



Technical Brief

Per-Pixel Lighting and
Bump Mapping with the
NVIDIA Shading Rasterizer™

*N*VIDIA

Executive Summary

The NVIDIA Quadro2™ line of workstation graphics solutions is the first of its kind to feature hardware support for real-time per-pixel operations. The Quadro2 family includes the NVIDIA Shading Rasterizer™ (NSR) technology first introduced in GeForce2 GTS™. The NSR uses innovative register-based combiners and hardware acceleration of dot-product calculations for dramatic, programmable pixel-shading capabilities. Quadro2 family graphics boards enable PC workstation users to model physical light sources that react accurately to dynamic events in world space. Previous schemes based on light maps or vertex lighting compromised quality and accuracy for performance, forcing users to choose between real-time rendering or full-featured rendering. The NSR enables real-time per-pixel rendering so that workstation users can see the results of their efforts immediately without waiting for offline rendering. Artists greatly benefit from this real-time feedback and interaction because the creative process isn't interrupted (and time isn't wasted) by the need to render offline to see the effects of edits. Mechanical CAD users benefit as well because they now have the choice of working in either wireframe or fully-textured modes with robust performance. With the Quadro2 family it is now possible to calculate the diffuse, specular, and ambient light properties of every rendered pixel in real time. As a result, professional users can enjoy features and performance levels previously available only to non-real-time software applications running on supercomputers and render farms.

By delivering the NSR architecture in professional graphics solutions, NVIDIA delivers a complete solution for concurrent develop-and-deploy content development on a single platform. Designers, graphic artists, modelers, animators and engine authors get a complete superset of the graphics technology that is leading the industry in the consumer entertainment PC and powering the next-generation Microsoft X-Box game console. The Quadro2 family shatters the barriers of the digital content creation process.

Introduction

In the fall of 1999, NVIDIA's introduction of the GeForce 256™ GPU, with its support for hardware accelerated transformation and lighting operations, allowed developers to dramatically increase the amount of geometry in a real-time rendered scene. Objects populating simulated worlds became much more complex, less "flat," and more lifelike than before.

In the pre-GPU world, programmers and designers were forced to compensate for the CPU's inability to handle significant amounts of transform operations by simplifying the geometry of their object models. Since the resulting world was limited in terms of geometry, developers resorted to clever uses of texture maps to create the illusion of detail with respect to both the underlying geometry of models as well as its interaction with light.

In cases where there is no requirement of real-time rendering, intensive software calculations are capable of producing spectacular virtual worlds such as those in Pixar's *A Bug's Life*. But with this approach, each frame of animation typically requires hours, even days of calculations for rendering, even when the process is distributed across a "farm" of thousands of high-end workstations. Ideally, this caliber of accurate, high-quality lighting and the palette of cinematic effects it enables would be available to designers in real time through the assistance of specialized graphics hardware.

NVIDIA's new Quadro2 family and GeForce2 GTS™ GPU overcome the limitations of previous graphics solutions and enable software developers to calculate lighting characteristics on a per-pixel basis in real time using the NVIDIA Shading Rasterizer. Per-pixel lighting functions are more accurate and more flexible than any previous lighting routine available. 3D visualization applications and entertainment software running on mainstream PCs will be able to achieve a level of realism previously impossible without inordinate time and the computing power of expensive workstations or supercomputers.

Traditional Lighting Methods

The mathematical process of calculating how even a single, static light source interacts with a solitary object is tremendously complicated. Even with the advent of modern graphics hardware featuring highly optimized multitexturing abilities and support for transform and lighting operations, traditional lighting techniques require compromises in quality and accuracy in order to achieve acceptable levels of performance. The two most popular techniques for lighting real-time 3D scenes today utilize either light maps or a method called *vertex lighting*. Both of these approaches have their advantages but unfortunately neither provides a solution that is appropriate for all lighting situations.

Light Maps

Light maps are 2D textures that are either produced by an artist or generated algorithmically. Brighter regions of a light map correspond to higher intensities of light and darker regions represent lower light intensities. *Figure 1* represents a standard light map that will be used to illustrate how a spot light effect is created.

