SOLUTION BRIEF
Intel® Xeon® Processor-based
Workstations for Interactive
and Final 3D Rendering

No-Holds-Barred Production Rendering

Expert Perspectives on Using Multi-Core, Multi-Threaded Intel® CPUs for Speed, Complexity, and Quality

Credit: Stuart Rowbottom

Final rendering composited on top of background plate. Inset picture shows virtual "clay" version of rendering prior to the application of textures and final lighting. Image rendered in Luxology modo.*

Who should read this document: *Production Designers, Art Directors, Creative Directors, Visual Effects Supervisors, CG Supervisors, Rendering TDs, Industrial Designers, CAD and CG Artists and Designers, and ISVs.*

The computer-generated imagery that pervades modern life is more sophisticated than ever. Blockbuster animated and visual effects-driven films take audiences to fantastic places by capitalizing on technologies that make it possible to render scenes, special effects, and virtual characters with breathtaking realism. Many of those same technologies are used by product designers, architects, and civil engineers to visualize prototypes as well as architectural and infrastructure designs with outstanding visual precision. The ever-increasing capabilities and performance of multi-core, multi-threaded processing, driven by Intel® Core™ microarchitecture advances, provide an optimal foundation for a wealth of software innovations and rendering improvements.

This document examines the advantages of using modern CPU technology for advanced offline production rendering, yielding exceptional image complexity and quality, as well as providing the flexibility and convenience of running sophisticated content creation software systems on a mature, high-performance computing platform.

Production Challenges

The perennial goal of creative professionals working on animation and special effects for motion pictures and television—as well as CAD professionals—is to deliver top-notch work on time and under budget. For the past 40 years, computing power has been increasing steadily in keeping with Moore's law, which says that transistor density will double approximately every two years. While production times might be expected to decrease in direct proportion to available processing power, the ambitions of creative professionals are outpacing Moore's law, ensuring that there is no shortage of ever-more complex rendering algorithms that deliver increasing levels of visual subtlety and devour advances in processing power as quickly as they become available.


Credit: www.7-t.co.uk

Building on a pier (architectural design rendered with Maxwell Render*).

A casual look at some popular CG- and visual effects-intensive films released over the last 10 years illustrates the point. For example, the production rendering for Peter Jackson's *Lord of the Rings* trilogy (2001, 2002, 2003) combined live action and highly realistic virtual characters. As production on the three films progressed, WETA continually upgraded its compute power to keep up with the ambitions of the filmmakers—by the end of the project, the render farm consisted of 3,200 processors.[1] It reportedly took about four hours to render a single frame of Gollum[2] and up to 48 hours per frame to render Treebeard.[3] *Ratatouille*, released by Pixar Animation Studios in 2007, was rendered on an arsenal of approximately 850 Intel® processor-based servers, which housed nearly 3,200 processors. The average rendering time for each frame of animation was roughly 6.5 hours.[4] James Cameron's *Avatar*, released in 2009, features animation and effects produced by WETA using a server farm of some 4,000 HP BL2x220c* blades hosting about 40,000 Intel processor cores. Rendering times ranged from 30 to 100 hours per frame.[5]

In the design visualization world, no detail is too small to leave to the imagination, and the demand for high-quality visualization continues to grow in direct proportion to client expectations. The ability to render highly precise, photorealistic product designs and large-scale architectural plans, including exterior and interior walkthroughs, helps speed time to market. Additionally, CAD firms can leverage the same high-resolution imagery for marketing purposes.

In contrast to high-end film production rendering, the focus for CAD professionals is less on large-scale, team-based workflows and pipelines that culminate in massive-scale high-performance computing (HPC) render farms, but more on creative individuals and small workgroups. Here too, advances in Intel® multi-core CPU architectures bring the benefits of distributed rendering to individual workstations.

Rendering at a Glance

Most popular rendering methods—rasterization, ray tracing, ray casting, and so on—simulate light moving through a scene (aka light transport) to produce photorealistic imagery or stylized non-photorealistic looks. At a basic level, renderers generally compute the visible geometry in a scene as triangles—and paint them onto the virtual screen. Color and surface properties are computed by shaders, which commonly reference or sample texture maps. Final color is calculated based on the amount of light emanating from the light sources in a scene and how much strikes a given point on each triangle, and finally how much of that light is reflected back towards the viewer.

