression

LMA II contains a lossless form of Z-compression that delivers a 4:1 benefit. Compression is a crucial technique in saving memory bandwidth for higher performance.

Mipmapping

A technique to improve graphics performance by generating and storing multiple versions of the original texture image, each with different levels of detail. The graphics processor chooses a different mipmap based on how large the object is on the screen, so that low-detail textures can be used on objects that contain only a few pixels and high-detail textures can be used on larger objects where the user will actually see the difference. This technique saves memory bandwidth and enhances performance.

Mobile AGP Package (MAP)

Brings the benefits of AGP add-in cards to notebook computers by fitting multiple graphics solutions into a package small enough for thin and light notebooks. High-performance graphics are no longer limited to larger notebooks. MAP also paves the way to faster adoption of new graphics technologies in current-generation notebook designs.

Media and Communications Processor (MCP)

Part of the NVIDIA nForce and nForce2 platform processing architectures, the MCP replaces the "Southbridge" of traditional motherboard architectures. The MCP (including the MCP-D and MCP delivers the most complete suite of integrated networking and communications devices including Ethernet, HomePNA 2.0, IEEE-1394a/FireWire(R) port, and up to six USB ports. In addition, the integrated audio processing unit (APU) provides support for Dolby(R) Digital 5.1 encoding.

NVIDIA nForce Platform Processors

The NVIDIA nForce and NVIDIA nForce2 platform processing architectures revolutionizes traditional motherboard architectures and provide new levels of performance and functionality to the AMD-based desktop PC market through the IGP, SPP and MCP platform processors.

NVIDIA StreamThru Data Transport System

NVIDIA's patent-pending isochronous data transport system, providing uninterrupted data streaming for superior networking and broadband communications. StreamThru assists in making streaming video and audio smoother and jitter-free.

NVIDIA nfiniteFX Engine

NVIDIA's programmable Vertex and Pixel Shaders, and 3D textures. The nfiniteFX engine gives developers the freedom to program a virtually infinite number of custom special effects, in order to create true-to-life characters and environments.

NVIDIA Shading Rasterizer (NSR)

Brings natural material properties (smoke, clouds, water, cloth, plastic, etc) to life via advanced per-pixel shading capabilities in a single pass.

nfiniteFX II Engine

The NVIDIA nfiniteFX II engine incorporates dual programmable vertex shaders, faster pixel shaders and 3D textures--giving developers the freedom to program a virtually infinite number of custom special effects to create true-to-life characters and environments. nfiniteFX II is a feature of all GeForce4 Ti GPUs; the GeForce4 4200 Go GPU; and the Quadro4 980, 900, and 750 XGL GPUs.

nView Multi-Display Technology

The nView hardware and software technology combination delivers maximum flexibility for multi display options, and provides unprecedented end-user control of the desktop experience. NVIDIA GPUs are enabled to support multi-displays, but graphics cards vary. Please verify multi-display support in the graphics card before purchasing.

NVIDIA SoundStorm

Combined with the APU, NVIDIA SoundStorm(TM) audio provides Dolby Digital 5.1 encoding and connections for headphones; front left and right, and rear left and right speakers; a center channel; and subwoofer connection.

NVIDIA Sceneshare Technology

NVIDIA Sceneshare technology extends the DVD experience by allowing users to bookmark their favorite movie scenes and easily share them with other NVDVD users. Users can send sceneshares to their friends via e-mail or post them for download.

NVIDIA SoundStorm Certification

The SoundStorm™ Program ensures NVIDIA nForce™2-based motherboards and turnkey PC systems are designed and produced to the high-quality standards defined by NVIDIA and Dolby Laboratories. This involves passing rigorous tests conducted by NVIDIA and Dolby Laboratories for Laboratories. This involves passing rigorous tests conducted by NVIDIA and Dolby Laboratories for Dolby® Digital 5.1 compliance, connectivity, and availability to consumers for the best out audio experience possible.

OpenGL

A graphics API that was originally developed by Silicon Graphics, Inc.™ (SGI) for use on professional graphics workstations. OpenGL subsequently grew to be the standard API for

CAD and scientific applications and today is popular for consumer applications such as PC games as well.

Pixel Shaders

Part of the nfiniteFX engine, Pixel Shaders alter lighting and surface effects that replace artificial, computerized looks with materials and surfaces that mimic reality.

PowerMizer

Mobile Technology Advanced hardware and software technology specifically designed to extend the battery life of notebook PCs. Ensures that users enjoy cinematic quality and performance for extended periods of time.

Per-Pixel Shading

The ability to calculate lighting effects at the pixel level, greatly increasing the precision and realism of the scene. With NVIDIA's GeForce3 GPU, game developers can now program custom per-pixel effects.

Pixel

Shorthand for "picture element." A pixel is the smallest element of a graphics display or the smallest element of a rendered image.

Pixels Per Second

The units used to describe the fill rate of a GPU. It is usually measured in millions of pixels per second (Megapixels/sec).

Polygon

The building blocks of all 3D objects (usually triangles or rectangles) used to form the surfaces and skeletons of 3D objects.

Programmable

Configurable with software commands. NVIDIA's nfiniteFX II and nfiniteFX Engines are programmable and can be configured with software to create an infinite variety of special effects.

Quincunx Antialiasing

A patented antialiasing technique enabled by the GeForce3 GPU. Quincunx AA offers the quality of the slower 4X AA mode at very near the performance of the faster, 2X AA mode.

