Autodesk University 2012

The Fantastic Four: Autodesk 3ds Max and Its Four Native Rendering Engines

Gary M. Davis – Autodesk Media and Entertainment

AV2823

Autodesk 3ds Max software has more out-of-the-box rendering options than any other digital content creation (DCC) software available today. In this class, Autodesk Senior Technical Specialist Gary M. Davis explores these different solutions and examines when and why users might choose to use the mental ray, iRay, Quicksilver and The Scanline rendering engines. The class also explores uses that combine multiple renderers for more complex workflows.

Target Users:

Beginner to Intermediate

Learning Objectives

At the end of this class, you will be able to:

- Describe the four native rendering engines and when to use each in production
- Evaluate the pros and cons of each
- Combine rendering engines for more advanced workflows

About the Speaker

Gary M. Davis began his career after receiving a BFA in computer graphics from Bowling Green State University in 1992. His career has covered disciplines within visual effects and motion graphics for clients in TV/film, video games, simulation and design visualization. In 1999, after nearly six years focusing on the development of simulator ride films and digital photography systems for themed entertainment venues, he created his own boutique; visualZ, LLC. At SigGraph 2007, Davis was awarded the title of Autodesk 3ds Max Master. Shortly thereafter, he joined the Media & Entertainment Division of Autodesk as a Technical Specialist focusing on 3D animation, compositing and creative finishing.

BLOG http://area.autodesk.com/blogs/garyd
TWITTER @garydvisualz

“Rendering is an art of trying to cheat compromises.”

Agenda:

- Intro to the rendering engines
- Recurring topics and considerations
- A brief look at each rendering engine
- Combining rendering engines
- QnA

Recurring Topics and Considerations

- GPU vs. CPU
- Time vs. quality (vs. budget... pick two)
- Point(s) of diminishing returns
- Project and studio costs of man hours vs. machine hours
- Raytracing; reflection and refraction
- Antialiasing
- Lighting
    - Area lights and soft shadows
    - Indirect Illumination and color bleeding
- Shaders
    - Primary shader(s)
    - Shader caveats

Default Scanline Renderer

The Default Scanline Renderer is a versatile renderer that renders the scene as a series of scanlines that are generated from top to bottom. Approximately 80 percent of 3ds Max users incorporate the Scanline renderer in their productions in some way or another.

“What’s old is new, again”

Scanline raytracing and light tracing (etc.) on today’s hardware. Scanline is an excellent option for and overall simple and effective renderer.

Antialiasing - Smooths the jagged edges that occur along the edges of diagonal and curves lines when rendering. Turning off Antialiasing disables the Force Wireframe setting (see proceeding). Geometry renders according to the assigned material even if Force Wireframe is turned on. Turning off Antialiasing also disables render elements. If you need to render

elements, be sure to leave Antialiasing on

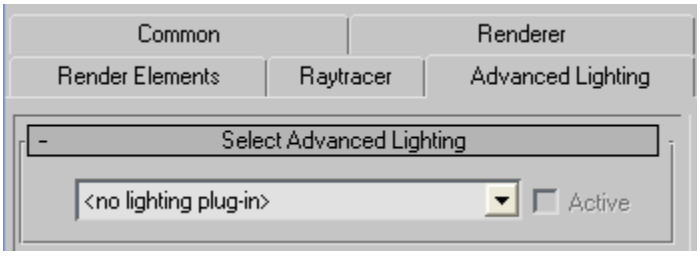
Conserve Memory - When on, rendering uses less memory at a slight cost of memory time. Memory saved is in the range of 15 to 25 percent. The time cost is about four percent. Default=off.

Antialiasing in the Scanline Render

The following table describes the available antialiasing filters in the Scanline renderer.