Modern production rendering, however, is anything but basic—motion blur, depth of field, deep shadows, subsurface scattering, color bleeding, and caustics are just some of the myriad optical effects that can be simulated using one or more rendering methods.

Credit: Bo Hansen

Photorealistic soda cans, ray-trace rendered in Luxology modo.* Notice the depth-of-field effect.

Whether the goal is to achieve physical accuracy for things like medical imaging or product design, or to enhance reality for the sake of getting audiences to suspend disbelief as a story unfolds onscreen—the level of visual complexity is usually a balance between time, budget, and hardware resources. Greater complexity usually equates with larger file sizes and heavier throughput computing workloads, because many rendering algorithms perform best when the entire scene file fits into physical memory, and main-memory-to-internal-memory transfers represent a critical performance bottleneck.

Raster is Faster, but is it Enough?

Historically, rasterization or scanline rendering has been the go-to method used to render scenes fast enough to meet demanding production schedules. Pixar RenderMan,* the dominant production-level renderer for films, began life as a raster-based rendering engine, but it has grown into a hybrid engine that offers a variety of ray tracing, global illumination, and point-based algorithms, as well as techniques based on the Reyes ("renders everything you ever saw") algorithm, an advanced form of rasterization.

Ray-Traced Realism

When the primary goal is photorealistic, physically accurate imagery, the go-to rendering technique is usually *ray tracing. Ray tracing* renders scenes by tracking the path of individual rays of light illuminating a scene. Ray tracing can be used to handle virtually any type of optical effect, including reflection, refraction, shadows, caustics, volumetric lighting, motion blur, and much more. Every ray can have a unique origin and a unique direction, and various intricate effects can be computed along each ray, providing incredible flexibility in rendering high-quality, photorealistic images.

Ray tracing, however, is unavoidably highly computation-intensive. Rays passing through highly detailed scenes can number in the tens of millions. As more effects are applied, the number of rays skyrockets. Advances in multi-core, multi-threaded CPU architectures, however, provide the compute power to track billions of simulated photons through a scene and the simulated light penetrating the scene's surfaces to achieve the highly convincing photorealistic images. Today's 3D software, combined with the power of Intel® processors are delivering blazing performance in both offline and interactive rendering scenarios.

Advances in multi-core, multi-threaded CPU architectures provide the compute power to track billions of simulated photons through a scene and the simulated light penetrating the scene's surfaces to achieve the highly convincing photorealistic images.

Credit: Courtesy Luxology

This ray-traced scene, complete with a procedurally rendered sky, virtual grass, and a detailed, textured model of a cherry picker rendered in 2 minutes, 2 seconds in modo* 501 due to recent CPU optimizations made by Luxology.

Interactivity

Applications such as flight simulators and computer games rely on raster techniques to render images at frame rates high enough to interact with in real time. Ingenious programming methods have been developed that allow raster images to include effects that would otherwise be too computationally intensive to reproduce in real time. For example, volumetric lighting effects can be baked into texture maps that can be sampled on the fly by game engines. A variety of techniques can be used to improve raster engine performance including, for example, backface culling. Hardware acceleration for rendering, as seen in recent PC video games, is available through graphics APIs such as OpenGL* and DirectX.*

Game developers often optimize their code for speed using tools such as the Intel® Graphics Performance Analyzers (GPAs) to detect performance hotspots. Once hotspots are identified, different techniques can be used to tune the complexity of the graphics processing tasks, so the result is optimal performance with no loss of the intended visual impact.

Hollywood productions also look for ways to streamline rendering performance, but the many data sets are simply too massive and the rendering goals far too complex to handle with anything other than modern CPU-based offline rendering pipelines.

Real-time raster techniques are often used for preview rendering in production pipelines. While real-time performance is often associated with special-purpose graphics acceleration hardware, thanks to Moore's Law, increases in CPU-based processing power deliver increasing degrees of realism and interactivity for both real-time game play and iterative preview rendering in production situations.

