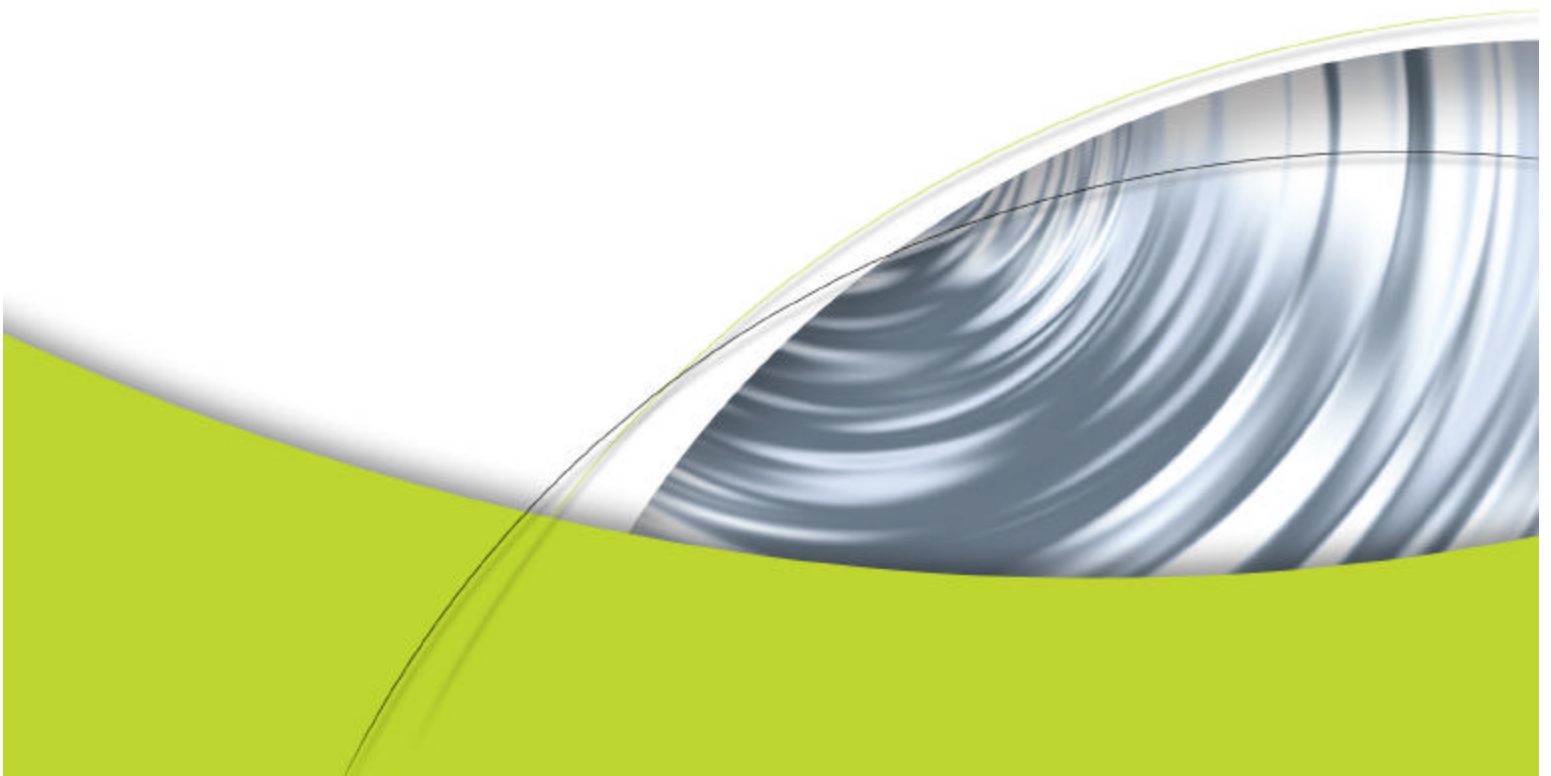




Technical Brief

Integrating the Cg Language into the Professional Workflow



Integrating the Cg Language into the Professional Workflow: Revolutionizing Productivity

"The joint introduction of NVIDIA's Quadro FX and GPU rendering within SolidWorks is revolutionary. There was a time when designers debated the value of 3D over 2D; then it was solids over wireframe; shaded images over black and white; then colored solids over greyscale. We have consistently seen that with visualization breakthroughs, the designers' productivity and satisfaction increase. Partnering with NVIDIA allows SolidWorks users worldwide to take advantage of this joint breakthrough."