Name	Description
Area	Computes antialiasing using a variable-size area filter. This is the original 3ds Max filter.
Blackman	A 25-pixel filter that is sharp, but without edge enhancement.
Blend	A blend between sharp area and Gaussian soften filters.
Catmull-Rom	A 25-pixel reconstruction filter with a slight edge-enhancement effect.
Cook Variable	A general-purpose filter. Values of 1 to 2.5 are sharp; higher values blur the image.
Cubic	A 25-pixel blurring filter based on a cubic spline.
Mitchell-Netravali	Two-parameter filter; a trade-off of blurring, ringing, and anisotropy. If the ringing value is set higher than .5 it will impact the alpha channel of the image.
Plate Match/MAX R2	Uses the 3ds Max **2** method (no map filtering) to match camera and screen maps or matte/shadow elements to an unfiltered background image. For details, see the section “Plate Match Filtering” in the introduction of this topic.
Quadratic	A 9-pixel blurring filter based on a quadratic spline.
Sharp Quadratic	A sharp nine-pixel reconstruction filter from Nelson Max.
Soften	An adjustable Gaussian softening filter for mild blurring.
Video	A 25-pixel blurring filter optimized for NTSC and PAL video applications.

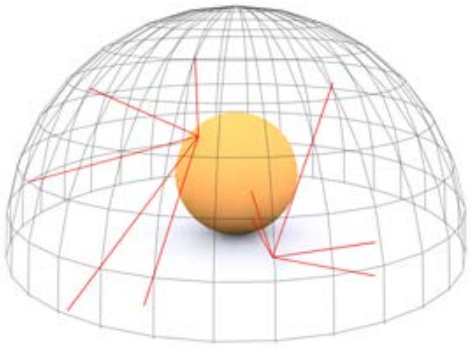
Scanline Advanced Lighting



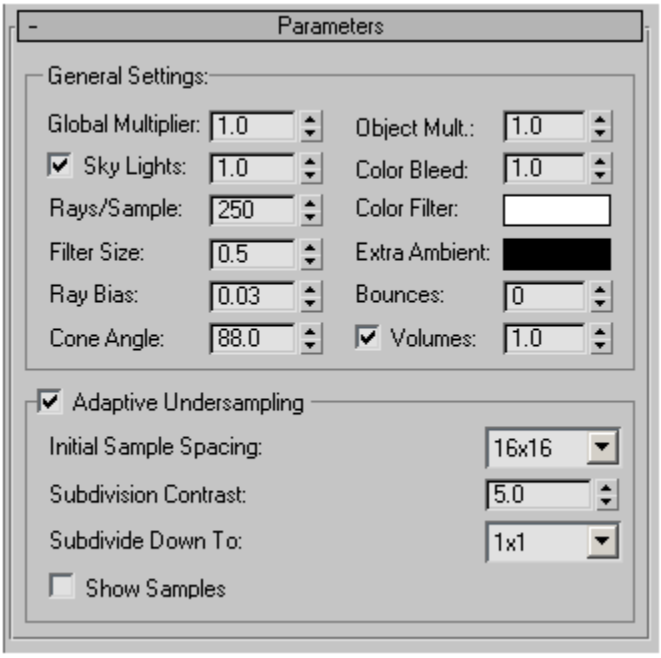
Choose an advanced lighting option from this drop-down list. Default=No advanced lighting. When an advanced lighting option is chosen, use Active to toggle whether the advanced lighting is used when you render your scene. Default=On.

Light Tracer - provides soft-edged shadows and color bleeding for brightly-lit scenes such as outdoor scenes. It is typically used in conjunction with a Skylight. Unlike radiosity, the Light Tracer does not attempt to create a physically accurate model, and can be easier to set up.

The Light tracer is extremely easy to use and the results look absolutely phenomenal when considering how easy it really is! However, it is a relatively slow process and network rendering should be considered. Exterior scenes benefit greatly from the use of a **Skylight** object in the scene but do not use the mental ray Sun and Sky system.



Scanline Renderer, one Skylight and Light Trader enabled as Indirect Illumination method.



Radiosity - This method requires a one-time, upfront calculation and the results are indirect illumination as vertex color information. Radiosity is rendering technology that realistically simulates the way in which light interacts in an environment.

Radiosity results are, admittedly, slightly dated at the time of this writing due to more robust methods of calculating and rendering indirect illumination. Having said this, if users are outputting models and scene assets to game engines or realtime simulation platforms, this indirect illumination method might be worth more investigation.