High-Performance Processors Meet the Challenge

The visual subtlety necessary to produce high-impact virtual images and effects that are indistinguishable from the real world requires a no-holds-barred approach to production rendering. This means being able to draw on any and all rendering techniques—including ray tracing and global illumination, ultra-large resolution texture maps, complex combinations of custom shaders written by in-house lighting TDs, and beyond. In this role, modern CPUs have no equal, thanks to a number of characteristics of the Intel® multi-core, multi-threaded processor architecture:

- **Access to large amounts of system memory.** Intel® 64 architecture† encompassed in Intel® Core™ i7 and 64-bit Intel® Xeon® processors eliminates the memory-addressing limitations of 32-bit processors and gives way to handling large data sets through support of up to 128 GB RAM on workstations and up to 1 Terabyte in server farms. The ability to address large amounts of system memory is essential to production rendering, because scenes are typically extremely large and advanced rendering algorithms such as ray tracing are best performed when the entire scene fits into physical memory.
- **Large caches.** Intel Core i7 and Xeon processors offer large, multi-level caches that benefit many rendering algorithms, in particular Monte Carlo ray tracing, global illumination, and ambient occlusion. An integrated memory controller enables three channels of DDR3 1333 MHz memory, resulting in up to 32 GB/sec memory bandwidth, speeding performance for production rendering.

Credit: Ryan Drue

This ray-traced rendering of cacti demonstrates how “replicator” functionality in the Luxology renderer can be used to simulate cactus needles.

Indeed, Intel CPUs have long been the cornerstone of production render farms. In addition, Intel CPUs typically run every software component in a 3D content creation pipeline, from the modeling and animation software to texture mapping tools and compositing systems, as well as special-purpose plug-ins and custom shaders. Other benefits of running your entire pipeline on Intel CPUs include:

- No need to partition the code-base of custom shaders into “core” and “accelerator” code (the latter, so it will run on third-party graphics accelerator hardware).
- Entire ecosystem of third-party software tools/libraries (such as OpenEXR* and so on) runs “out of the box.”
- No need to be concerned over whether next year’s rendering algorithms will map well—CPUs can run it all.
- No need to rewrite the entire code base just to bring it up on new hardware—although you still have the option to optimize the kernels that matter for a new microarchitecture.
- Writing custom shaders in-house is expensive. Being able to run legacy and new code on the same CPU nodes helps ensure that this investment can be amortized over many years going forward.

Seriously Fast CPU-based Ray Tracing

While artists and designers rarely have to think about the code that enables their digital content creation tools, they reap the rewards of well-crafted code optimized to take advantage of the breakthrough performance offered by Intel Xeon processor-based workstations and servers, as well as mobile workstations powered by Intel Core i7 mobile processors.

Luxology, developers of the award-winning modo* software is no stranger to writing highly efficient code that's optimized for maximum performance on Intel-based Windows* and Mac* computers. The innovative 3D modeling and rendering software has been delivering fast ray tracing-based rendering since 2006, when the modo offline renderer and preview renderers were introduced. The modo Preview Renderer delivered world-class photorealism in near real time running on previous-generation Intel CPUs and did not require any specialized graphics hardware.

Interactive preview rendering gives artists the ability to:

- Iterate designs faster, because there's no waiting around while work renders
- Reduce review and approval stages of production, thanks to the ability to see results continuously
- Eliminate uncertainty in final rendering results, since hundreds of test renderings are performed naturally as the scene is developed
- Shave valuable time off production cycles or simply reinvest that time in improving the quality of images

Running on current-generation Intel Core i7 and Xeon processors, the latest release of modo delivers dramatic performance that scales nearly linearly as more processors are added to the pipeline. (Each copy of modo is licensed to run on 50 render nodes and supports up to 32 processors per node.) The modo renderer supports instancing and procedurally generated geometry—displacement mapping, and so on—so that at render time it can handle trillion-polygon detail at enormous frame sizes, support 20K x 20K print resolution output, and provide an array of global illumination and physically based shading models. Reflection, refraction, anisotropic blurry reflections, caustics, dispersion, blurry refraction, and subsurface scattering are among the advanced optical phenomena that can be rendered.

Expert Perspective

Allen Hastings, Co-Founder and Chief Scientist, Luxology

How does ray tracing benefit from Intel multi-core, multi-threaded processors?

Hastings: "Because ray tracing involves looking at a vast number of individual rays as they travel through a scene, ray tracing lends itself to parallelism—something Intel multi-core, multi-threaded processors are very good at. Since every ray can be independent, you can divide up your scene into batches of rays. Typically, that translates to regions of the final image (often called tiles or buckets) and that yields fantastic scaling."

Do you take advantage of the advanced tools Intel offers to help developers optimize their code?

Hastings: "We handle our own cross-platform threading abstraction and use the Intel® Thread Profiler, which helps us identify bottlenecks in our code faster. We also use the Intel® Compiler for production."