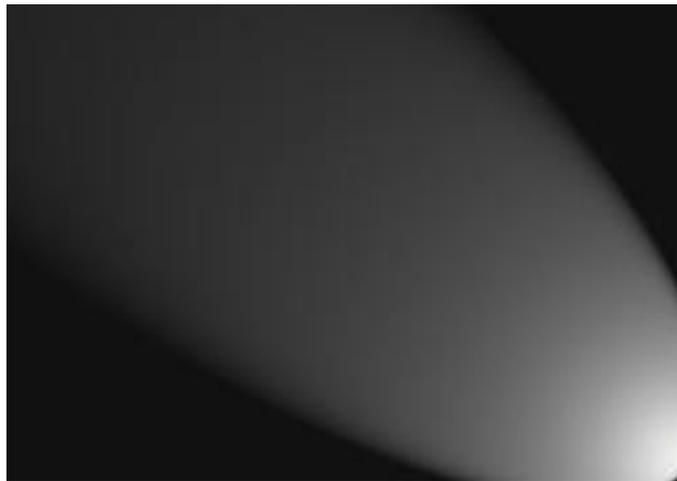


Figure 1. A standard light map

Once a light map has been generated for a given light source, it is generally saved as part of the application data. At runtime, the light map is combined with a standard texture map to produce the appearance of luminance. *Figure 2* shows a standard texture map before this step has occurred.

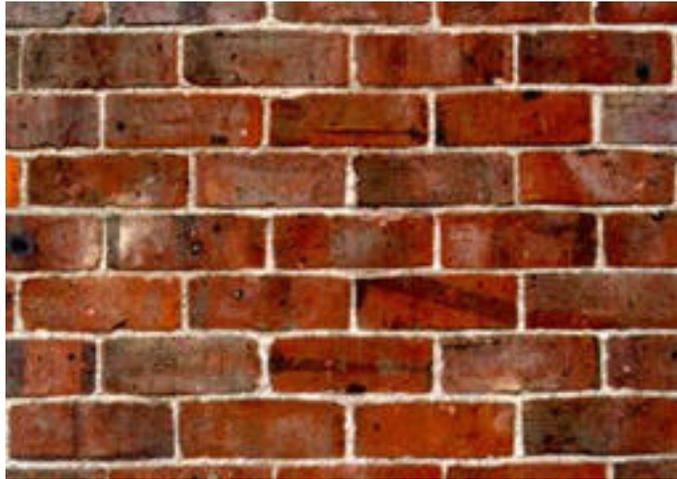


Figure 2. A standard texture map



Figure 3. Light map combined with texture map

In *Figure 3*, the texture map and its corresponding light map have been combined to achieve a dramatic spot light that appears to illuminate the underlying base texture. The process of combining one or more textures to produce this type of effect is known as *multitexturing*. Modern graphics processing hardware can perform multitexturing effects in a single pass, making it possible to achieve smooth, high-quality lighting with light maps while maintaining high performance levels. But unfortunately, the performance and quality advantages of light maps come at the cost of flexibility.

A programmer or designer must generate light maps at the same time the 3D scene is authored. Unless a light map exists for a given lighting condition, it cannot occur in a scene without first completely recalculating the light map in real time. Computing a new light map programmatically for each frame in a scene would be prohibitively expensive, even on modern graphics processors. The fact that light mapping cannot calculate dynamic lighting events without incurring significant performance penalties is a serious shortcoming of the technique.

Vertex Lighting

Vertex lighting, an alternative method to light maps, computes lighting values on-the-fly for each polygon in a scene. All real-time 3D graphics are composed of triangular polygons that consist of three vertices. Each vertex contains information about its position, color, lighting, texture, and other parameters that, together, can be used to construct an image. Per-vertex lighting calculations determine the level of light for every vertex of every object in a scene and then interpolate those values across the face of each triangle to produce its color as seen in *Figure 4*.

Light maps provide high-quality lighting effects because they are bit maps and are applied at the pixel level. Vertex lighting applies its effects at the triangle level, which is a much coarser-grained working unit. As a result, unless a model consists of a large number of triangles, per-vertex lighting will result in a faceted, coarse-grained appearance.

