

Jonathan Ragan-Kelley
P.O. Box 12290
Stanford, CA 94309
katokop1@stanford.edu

May 7, 2003

Undergraduate Honors Program Committee
Stanford University Department of Computer Science
353 Serra Mall
Stanford, CA 94305

To Whom It May Concern:

For consideration of departmental honors in Computer Science, I propose to pursue a solution to the problem of lighting design in computer graphics for feature film. I propose to approach this problem by using a combination of automatic computational specialization and interactive rendering on programmable graphics hardware. In the process I hope to profile the computational flow of feature film Renderman shaders and the implications on mapping cinematic rendering to graphics hardware.

The RenderMan¹-centered production pipeline of the modern special effects or digital animation studio culminates in the work of the technical directors (TDs) creating the final "look" of each scene through lighting and shader design. The work of the TD mirrors that of the gaffer on a conventional film set, positioning and fine-tuning hundreds of lights to perfect the illusion of complex natural or artificial illumination called for by a given scene. Despite the rapid development of interactive computer graphics in recent years, the look design tasks of the TD remain completely non-accelerated and non-interactive. Where a modeler or animator

might tweak control vertex or motion keyframe parameters and receive instant feedback in an OpenGL-accelerated view port, a TD is forced to wait tens of minutes for feedback because even a single change requires executing the entire final-frame software rendering pipeline over again.

While software rendering systems can be optimized for such work,² they are still far from interactive when bearing the full complexity of film scenes – scenes complex enough to routinely require many hours to render a single frame on state of the art computers. Furthermore, software rendering systems inherently fail to take advantage of the rapid development of hardware graphics systems, since these chips still are not flexible enough to render complex film scenes. Even if a reasonably high-quality rendering can be created with a modern graphics processor, the look design process is inherently dependent upon the intricacies of the specific graphics pipeline used in the final renderings. Therefore, any practical interactive look design tool must replicate the exact results of the software rendering pipeline used for final rendering. Thus, interactive film-quality RenderMan output from non-optimized film scenes is not expected for the indefinite future. Because of these challenges, Pixar and others have expressed strong interest in research into the application of graphics hardware to interactive lighting design in complex RenderMan scenes.

Gershbein and Hanrahan presented the first reasonable approach to interactive lighting design for film production in 2000.³ Their work leveraged the critical observation that lighting design is generally an iterative process of fine-tuning lighting parameters from a single, fixed viewpoint. Therefore, the majority of the computation to render after each tweak is redundant, never varying from one preview to the next. They exploit this by carefully factoring the rendering process, caching common intermediate data in a screen space deep framebuffer, and only recomputing the final shading of each pixel from the light-dependent portions of the shading computations and the cached intermediate values. Using this approach, they achieve interactive rates in moderately complex scenes.

However, in spite of its promise, Gershbein and Hanrahan's deep framebuffer approach has faced significant challenge in adoption. Because the approach relies on the use

¹ [<http://renderman.pixar.com>] RenderMan is a flexible and high quality rendering architecture designed by Pixar and in widespread use throughout high-end effects and animation production.

² Pixar's relatively young Irma tool professes to accelerate the look design process by caching a large amount of the intermediate rendering work in a given frame so that less needs to be recomputed for each small parameter change.

³ R. GERSHBEIN, AND P. HANRAHAN. *A Fast Relighting Engine for Interactive Cinematic Lighting Design*. SIGGRAPH '00, p. 353-358.

of a small set of predefined materials, it is difficult to apply in production, where RenderMan is used specifically because it allows artists to define complex, custom material and light models through the use of “shaders.” Established studios and artists possess a large legacy of existing shaders, as well as tools, workflow, and skills invested in RenderMan and the RenderMan Shading Language, and the prospect of transforming an entire production pipeline to support an interactive preview tool for lighting has proved unappealing. Furthermore, because Gershbein and Hanrahan’s system was not targeted to programmable graphics hardware, there are significant limitations to the complexity of the materials for which lighting can be visualized efficiently and interactively, even if complex new material models were programmed in tandem into the lighting design system for interactive preview, and into RenderMan shaders for final rendering.