Image Motion Blur

If you decide you need motion blur from the Scanline renderer and do not plan to add it as a post process, there is little argument to the preferred method is using Image Motion Blur and not Object Motion Blur. IN general, the results are smoother and typically faster to generate than Object Motion Blur, which is prone to artifacts. You determine which objects have image motion blur applied to them by setting Image in the Motion Blur group of the Properties dialog for selected object(s). Image motion blur blurs the object by creating a smearing effect rather than multiple images. It takes camera movement into account. Image motion blur is applied after scanline rendering is complete.

The coin on the right has Image Motion Blur applied using The Scanline renderer.

You can’t put image motion blur on objects that change their topology.
Tip: When blurred objects overlap, sometimes blurring doesn't work correctly and there are gaps in the rendering. Because image motion blur is applied after rendering, it can't account for object overlap. To fix this problem, render each blurred object separately, to a different layer, and then composite the two layers using the Alpha Compositor in Video Post.
Note: Image motion blur doesn't work for NURBS objects that are animated so their tessellation (surface approximation) changes over time. This happens when sub-objects are animated independently of the top-level NURBS model. Nor does image motion blur work on any of the following:

- Anything with an Optimize.
- Any primitive with animated segments.
- MeshSmooth of any type with a "Smoothness" value (under iterations) other than 1.
- MeshSmooth on polygons with Keep Faces Convex on.
- Anything with Displacement Material.

The mental ray Renderer

Mental ray is the most sophisticated and complex of the Fantastic Four (rendering engines).

'…and with great power comes great responsibility"

The mental ray renderer from NVIDIA is a general-purpose renderer that can generate physically correct simulations of lighting effects, including ray-traced reflections and refractions, caustics, and global illumination.

Compared to the default 3ds Max scanline renderer, the mental ray renderer relieves you of the need to simulate complex lighting effects "by hand" or by generating a radiosity solution. The mental ray renderer is *highly* optimized to use multiple processors and to take advantage of incremental changes for efficient rendering of animations.

Unlike the default 3ds Max renderer, which renders scanlines from the top of the image downward, the mental ray renderer renders rectangular blocks called buckets. The order in which the buckets are rendered can vary, depending on the method you choose. By default, mental ray uses the Hilbert method, which picks the next bucket to render based on the cost of switching to the next one. Because objects can be discarded from the memory to render other objects, it's important to avoid having to reload the same object multiple times.

While this might be the case, new users should not be intimidated by mental ray as those in years gone by. For the last several releases of 3ds Max, there have been 'templated' workflows and things have been made much easier than the "days when you need a physics degree to use mental ray". There are presets and simple adjustments for fine-tuning materials, lights, and renderer settings that should prove to be ample starting points for productions of various types. This might be the case, but as users progress and learn more about the different technologies present in mental ray, they will undoubtedly feel more comfortable going in and twiddling with the settings to experiment beyond the template presets and workflows.

Some important considerations of mental ray to research for users of all levels include

- Antialiasing and tuning these settings
- Arch and Design Shader (A+D shader)
- Final Gather basics with Skylight
- Final Gather basics with Sun and Sky System with mr Photographic Exposure Controls

Raytracing Rules of Thumb with mental ray; an Example...

Say you're rendering a (lathed) wineglass, with an inner and outer surface and a piece of geometry representing the wine. The wine geometry is just slightly smaller than the inner surfaces of the wineglass, and capped with a flat top. Now, you go to render the glass. After rendering the scene, however, there's something wrong: the inner surfaces of the glass don't seem reflective enough, and the wine isn't refracting properly. What's wrong?
It's possible that you have the number of reflections and refractions set too low for the number of surfaces you have. To check this, go to the Renderer panel ➤ Rendering Algorithms rollout and look at the Maximum Trace Depth settings. If you haven't changed the parameters, then you should see Max. Reflections and Max. Refractions set to the default of 6, and Max Depth set to 6.