You're able to achieve interactive ray tracing without the aid of specialized graphics hardware. How are you accomplishing that?

Hastings: "We're leveraging the cache and writing algorithms that are purpose-built for speed. The result is that modo delivers quantitatively accurate, interactive rendering that allows artists and designers to use ray tracing-based rendering continuously during the creative process, instead of only as the final step—a claim that some would have you believe is exclusive to GPU-based rendering. But most of the GPU demos I've seen are run on exotic—and expensive—graphics hardware and cleverly disguise limitations.

"Modern GPUs offer a brute force solution to ray tracing, but the memory available to GPUs is relatively limited compared to the system memory available to 64-bit CPUs such as Intel Core i7 and Xeon processors. That means that GPUs typically can't handle the huge scene files required in full-scale production rendering, which may involve tens of millions of polygons and hundreds of high-resolution texture maps. And CPUs offer greater flexibility in terms of shading complexity and plug-in shaders, which may or may not have been ported to run on a GPU."

modo supports stereoscopic 3D rendering. What kind of control over the stereo image do you offer users?

Hastings: "We provide control over the convergence distance and the inter-ocular distance. We also speed up the rendering of the second eye by reusing some of the shading information, for example the first eye's irradiance cache and photon map. Likewise, we can retain the irradiance cache if you're just moving the camera as you would in an architectural walkthrough. As soon as you move any object or change a color, the irradiance information needs to be recalculated. With light transport rendering techniques almost anything can affect everything else, unless it's locked in a sealed box."

Efficient CPU-based Path Tracing for Offline Production Rendering

The "Arnold*" renderer by Solid Angle SL uses path tracing to achieve incredible realism. Path tracing is a type of ray tracing that is typically thought of as the most physically accurate rendering method, but it also has the reputation of being the slowest. Marcos Fajardo has been developing Arnold for 13 years, carefully tuning its algorithms for speed and performance suitable for demanding offline film production, as opposed to physically accurate lighting simulation for architectural and GIS applications. Sony Pictures Imageworks used Arnold exclusively to render *Cloudy with a Chance of Meatballs* and is using Arnold as its sole in-house rendering engine.

Arnold reproduces a full complement of optical phenomena, including global illumination, subsurface scattering, motion blur, deformation motion blur, depth of field, and so on. Arnold, in its current incarnation, is a batch renderer and a licensable API. Solid Angle plans to release Arnold as a standalone renderer and as plug-ins for Autodesk Maya* and Softimage.* Arnold is entirely CPU-based and leverages some of the latest technologies in high-end Intel CPUs to take full advantage of the hardware capabilities available in studios' desktop and server farms.

Expert Perspective

Marcos Fajardo, Chief Architect of Solid Angle's Arnold Renderer

In 2001, Arnold was used to render the Oscar-nominated 50 Percent Grey. The film was a one-person production and had a small CPU budget. Obviously, Arnold can scale to accommodate large-scale render farms, like the one at Sony Pictures Imageworks. What kinds of optimization have you done to achieve scalable, high-performance ray tracing?

Fajardo: "The most important components in a ray tracer are the acceleration data structures. We've been tweaking them from the beginning. There are many different ways that we do that. For speed, you can try to lower the algorithmic complexity and you can squeeze more performance out of lower level code. For example, with our current acceleration structure, we're using the Intel® Streaming SIMD Extensions 4 (Intel® SSE4) instructions to process data. That produces a nice increase in speed, not just in the ray tracing but in the intersection code. For example, we use different primitives—triangle meshes and curves (Bezier splines) for hair—to optimize the intersection routines.

"In addition to optimizing for speed, there's memory use. If you're rendering complicated scenes for visual effects or animated features, you're typically handling super-complicated things with hundreds of millions of polygons, hairs, or both. So we also need to optimize the memory use of these data structures. We try to compact or compress as much geometry information as we can, in order to quickly uncompress that information on the fly while ray tracing. That's tricky, because you want to compress as much as possible, but with higher compression rates comes slower decompression speed. So it's a balance. One particular technique we use for compression is quantization—for example, things are often stored in 32-bit floating-point format, which is often unnecessary. We also try to compact things like x, y, and z vertices, normals, and UV coordinates, as well as arbitrary data that's attached to measurements and vertices.

"Essentially, we're tuning our code so things use less memory and render faster. We do extensive testing to ensure that no feature is added unless its effect on performance is well-understood and deemed acceptable given the added benefit. In particular, the performance hit due to enabling motion blur, and especially geometry-deformation motion blur, is very small and on par with Reyes-style renderers, which is atypical for a ray tracer.