In late 2001, Ujval Kapasi and I described an approach combining the efficient caching of the deep framebuffer system with automatic specialization of RenderMan shaders for lighting-dependent computations, and cross-compilation of these specialized shaders to programmable graphics hardware.⁴ We demonstrated the basic viability of the approach through an early prototype running on the first generation of moderately programmable graphics hardware.

The approach to the lighting design problem I propose pursuing is a direct evolution of that described in 2001. In it, I will break the lighting design process into four stages. First, a Renderman Shading Language compiler employs dataflow analysis to specialize the shader computation to only the light-dependent variables in the system. Second, the light-dependent portion of the shader computation is mapped to programmable graphics hardware by cross-compilation to the Cg language. Third, a deep framebuffer of light-independent intermediate values is generated for the shaders. Finally, an interactive preview application executes the Cg computation against the intermediates cached in the deep framebuffer and the light variables as they are configured and refined by the user. Because the cached intermediates and cross-compiled shading code correspond exactly to the computation performed for each pixel in the RenderMan scene, the interactive preview exactly matches the final output of the software rendering system.

While prior work has outlined my core approach and demonstrated its initial viability, the prototype was extremely simple – featuring at best minimal specialization and cross-

compilation – and was severely limited in its scope by the woefully limited generality of contemporary graphics hardware. Limited by the simplicity of contemporary hardware, we were unable to demonstrate anything more than simple surfaces which could be easily factored and mapped to hardware by their sheer lack of complexity. With today’s graphics hardware only recently becoming sufficiently general to support lighting design in scenes of realistic complexity, the full viability of such an approach has yet to be explored. Critical hypotheses posited during and following my initial research lack experimental demonstration or exploration. Furthermore, myriad interesting challenges remain in implementing a fully functional specializing cross-compiler for mapping RenderMan Shading Language to graphics hardware. The RenderMan Shading Language, especially as used in modern production environments, is a multifarious and complicated system, possessing many critical language constructs which cannot be mapped even to proposed future graphics hardware, and many more which cannot maintain efficiency if mapped directly.

Thus, beyond the major implementation entailed by this project, I believe that this space still has tremendous room for exploration, which I intend to pursue as my primary focus. I will demonstrate and explore successes and failures of my approach to lighting design, and its implementation, with real-world shaders in film-complexity scenes. Further, I will explore the many challenges of mapping from the RenderMan Shading Language to modern graphics hardware: Is most light-dependent computation frequently only a simplified subset of the RenderMan Shading Language that can be mapped directly to graphics hardware, by the nature of common realistic lighting calculations? What compiler, specialization, and mapping tricks can be employed to enable mapping a greater subset of the RenderMan Shading Language to hardware, or doing so efficiently? How critical are these more complex features to enabling interactive lighting of a significant majority of real-world RenderMan scenes? In general, what are the critical paths of most real-world shading computations? How do these map to hardware graphics pipelines? What implications do these present for future pipeline enhancements to enable film-like rendering?

My background in this research stems not only from my own prior work in the area, but also from an extensive background in computer graphics and digital effects predating my matriculation to Stanford. While here, I have excelled in the Stanford graphics curriculum, including CS248, CS348B, and CS448A (Fall 2001, on real-time graphics hardware

⁴ U. Kapasi and J. Ragan-Kelley. *A Hardware Accelerated Relighting Engine for Renderman*. Final project for

architecture). I have long maintained a connection with the Stanford Graphics Lab, participating in weekly research group meetings consistently since February 2001, in my freshman year, working as a full-time research assistant in the summer of 2001, conducting independent research throughout the 2001-2002 year, and beginning again earlier this year. Finally, during summer 2002 I researched the application of film rendering methods to future graphics hardware while working in a small independent group at NVIDIA to integrate the advanced rendering expertise of Pixar R&D-spin-off Exluna into future NVIDIA technologies.

Thank you for your consideration, and I greatly look forward to the opportunity to focus my intellectual pursuits through this program of research.

Respectfully,

Jonathan Ragan-Kelley