There's the problem: you actually have six surfaces that need to be traced by the light rays for both reflections and refractions. The way to always calculate the number of rays needed for a scene is to take the ray-traced objects in your scene and draw an imaginary line through them, originating at the point of view. Then, count the number of surfaces the line intersects.
For the wineglass and wine, you need at least six reflections and refractions that correspond to the following surfaces:

- Near outer glass surface ("near" relative to your Camera viewpoint)
- Near inner glass surface
- Near wine surface
- Far wine surface
- Far inner glass surface
- Far outer glass surface

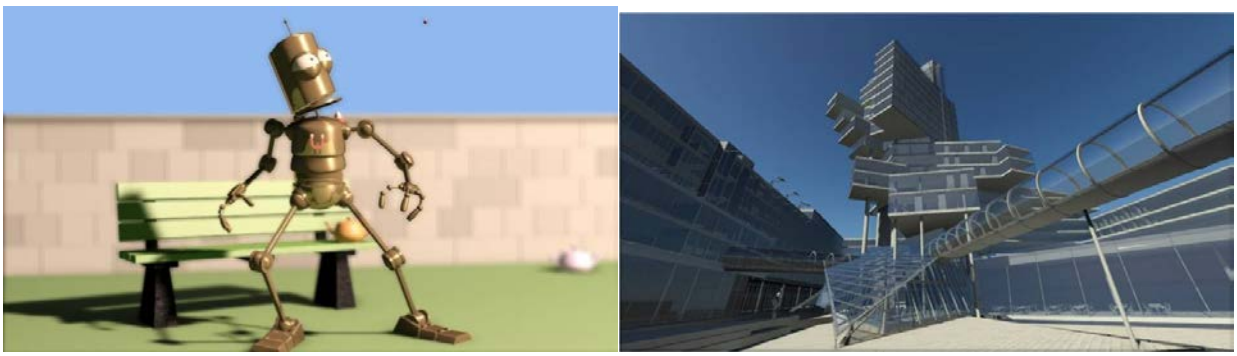
Therefore, increase the value of Max. Depth to 12.

Quicksilver Hardware Renderer

While Quicksilver is directly tied to the Nitrous viewports, you can use any viewport driver you want. It is a 'proper' rendering engine (not just a playblast). It uses both the GPU and the CPU to render.

XBR Graphics: Quicksilver

As part of our XBR effort, we are completely reworking our viewport system, the first part of this was our MetaSL integration back in 3ds Max 2010. The next part of the effort is a new GPU-based rendering system that will first show up as a hardware-based rendering system in 3ds Max 2011. Our goal wasn't so much to perfect yet another production rendering solution, as much as take something we were working on and release it as a feature that you can hammer on now and get some benefits from. Some of you will be able to use Quicksilver in your workflows and others will not because your needs don't map to what Quicksilver is good at. Our orientation is to give you a better tool than viewport capture but not spend our resources trying to replace existing production rendering solutions. Basically, it is a very simplified workflow that is pretty identical to our mental ray workflow (whatever you set up in Quicksilver can be rendered in mental ray without much effort). Some people will use it for faster previews, and client sign-offs, some might use it for actual production purposes because their needs match Quicksilver's capabilities and they gain a big advantage with its performance. It's certainly not for everyone or for any problem. We have measured 10X or more improvements at the same visual quality for many scenes over mental ray, but depending on your hardware and needs your results could be a lot different. If you have 8 or more CPU cores and not much of a GPU, you're going to likely be better off with a software renderer. If you have 4 cores or less, and a relatively beefy GPU, then you might get the benefit depending on what you're trying to render.