"The bottom line is that we render complex production scenes that were thought to be impossible to render with a ray tracer at reasonable render times, and using much less memory than you'd think. We generally render 100 million triangles in 8 GB—and that's uniquely stored triangles, not instances. With a bit more work, we think we will soon achieve the 1-billion triangles mark."

You worked at Sony Pictures Imageworks for a number of years. What lessons did you learn?

Fajardo: "With production rendering today, studios that utilize hybrid (raster/ray tracing) renderers spend a lot of time precomputing rendering passes—things like shadow maps, point clouds, radiance caches, and all sorts of things that you have to compute before you actually do the final rendering of a frame. This can be a massive task. Artists might spend half a day baking shadow maps and point clouds before they can actually render the scene. These techniques are used to optimize the final render time. And we think that's the wrong approach, because it doesn't take into account the cost of artists doing all this work to make sure that the rendering pipeline is put to efficient use. For example, a lighter's time might be worth $40 per hour, whereas an hour of CPU time is only about $0.10. Plus with each added pass, the pipeline gets more complicated. You have to manage all these files and passes; then you have to synchronize them. It becomes a data management challenge—something that simply disappears when you use a brute force path tracer like Arnold.

"Another benefit is that because we don't combine ray tracing with raster methods the way other production renderers do, there's no need to store two versions of a given model—one for the ray tracer; one for the rasterizer. You also don't have to worry about syncing two renderers to avoid issues where, for example, one thinks a surface is in one location and the other gives a different result."

Arnold is CPU-based. Have you looked at porting Arnold to GPUs?

Fajardo: "No. There are many reasons for that, the most important one being that when work on Arnold began 13 years ago, there was no GPU technology at all. Plus, none of our customers have GPUs in their render farms."

Iterative Rendering on Modern CPUs for Visualization Professionals

Visualization professionals need to take massively detailed CAD models from concept through construction with as much visual precision as time and budget will allow. Thanks to CAD applications such as Bentley Systems MicroStation,* visualization professionals—engineers, architects, GIS professionals, and owner operators—can use the same ray tracing algorithms employed by Hollywood filmmakers to output print-resolution plans, display animated simulations of plant processes and special building features, and produce physically accurate representations of building materials with highly detailed, complex surfaces. All Bentley System MicroStation-based applications incorporate the Luxology rendering engine, providing near real-time rendering capabilities enjoyed by modo users—without specialized graphics hardware.

MicroStation offers multi-core support for 32 Intel processors, as well as distributed network rendering. MicroStation's rendering engine is 64-bit, so that when run on a 64-bit operating system, users can take advantage of all available system memory.

Rendering performance scales based on the number of CPUs, so architects working solo or civil engineers working in teams can make use of Intel multi-core, multi-threaded processor performance on individual workstations or larger render farms.

Credit: Eduardo Makamura

Design visualization is also benefitting from faster rendering performance. This luxury yacht was rendered in modo.*

Expert Perspective

Jerry Flynn, Visualization Systems Product Manager, Bentley Systems

What are some of the key differentiators between the rendering needs of animators and visual effects artists working in film and visualization professionals using CAD software?

Flynn: "Physical material is very important to CAD users. It's essential for renderings to look as real as possible. For example, architects often use metal and glass in their building designs. Using photon mapping to do direct caustics gives us the ability to accurately display properties like absorption distance and set exit color so that light passing through a glass where it's cut on its edge is accurately displayed as a darker color. Rendering techniques such as anisotropic highlights and Fresnel reflections produce realistic metallic surfaces with correct reflections and refractions at any angle to the camera. Materials like carbon fiber can be accurately textured using a blurry reflection anisotropy direction. And complex translucent surfaces—various types of stone and tile, for example—can be displayed accurately with transparency absorption that automatically considers subsurface density."

What kinds of performance gains do you see when MicroStation is run on multi-core processors?

Flynn: "We've seen incredible performance across the entire HP workstation line, from the Z200* with a single Intel Core i7 processor in it to the Z800* with dual six-core Xeon processors, as well as mobile workstations running Core i7 mobile processors. Because the renderer is optimized for multi-core, multi-threaded CPUs, its performance scales with each added processor. And we see a 20-percent performance boost just by enabling hyper-threading."