Quad Cache

LMA II contains a caching system for primitives, vertices, textures and pixels. These caches are individually dedicated and optimized for almost instant graphics pipeline access and reuse.

RAMDAC

Acronym for random access memory digital to analog converter. A RAMDAC is a functional unit of a graphics processor that converts the digital data stored in the frame buffer into analog data that can be sent directly to an analog monitor.

Refresh Rate

The frequency at which the electron guns in your monitor redraw the image, measured in Hertz (Hz) or cycles per second. As an example, a refresh rate of 60 Hz means the screen is redrawn 60 times per second. Higher refresh rates reduce or eliminate image "flicker" that can cause eye strain.

Rendering

The process of taking information from a 3D application and displaying it as a final image.

Shadow Buffers

Part of the nfiniteFX II and nfiniteFX Engines, shadow buffers enable self -shadowing for characters and objects, and soften the edges of shadows for realistic effects, adding depth to scenes and highlighting spatial relationships between objects.

SSE/SSE3

Intel® Streaming SIMD Extensions (SSE) and SSE2 are sets of instructions for accelerating multimedia applications. SSE is found on Intel Pentium® III processors; SSE2 is Intel's new instruction set supported on Intel Pentium® 4 processors. Some of the benefits of SSE/SSE2 include rendering higher quality images, high quality audio, MPEG2 video, and simultaneous MPEG2 encoding and decoding, and reduced CPU utilization for speech recognition. NVIDIA's GPUs are fully optimized to take advantage of SSE and SSE2 instruction sets.

Stencil Buffer

The section of the graphics memory that stores the stencil data. Stencil data can be used to mask pixels for a variety of reasons, such as stippling patterns for lines, simple shadows and more.

System Platform Processor

(SPP) Part of the NVIDIA nForce and nForce2 platform processing architectures, the SPP replaces the "Northbridge" of traditional motherboard architectures. The SPP offers the same features as the nForce IGP, with the flexibility of an add-in card graphics solution—so you can always upgrade to the latest, NVIDIA-based, high-performance graphics card. The SPP features an external AGP 8X or 4X bus for the ultimate in graphics upgradeability.

TwinBank

Memory Architecture The first-generation nForce memory architecture. TwinBank allows the CPU, and the graphics and audio sub-systems simultaneous access to the system's memory bandwidth, guaranteeing continuous access for all applications, all the time.

TwinView

Dual -Display Architecture NVIDIA's multiple display technology. TwinView boosts productivity by enabling the user to have two simultaneous displays without a second graphics board. NVIDIA GPUs are enabled to support multi-displays, but graphics cards vary. Please varify multi-display support in the graphics card before purchasing.

Transform & Lighting (T&L)

Two separate engines on the GPU that provide for a powerful, balanced PC platform and enable extremely high polygon count scenes. Transform performance determines how complex objects can be and how many can appear in a scene without sacrificing frame rate. Lighting techniques add to a scene's realism by changing the appearance of objects based on light sources.

Texel

The smallest unit of a texture map, similar to pixels being the smallest unit of a rendered image.

Texture

An image file (such as a bitmap or a GIF) that is used to add complex patterns to the surfaces of objects in a 3D scene.

Texture Compression

Compressing larger textures to smaller ones in order to conserve memory. This can make games run faster as well as allow more textures to be used per scene, adding richness and detail to the 3D environment. NVIDIA GPUs support DirectX and S3TC texture compression, making NVIDIA the only company to offer complete support for texture compression on all industry standard APIs.

Texture Mapping

The process of applying a texture to the surface of 3D models to simulate walls, sky, etc. Texture mapping enables developers to add more realism to their models.

Triangles Per Second

The rate at which a GPU processes triangles. It is a common industry metric for describing performance. The higher the number of triangles per second, the faster the GPU. The GeForce3 GPU delivers the highest triangles per second rate of any GPU. Triple Buffering A step beyond double buffering that uses an additional back buffer to process the next image, resulting in smoother animation. With triple buffering, the GPU can start rendering a third frame while the first frame is being displayed and the second frame is waiting to be displayed.

Triple buffering helps to insure that the GPU is never idle because it is waiting for rendered frames to be sent to the monitor.

Unified Driver Architecture (UDA)

Part of the NVIDIA Forceware unified software environment (USE). The NVIDIA UDA guarantees forward and backward compatibility with software drivers. Simplifies upgrading to a new NVIDIA product because all NVIDIA products work with the same driver software.

USB 2.0

A standard plug and play interface providing easy-to-use connectivity for USB devices.

Vertex Shaders

Part of the nfiniteFX engine, Vertex Shaders are used to breathe life and personality into characters and environments. For example, through vertex shading developers can create true life dimples or wrinkles that appear when a character smiles. Video Processing Engine (VPE) Integrated high-definition Video Processing Engine delivers the highest quality DVD, video, and display output available in the market today. Integrated hardware MPEG2 decoder reduces CPU utilization for DVD playback to provide a longer viewing experience.

Z-Buffer

The area of the graphics memory used to store the Z or depth information about rendered objects. The Z -buffer value of a pixel is used to determine if it is behind or in front of another pixel. Z calculations prevent background objects from overwriting foreground objects in the frame buffer.

Z-Correct Bump Mapping

The nfiniteFX II engine is capable of making intersecting bump-mapped polygons look realistic and accurate. This is especially important in scenes where water and land interact with each other.

.