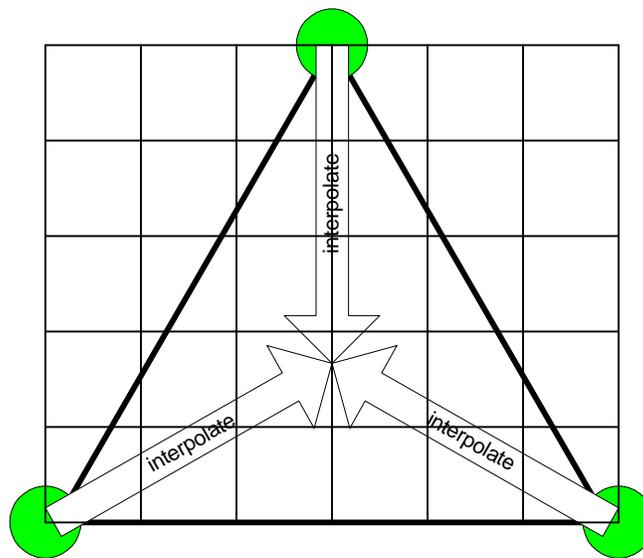


Figure 4. Interpolating vertices to determine color values for a triangle

One way to overcome this perceptible defect is to increase the number of triangles in a scene and by doing so decrease their size. The process of raising the number of triangles (and vertices) in a model is known as *increasing tessellation*. A highly-tessellated model, when compared to a simpler representation of the same object, contains much more information and requires more computational resources to render. But under a vertex lighting scheme, the additional vertices will enable much smoother lighting effects.

The following figures illustrate this concept with a simple scenario involving a single quad illuminated by several hardware lights. *Figure 5* represents a quad that is composed of fewer and hence larger triangles. The point lights reflected on its surface exhibit distinct and obvious faceting. Each light should have a smooth, circular perimeter, a bright center, and gradual fall off at its edges. But vertex lighting, in this case, produces scarcely recognizable blotches of color.

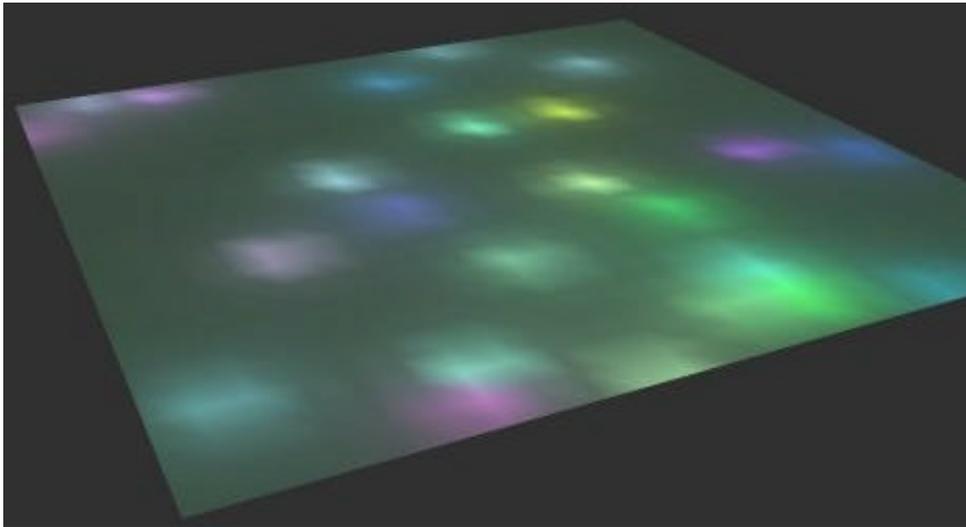


Figure 5. Vertex lighting of a quad with insufficient tessellation

Figure 6 utilizes exactly the same vertex lighting scheme, but this time applies it to a highly tessellated quad. The extremely large number and therefore small size of the triangles in this scene makes it possible to achieve high-quality vertex lighting results with imperceptible faceting and accurate fall off.