What other aspects of a workstation architecture are important for efficient CAD rendering?

Flynn: "CAD users need to efficiently address vast amounts of system memory. Our users design all manner of production facilities—plants with pipes everywhere. The models alone are massive—hundreds of megabytes. They take a long time to just load. I don't think there's a GPU renderer that wouldn't choke if you tried to send it scenes of that size. And I have yet to see a substantial model—for example, a 60-story building with all of its details—rendered using a GPU. You can look all over the web and find GPU rendering demos, but they're usually based on very small, optimized models. Those same scenes rendered with the Luxology renderer using CPUs would render in near real time."

CPU-Driven Interactive Rendering

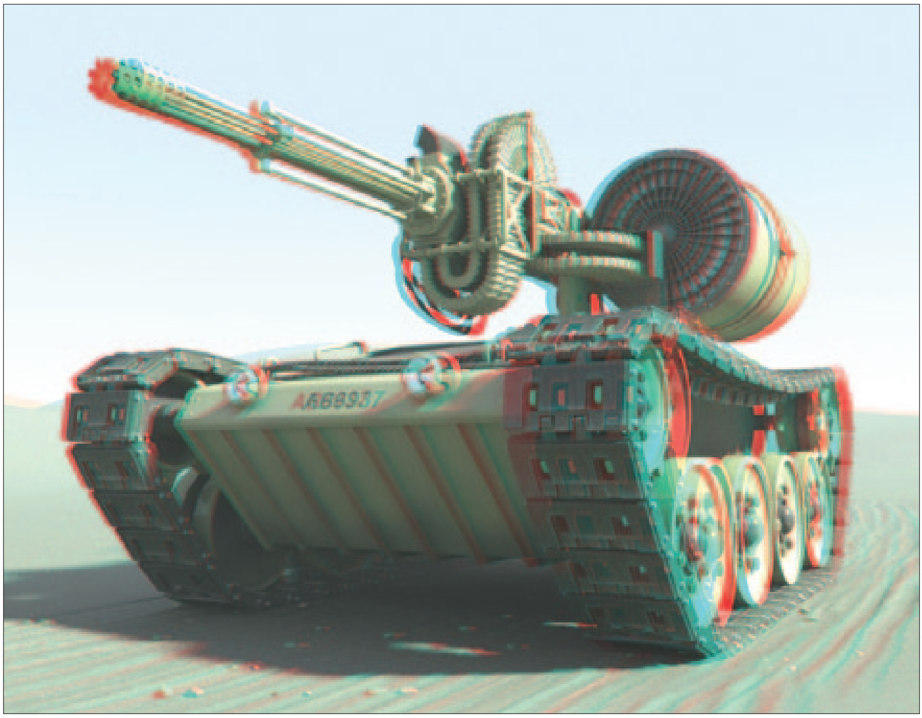
KeyShot* is a CPU-driven interactive ray tracing and global illumination program developed by Luxion ApS for Intel-based Windows and Macintosh computers. KeyShot uses a physically correct and CIE-verified rendering engine featuring unbiased sampling techniques that compute mathematically correct results, scientifically accurate materials, multi-core photon mapping, and adaptive material sampling. Additionally, a dynamic core lighting engine allows users to see results at the same time as changes are being made. KeyShot is optimized for multi-core, multi-threaded Intel CPU architectures.

Expert Perspective

Dr. Henrik Wann Jensen, Chief Scientist, Luxion, 2004 Academy Award Winner, and Inventor of Photon Mapping

Your research has broken a lot of ground. Your bi-directional subsurface reflectance distribution function (BSSRDF) algorithm was used to simulate Gollum's skin in two of the Lord of the Rings films. And you introduced the concept of photon mapping for global illumination caustics. Could you comment on the role of the modern CPU in rendering?

Dr. Wann Jensen: "As rendering gets more and more complex, you move into domains where modern CPUs are stronger. Applications like KeyShot use very computationally advanced algorithms that the multi-core, multi-threaded CPU is made for. We're able to run separate programs on each core, utilize the larger cache, and leverage the overall memory bandwidth to achieve real-time interactive ray tracing that has been verified by CIE (International Commission on Illumination) to compute light correctly. Specialized graphics hardware—GPUs—on the other hand, don't allow separate disparate programs to run simultaneously, but require you to perform the same, essentially simple, operation at the same time on their architecture.