Ilya Mirman, vice president of marketing at SolidWorks.

The level of realism possible with any particular computer-aided design (CAD), computer-aided manufacturing (CAM), or digital content creation (DCC) application directly affects the productivity of each user. Standard application programming interfaces (APIs) such as Microsoft® DirectX® and OpenGL® provide libraries of common graphics operations that allow developers to incorporate many more realistic effects into their applications. But, to incorporate accurate and therefore believable material and lighting programs, frequently referred to as "shaders," graphics application developers have to work with low-level assembly programming that is both unproductive and limited in scope. By integrating high-level graphics programmability, application vendors can give workstation users significantly increased levels of realism resulting in enhanced overall work efficiency.

Today, NVIDIA offers Cg programming language solutions, including the NVIDIA Cg toolkit and NVIDIA Cg Compiler, as the next step in the evolution of graphics programming. The Cg language, the "C" for graphics, represents a major leap forward in ease and speed of programming for real-time, life-like graphics. The common, familiar syntax enables rapid development of stunning, real-time effects for both DirectX and OpenGL applications.

Cg makes it possible for CAD/CAM and DCC application users to gain:

- Faster access to new hardware and software technology
- Significantly higher quality and more realistic graphics
- More innovative and versatile software solutions
- Easier and more frequent upgrades to popular software solutions

This paper describes the NVIDIA Cg Language solutions and their relevance to professional workstation users.

Benefits of Cg for Workstation Professionals

Computerized visual effects involve the manipulation of display data. Shaders—the programs and parameters necessary to implement an effect on a set of pixels or vertices—traditionally target a specific graphics platform. Cg provides a language and constructs for creating platform-independent shaders (see the appendix, “Overview of the Cg Language and NVIDIA Cg Toolkit”). Cg greatly simplifies shader programming and shortens the amount of code required for even the most complex graphics effects. The benefits for workstation users include the adoption and proliferation of many libraries of shaders to represent advanced materials, surfaces, and finishes. Software applications that take advantage of Cg can allow creative and technical professionals to easily use and apply shaders, enabling users to better create, analyze, and distribute unique models and digital content. Similarly, higher-quality content can be created in a significantly shorter amount of time; Cg yields more adjustable applications tools so that users spend less time adjusting the situation to the tool instead of adjusting the tool to the situation. Cg also makes it easier to repurpose content, allowing designers to share digital content and models with other users in the organization. For example, if an engineering image is created using Cg-based effects, users can cost-effectively leverage the image into marketing materials and company Web content.

Benefits Specific to DCC Applications

DCC developers always push the limits of graphics technology. Success depends on generating the most realism. The Cg language helps DCC application developers produce tools and solutions that do just that. The excellent code efficiency of the Cg language and built-in optimizations of the NVIDIA Cg Compiler maximize performance and speed the development of numerous and varied cinematic effects.

The Cg file format also provides a big plus for DCC users. The ability to transfer shader effects between applications, while maintaining access to selectable parameters, greatly simplifies the work required by artists and directors throughout the entire creation workflow to create just the right effect for a particular project.

Other benefits that the Cg language offers to DCC workstation professionals include:

- **Greater empowerment with flexibility:** Cg allows more sophisticated tools to be created for DCC applications ranging from 2D compositing to 3D scene creation.
- **Platform and API independence:** Effects created using the Cg language are independent of graphics APIs and hardware platforms.
- **Effect sharing and reuse:** Capturing and transferring sophisticated shader effects between all stages of the workflow affords significant productivity gains. Cg enables shader repositories.
- **Fast effects previewing:** Integrating Cg real-time rendering capabilities into DCC workstation applications removes the need to wait for off-line rendered views, greatly improving productivity.