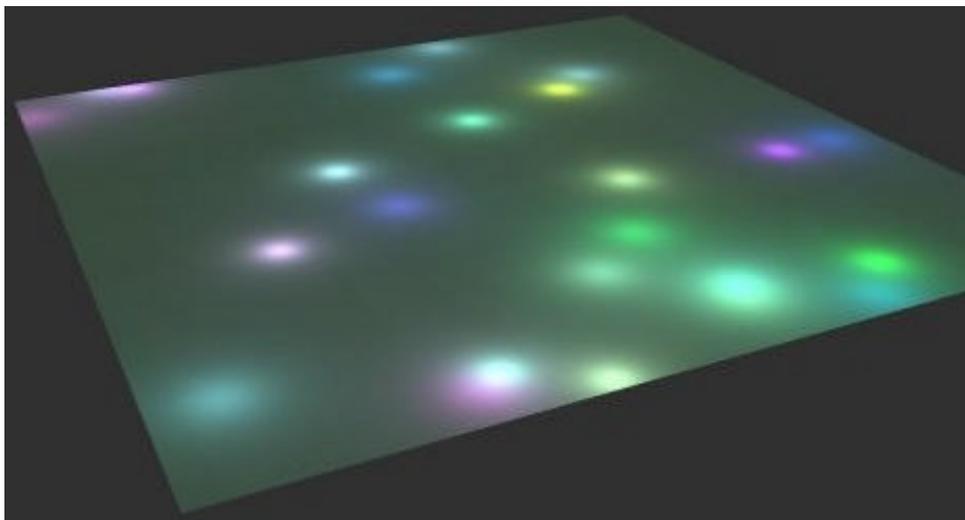


Figure 6. Vertex lighting of a very densely tessellated quad

Comparison of Light Maps and Vertex Lighting

Vertex lighting is a flexible, dynamic way of calculating light values for a 3D scene. Unlike light maps, this scheme does not require artwork to be produced in advance for every possible lighting scenario. And unlike light maps, it does not require multitexturing to produce sophisticated effects. But unfortunately, vertex lighting is of limited use in some scenarios because its base unit is the triangle. In situations where dense tessellation is undesirable, light maps provide higher quality light effects because they are bit-mapped textures and provide pixel-level accuracy.

Developers have pushed the limits of both lighting schemes in an attempt to strike a difficult balance between quality, flexibility, and performance. An ideal lighting technique would combine the calculative, dynamic nature of vertex lighting with the pixel-level accuracy of light maps.

Per-Pixel Lighting with the NVIDIA Shading Rasterizer

Real-time per-pixel lighting is a revolutionary technology made possible for the first time by the NVIDIA Shading Rasterizer (NSR), an architectural feature of the Quadro2 family. Combining the best aspects of both light maps and vertex lighting, dynamic per-pixel lighting frees developers from the restrictions of other lighting schemes and provides them with a sophisticated palette of effects including per-pixel bump mapping.

As mentioned earlier, triangle vertices in a 3D scene carry information about their position and color that vertex lighting equations manipulate to assign color values to triangles. Vertices also contain a direction normal that describes their orientation relative to a light source. Having this same information for each pixel would enable the same type of real-time lighting calculation but without the overhead of interpolating those values across a triangle.

Textured 3D graphics are said to be composed of primitives called texture elements (texels). Like triangle vertices, texels contain information about their color and position and are commonly used for shading techniques including light maps. The innovation of per-pixel lighting is that it utilizes texels as a component of a lighting equation.

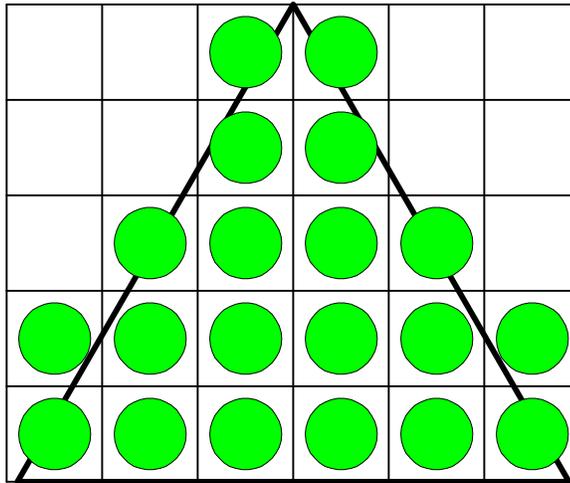


Figure 7. Light characteristics are calculated for each pixel of a triangle rather than being interpolated

To obtain the final piece of necessary information for per-pixel lighting calculations — the direction normal — developers must create a special type of texture called a normal map. Normal maps contain per-pixel normals that, together with texel information, are used to calculate the intensity and direction of light for each rendered pixel.

The NSR provides hardware support through its register combiner functionality for the texture blending operation called a dot product that makes per-pixel calculations possible. Available to developers in the Microsoft® DirectX®6 API and above (D3DTOP_DOTPRODUCT3) as well as OpenGL® (via the NV_register_combiners extension), dot-product texture-blending operations allow diffuse, specular, spot, and point light effects to be calculated dynamically on a per-pixel basis.