Credit: Arnie Cachelin

A stereoscopic anaglyph image rendered in Luxology modo,* which provides control over-convergence and inter-occular distance for stereoscopic rendering.

"For example, to do photon mapping on a GPU you have to sacrifice physical accuracy for the sake of achieving a high frame rate. Traversing advanced data structures on a GPU is trickier than it is on a CPU, because with a CPU you can leverage the cache to store values that help you more quickly compute things on the fly. When programming GPUs it's preferable to perform such calculations from scratch each time you do them.

"GPUs allow you to achieve great-looking results in real time using highly specialized methods to achieve a particular effect. Those specialized methods don't often mix well with each other. Another drawback to the GPU is that there's no single standard in place. You have to write specialized code for all the different platforms. CPUs provide a more standardized and flexible programming model that allows programmers to focus on developing new algorithms, rather than spend time reworking code so it will run on non-standardized, specialized platforms."

How can end users configure their hardware to get optimal performance from KeyShot?

Dr. Wann Jensen: "Simply add more CPU cores—which is quite a contrast from relying on GPU-based rendering, where you often have software that's optimized for a specific graphics card, which might be replaced the following year. Being able to leverage the raw compute power in CPUs definitely gives you an edge."

CPU-based Interactive Ray-Traced Lighting

Maxwell Render,* from Next Limit Technologies of Madrid, is a ray tracer that utilizes unbiased algorithms to fully capture all light interactions between elements in a scene resulting in extremely realistic images. The software gives users a straightforward, easy-to-understand user interface and an effective workflow. A Maxwell Render license includes plug-ins to a wide range of popular 3D, CAD, and post-production applications.

Maxwell Render includes unique features like Multilight and Color Multilight, which allow users to adjust in real time the intensity of lighting and the color of lighting in scenes before or after a render has been completed without needing to re-render the scene. Next Limit Technologies has announced a new interactive preview feature for Maxwell Render. As with the other renderers mentioned in this article, Maxwell Render's real-time features run on modern multi-core, multi-threaded Intel processors and require no external hardware acceleration.

Expert Perspective

Victor Gonzalez, Co-Founder, Next Limit Technologies

What factors made it possible to implement an interactive ray tracer running on CPUs?

Gonzalez: "From the beginning we tried to create very multi-threaded algorithms. Five years ago we were using all the cores available to us. Now we're taking advantage of 16 cores, so Maxwell Render is 16 times faster. Another factor is that data transfer between CPU and system memory happens automatically in groups thanks to Intel Streaming SIMD Extensions 4 (Intel SSE4) instructions. Creating an application that's SIMD-friendly from the very beginning lets us get the most out of the latest Intel processor technology."

Can you comment on your interactive engine?

Gonzalez: "Our goal has always been to create high-quality renders through physically accurate algorithms. Though the market appreciates this, there has been a strong demand for real-time/interactive approaches that allow users to instantly see the results of their scene changes. We took the decision to develop an interactive engine specially oriented to improve the workflow of the user, without sacrificing the quality element."

Your market includes CAD professionals. Can you comment on their rendering needs?

Gonzalez: "CAD users ultimately want photorealism for their models and designs. That said, in many cases, they are willing to live with less in exchange for speed of image generation and simplicity of operation. They want to be able to previz their images and designs quickly before making adjustments, then create high-quality final images for their clients. This is clearly the case for any users, not just in CAD/GIS, which is why the new interactive preview feature in Maxwell Render is so important to many types of users."

Next Limit offers high-speed rendering solutions that don't require a graphics card. What advantages does CPU-based rendering offer that allowed you to eschew GPUs?

Gonzalez: "We are neither pro-CPU nor pro-GPU. Our point of view is that the current GPUs are experiencing a dramatic evolution but, today, they are still quite restrictive in several aspects, which limit the practicality of hosting a full unbiased render solution like Maxwell Render. At the time of deciding where to concentrate our efforts, we have to balance the options available to us, and the impact they have on users. We also look at other aspects like maturity—CUDA* or OpenCL* are in their early stages, many things are subject to change that can affect our development roadmaps. We want to work on a standard approach—that is, ubiquitous hardware—without asking customers to buy additional, specific hardware.

"The CPU is not dead. Multi-core CPU technology is evolving quickly, and we proved that interactivity can be achieved using pure CPU approaches. We don't discard GPU, of course, we are actually doing research on them. It's just a trade-off question. The interactive engine is taking only a few months of development, which wouldn't be the case in a pure GPU approach. Simple ray tracers can be implemented easily on GPU, but as soon as you get into more complicated ray tracers for more complicated scenes, GPUs can't cope.