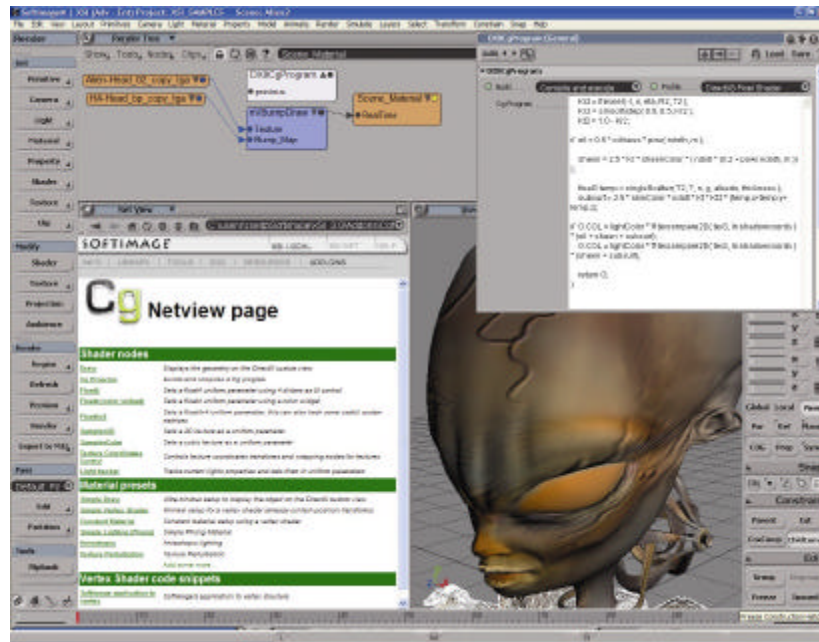


Image courtesy of AviD technology Inc.

Figure 1 - Integration of Cg within Softimage

Benefits Specific to CAD Applications

CAD applications involve an immense amount of existing data and significant amounts of legacy code. Protecting this investment makes developers reluctant to introduce leading-edge visual capabilities into existing CAD software. The Cg language, however, provides an opportunity to introduce significantly more advanced materials and surface finishes into CAD visualizations without requiring applications to be rewritten and without restricting vendors to one hardware platform. The Cg file format also allows these more realistic materials and finishes to be easily interfaced with other programs. Re-purposing CAD data opens up significant benefits in both capability and productivity.

Other benefits that the Cg language brings to CAD workstation professionals include:

- **Increased realism:** Product styling designers appreciate the more realistic looking materials and metals that can be rendered in real time with Cg.
- **On-the-fly iterations:** Cg takes full advantage of graphics hardware to deliver complex visual effects on the fly. CAD users can quickly iterate complex designs, and avoid lengthy delays from offline rendering.
- **Smooth migration of existing code:** Current libraries of materials can be converted to Cg shaders over time, ensuring standardization and portability between platforms.

- **Investment protection and leveraging hardware:** Shaders developed using Cg remain independent of both hardware and APIs. When executed, Cg programs take maximum advantage of the hardware available, including any new upgrades.

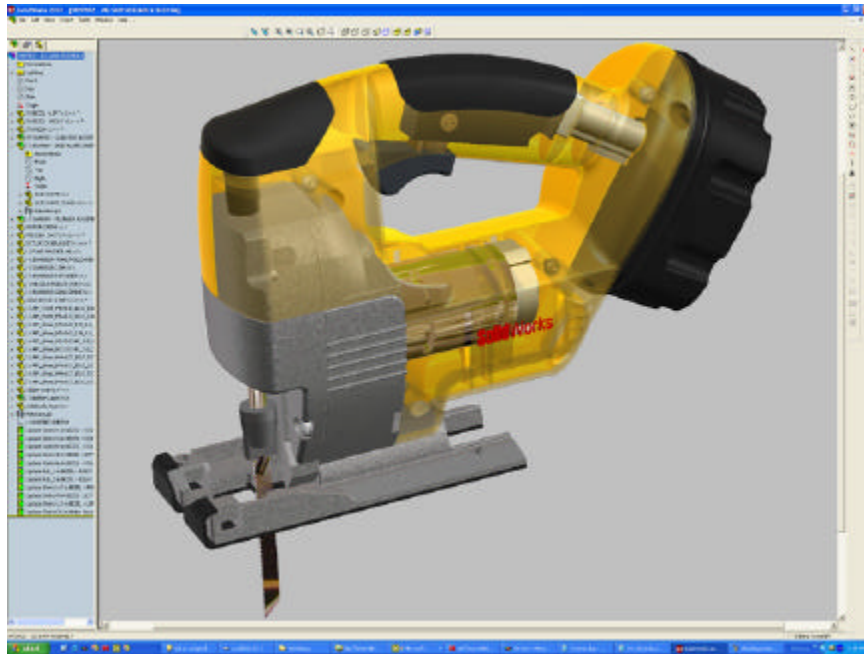


Image courtesy of SolidWorks Corporation

Figure 2 - Demonstration of Cg integrated within Solidworks

Fundamentally Changing the Creative Workflow

Cg greatly contributes to the availability of real-time rendering solutions. Since more complex images can be generated interactively, the workflow model for CAD/CAM and DCC user bases dramatically changes. User productivity increases in proportion to the proliferation of real-time rendering. With the simplified programming language, users will see more effects and more realistic graphics in popular software solutions.