Because dot-product operations permit lighting to be determined for each pixel, they also enable dramatic bump mapping effects. Per-pixel bump mapping is a texture-based rendering technique for simulating lighting effects caused by irregularities on the surface of an object. By encoding surface patterns in texture maps, bump mapping simulates a surface's irregular lighting appearance without requiring that the perturbations actually be modeled in geometry. This approach avoids unnecessary object tessellation and improves performance while allowing for a dramatic increase in visual detail.

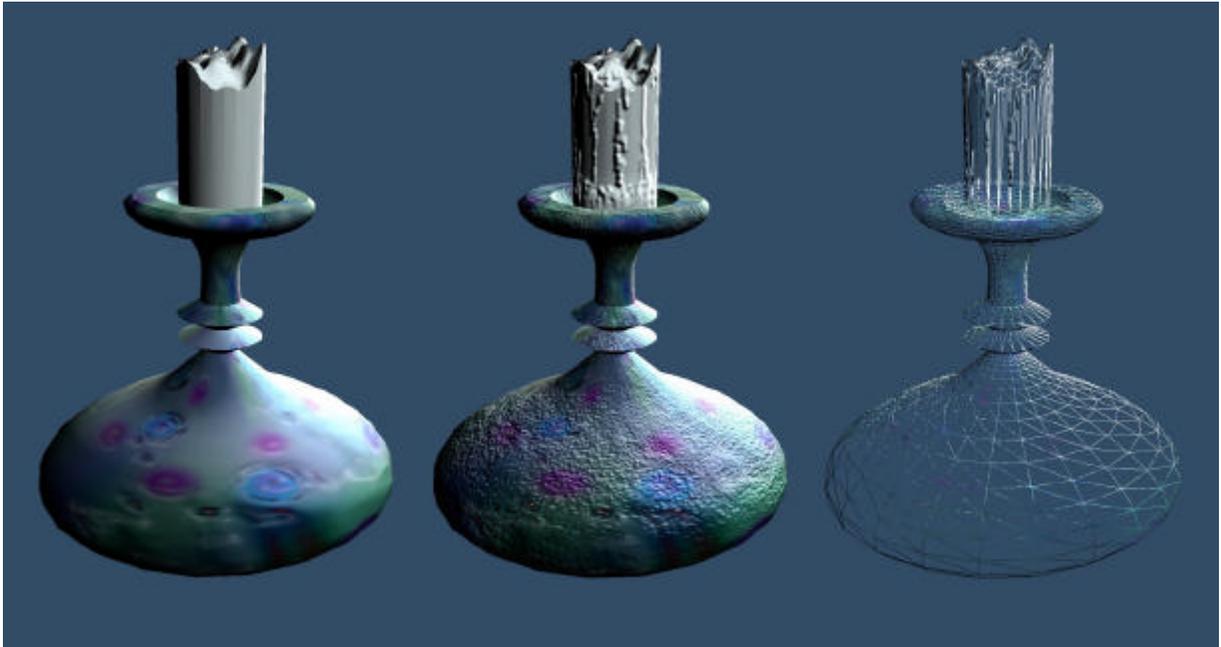


Figure 8. Per-pixel bump mapping increases visual detail without increasing geometry

Conclusion

The per-pixel shading capabilities of the Quadro2 family of workstation solutions, build on the foundation of hardware-accelerated vertex lighting integrated into all of NVIDIA's GPUs, starting with the original GeForce 256 and Quadro GPUs. By incorporating real-time support for the complex texture blending techniques of dot product operations, NVIDIA workstation products enable programmers, artists and engineers to design and deploy their work on a single platform, eliminating the wasted time of the edit/render cycles of offline rendering that interrupt work flow and creativity. The NSR architecture provides a radical new dimension to a variety of professional design spaces, for the first time enabling a near off-line visual quality to be utilized in real-time, enabling a new style of work, and dramatic increases in productivity.

© Copyright 2000, NVIDIA, the NVIDIA logo, GeForce, GeForce2 MX, GeForce2 GTS, GeForce 256, Quadro2, NVIDIA Quadro2, Quadro2 Pro, Quadro2 MXR, Quadro, NVIDIA Quadro, Vanta, NVIDIA Vanta, TNT2, NVIDIA TNT2, TNT, NVIDIA TNT, NVIDIA RIVA, RIVA, NVIDIA RIVA 128ZX, and NVIDIA RIVA 128 are registered trademarks or trademarks of NVIDIA Corporation in the United States and/or other countries. Other company and product names may be trademarks or registered trademarks of the respective owners with which they are associated.