"Where CPUs are significantly faster than GPUs is when the scene description involves tens of millions of polygons, lots of textures, lots of additional data that you have to access in memory. As soon as all of that doesn't fit in the first level of the cache, you get into caching and quality issues. Moreover, modern CPUs allow more freedom to create complex structures. Our material model in Maxwell can have an arbitrary size as the user adds more layers. This freedom is not guaranteed in GPUs where the algorithm has to be packed and sized to fit inside the GPU registers. This means that if the algorithm fits perfectly in the GPU, you are lucky—you'll get that X factor in performance. But it's very probable that the more complex the algorithm, the less probable the GPU can digest it.

"With CPUs, you have freedom to work with any algorithm you want without being restricted. You have more freedom to innovate."

Credit: Benjamin Brosdau, Pure. www.purerender.com

Chesterfield leather couch, a living room scene rendered with Maxwell Render,* which uses unbiased algorithms to capture light interactions between elements.

Summary

Computer-generated imagery is more sophisticated than ever. Real-world production rendering involves enormous scene files, multiple high-resolution texture maps, and highly complex shader models to produce high-impact imagery with exacting realism. For these tasks, the modern multi-threaded, multi-core Intel CPU architecture offers:

- The familiar "x86" Intel Architecture that supports an enormous and robust ecosystem of compilers, languages, libraries, tools, and code
- Access of up to 128 GB system memory on modern workstations; up to 1 TB in server farms
- Superior cache size, which dwarfs that of specialized graphics hardware
- High-speed 32 GB/sec data throughput, thanks to three channels of 1333 MHz memory
- The ability to handle virtually any combination of rendering algorithms, accommodating the most demanding and highly complex rendering schema
- Exceptional performance that scales nearly linearly as more processor cores are added to the pipeline
- A mature ecosystem of programming tools streamline development of new rendering techniques that don't run on specialized, non-standardized hardware
- The ability to use standard or legacy hardware configurations
- The ability to run legacy as well as new code on the same CPU nodes

Intel CPUs have helped to enable decades of innovation in offline rendering. Recent speed gains and innovation by Intel Solution Partners has enabled the creation of interactive renderers that complement offline rendering. With great performance and a mature ecosystem of languages, libraries, and tools, Intel processors remain the platform of choice for tomorrow's render farms.

For More Information

Intel processors
www.intel.com

Luxology modo rendering
www.luxology.com

Bentley Systems
www.bentley.com

Luxion KeyShot
www.keyshot.com

Solid Angle Arnold
www.solidangle.com

Next Limit Maxwell Render
www.maxwellrender.com

SOLUTION PROVIDED BY:

[†] 64-bit computing on Intel architecture requires a computer system with a processor, chipset, BIOS, operating system, device drivers and applications enabled for Intel® 64 architecture. Processors will not operate (including 32-bit operation) without an Intel 64 architecture-enabled BIOS. Performance will vary depending on your hardware and software configurations. Consult with your system vendor for more information.

[1] Lord of the Rings render farm stats: http://www.digitalartsonline.co.uk/features/index.cfm?featureid=1162.

[2] Gollum render times: http://en.wikipedia.org/wiki/Special_effects_of_The_Lord_of_the_Rings_film_trilogy.

[3] Treebeard render times: http://www.imdb.com/title/tt0167261/trivia.

[4] The Race for Real-time Photorealism, Dr Henrik Wann Jensen and Tomas Akenine-Möller, American Scientist, Volume 98.

[5] Avatar rendering stats: http://www.datacenterknowledge.com/archives/2009/12/22/the-data-crunching-powerhouse-behind-avatar/. http://blogs.pcmag.com/miller/2010/06/james_cameron_tells_d8_how_he.php

Debunking the 100X GPU vs. CPU Myth: An Evaluation of Throughput Computing on CPU and GPU, Victor W Lee, Changkyu Kim, Jatin Chhugani, Michael Deisher, Daehyun Kim, Anthony D. Nguyen, Nadathur Satish, Mikhail Smelyanskiy, Srinivas Chennupaty, Per Hammarlund, Ronak Singhal,and Pradeep Dubey, ISCA, June 2010.

Printed in USA 1110/MR/OCG/PP/XK ♻ Please Recycle 324545-001US