From a practical standpoint, Cg enables technical and creative professionals to enjoy more forward-compatible applications. When written in Cg, programs automatically benefit from the addition of a new graphics solution when it becomes available—without recompiling or upgrading the software. The high-level language protects software investments over longer periods of time while providing excellent performance from hardware advances.

Conclusion

Visualization applications that leverage Cg shader technology will get the most out of desktop hardware and enable the most stunning visual effects for enhanced realism. As workstation application developers embrace the Cg language, users will see an increase in the number of application plug-ins available, and in the quality of the visual effects that are merged into the existing applications. All of this accelerates the move towards making cinematic experiences a reality on the desktop.

Digital content creation (DCC) applications have already embedded Cg into the digital artist's workflow, making shader technology friendlier and more accessible to creative professionals and artists. Offline rendering application developers are also integrating the NVIDIA Cg Compiler into future products, allowing them to accelerate and enhance their image generation capabilities. In conjunction with this, there is a significant movement of artists towards the Cg language.

APPENDIX – Overview of the Cg Language and NVIDIA Cg Toolkit

NVIDIA offers a complete NVIDIA® Cg Toolkit to application developers. This toolkit allows application developers access to compiler technology and the other tools that streamline the development of graphics-intensive applications. The current toolkit for writing graphics programs, called *shaders*, includes:

- The Cg language specification
- NVIDIA Cg Compiler version 1.0 that supports DirectX 8.1 vertex and pixel shaders, DirectX 9.0 vertex and pixel shaders, OpenGL 1.4 (with ARB_vertex_program)
- NVIDIA's Cg Effects Browser including sample shaders
- The *NVIDIA Cg Toolkit User's Manual*

Future releases of the NVIDIA Cg Compiler will integrate support for later versions of DirectX and OpenGL as they become available.

Common Syntax and Format

The industry-standard Cg graphics language, developed by NVIDIA, serves as a very general C-like language specification for shader design. With support for current and future OpenGL and DirectX APIs, Cg gives application developers the ability to create one application that supports these different APIs and that will run on many varied target systems.

The Cg language standard also includes the specification of a standard file format for shader programs. Cg-enabled files make it possible to transfer shaders between different applications, catering for the needs of the professional workstation as they move data through their workflow. Users will gain flexible interfaces and run-time parameter settings for more customizable application execution. The Cg standard and file format supports:

- User interface (UI) specifications (enables editors and digital content creation applications to present appropriate UIs so that artists and creative professionals can more easily work with shaders)
- Common parameters that can be adjusted at run time (for a more customizable solution and tools that can be adapted to unique situations)
- Texture types required by the shader (for more realistic depictions of materials and objects)
- Vertex information (for more accurate geometry operations)

Unified Compiler Architecture

The NVIDIA Cg Compiler provides developers the benefits of a Unified Compiler Architecture (UCA), similar to the NVIDIA Unified Driver Architecture.

Developers can write a single Cg version of a shader and use the NVIDIA compiler to produce executables targeted to multiple platforms (NVIDIA and non-NVIDIA GPUs). The NVIDIA compiler allows developers to produce either DirectX (8.0 or later) or OpenGL (1.3 or later) executables. The forward and backward compatibility simplifies the design, production, and management of all shaders, minimizing or eliminating the need for code changes to support new or additional platforms.

As a runtime compiler, UCA automatically takes advantage of the hardware that is available at the time that the shader application is executed. Application performance is also enhanced since the NVIDIA Cg Compiler includes optimizations for NVIDIA GPUs, making it possible to get the best performance out of the hardware. Applications become better able to support a broader range of hardware platforms and to get the best performance out of each.



Notice

ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE.

Information furnished is believed to be accurate and reliable. However, NVIDIA Corporation assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties that may result from its use. No license is granted by implication or otherwise under any patent or patent rights of NVIDIA Corporation. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. NVIDIA Corporation products are not authorized for use as critical components in life support devices or systems without express written approval of NVIDIA Corporation.

NVIDIA, the NVIDIA logo, NVIDIA Quadro, and the NVIDIA Quadro logo are registered trademarks or trademarks of NVIDIA Corporation.

Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright © NVIDIA Corporation 2003.



NVIDIA.

NVIDIA Corporation
2701 San Tomas Expressway
Santa Clara, CA 95050
www.nvidia.com