What Quicksilver is:

It uses both the GPU and CPU to perform the rendering. You can somewhat think of it as a game-engine for rendering since it uses DirectX 9c and the GPU to get the job done. This is driven using advanced shader technology from mental images (MetaSL) and some technology we've built. Because we're using these game engine techniques, you can render a 10M+ poly scene with just a 512Mb graphics memory since we use vertex buffers very efficiently. We've built Quicksilver from the ground up (except mental image's MetaSL) and there is some magic sauce involved in generating adaptive shadow maps and our indirect lighting mode that I don't believe you'll find in any other generalized GPU rendering system. We use the CPU primarily to translate the scene data to the renderer. It will also be used to compile the shaders and generate data for the indirect lighting pass. Both systems are fully multi-threaded, including the shader compilation. We have to compile shaders for specific GPU configurations – which helps future-proof this technology. When you first run Quicksilver, you'll experience a pause on the first frame. The first frame can take a while due to the fact the shaders need to be compiled. This can be a lightweight operation or costly. For example adding noise maps to a shade tree can add up to 3,000 instructions. This not only takes time to compile, but is also close to the limits of the shader model 3.0 standard. 3ds Max maintains a cache of these compiled shaders that effectively live forever. The more you use Quicksilver the faster it becomes – it will only rebuild a shader if the system detects a newer shader on the system. The user is free to flush the cache whenever they desire. Quicksilver only supports MetaSL-based shaders. The good news is that you now have "Slate" our new node-based material editor which lets you create shade trees directly with MetaSL and we've converted most of our legacy shaders in 3ds Max to MetaSL. This means a lot of existing materials will just work "as is". It's important to realize that the new MetaSL shaders are a one to one reproduction of the original 3ds Max procedural maps. This has one important benefit - the viewport can display the shader as it will render in Quicksilver or in mental ray or even in scan-line. Of course, making sure you have pixel-level detail for this requires lots of shader instructions – which requires more GPU resources to do it quickly.

Some of the key technology/features that you'll find in Quicksilver include:

- Use of MetaSL shaders
- Sophisticated shadowing system including adaptive shadow map creation (provides fine detail of shadows in large scenes)
- Photometric lights
- Indirect Lighting simulation (not screen space like 3ds Max 2010, emulates effects of indirect lighting)
- Reflections of dynamic objects (non-raytraced)
- Ambient Occlusion (screen-space)
- DOF (shader effect)
- Motion blur (from camera passes)
- Hardware and software based anti-aliasing and super sampling
- Alpha and Z-buffer render elements
- Larger-than-screen resolutions
- Hundreds of dynamic lights (if you have enough GPU for this)
- Floating point render pipeline with tone-mapping
- Support for Backburner (you better have the same GPU on each node or you'll get unpredictable results)
- Multi-threaded CPU optimization

What Quicksilver isn't:

It is not a ray-tracing approach. If you have those kinds of needs, use mental ray or the scanline renderer or your favorite renderer. You can't throw everything at Quicksilver and still expect it to render faster than anything else you have. It is very, very good at a class of problems and that class of problems is bigger with more powerful GPU hardware. There are many limitations in Quicksilver that may prevent you from applying it in your workflow.

Quicksilver Limitations:

These are various bugs and issues with using Quicksilver. Some of these limitations are targeted for bug fixing while others require additional research and development (and we can't talk about future plans for features). We'll try to keep you all updated (and this doc) when the status of some of these change.

- You will sometimes see "black surfaces". These are usually signaling to you that a material failed to compile – either because it is too complex or because it is not supported by Quicksilver. You will need to use a different rendering solution or replace these materials to get things to work properly.
- Quicksilver currently needs to have tiling enabled in the Bitmap texture map - the shader currently doesn't like it when these are unchecked
- No blend material, matte/shadow material, composite map or material
- 3ds Max's falloff node is supported to a degree - basic setups, the rest is baked. However, the new falloff MetaSL node is supported.
- No "ink n 'paint" but there are some ways to use MetaSL to achieve some of these affects

- Doesn't respect object properties (can't hide reflection planes, etc.)
- Supports all 3 types of 3ds Max sky lights but without Image Based Lighting (IBL) - you'll see the mr sun/sky system - but without IBL
- No support for the self-illumination shader in A&D nor the rounded corners shader (the shader would be too complex to compile/run)
- Only supports cube map reflections (no planar reflection at this time)
- No support for shadow patterns
- It does not support Render to Texture

Things that will kill performance:

- Noise maps are very expensive shaders
- Lots of transparent surfaces
- Too many things selected for dynamic reflection

Hardware considerations:

- We only use a single GPU at this point, either consumer boards or professional boards – we recommend professional boards as these are the ones we focused on certifying. See certified boards http://www.autodesk.com/max-hardware (will be updated for 3ds Max 2011 at some point)
- Quicksilver does not use CUDA or OpenCL at this time - it only uses DirectX 9.
- Quicksilver requires a minimum of 512Mb of dedicated graphics memory to get results, in practice you'll want 1Gb+
- Both NVidia and ATI hardware are equivalent in feature support – performance-wise is impossible to state because it will depend on your scene and specific hardware driver configurations
- Use the GFX Checker app to help understand if your hardware has any obvious problems with running Quicksilver. Install and double-click on the .bat file - that will generate a log file.

Tips/Tricks using Quicksilver:

- If you run into problems, try running QS in forward shading mode. In the MaxScript listener, type the following: renderers.current.lightingmode = #forward
- If you don't care about the 2D Coordinate rollout of the bitmap texture, use the MetaSL node "Texture Lookup 2D" – you will be able to cram more into your shade trees.

Additional details about Quicksilver:

Supported materials:

- Arch & Design material, including the new Autodesk Materials (but not ProMaterials)
- Standard material
- Double-Sided material
- Multi/Sub-Object material

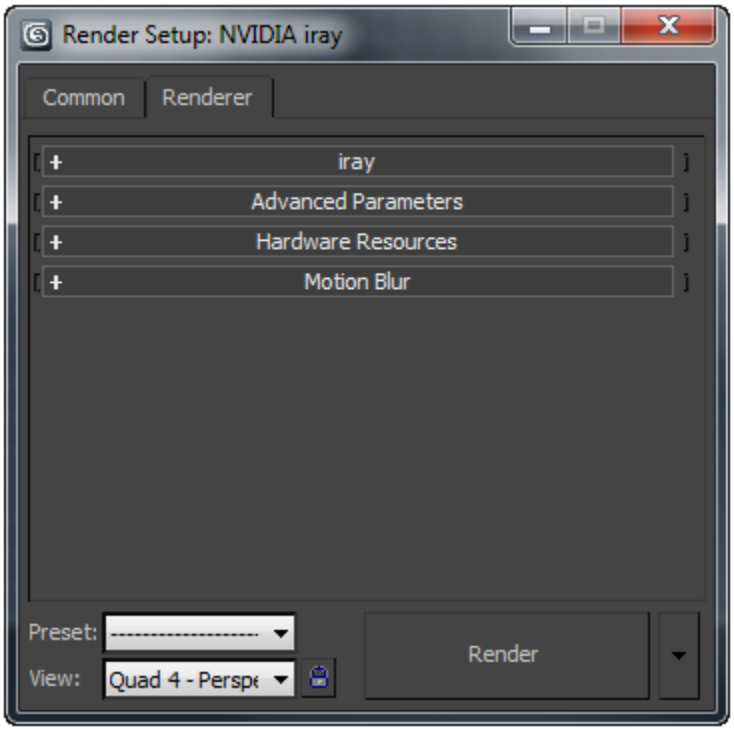
Supported maps/shaders:

- All MetaSL shaders (no support for DirectX shaders at this point)
- mental ray shaders:
    - mr Physical Sky
    - Utility Gamma & Gain shader

Standard maps:

- All Standard maps are supported except for the following:
    - Cellular map (converted to baked map)
    - Gradient Ramp map (converted to baked map)
    - Output map (converted to baked map)
    - Vertex Color map

The iRay Renderer



The NVIDIA iRay renderer creates physically accurate renderings by tracing light paths. It requires little setup compared to other renderers. iRay is an intuitive to operate, interactive, consistent, high-performance global illumination rendering technology that generates photorealistic imagery by simulating the physical behavior of light.

Unlike current ray-tracing renderers, iRay does not depend on complex renderer specific shaders and settings to approximate global illumination. iRay achieves its high level of performance by taking full advantage of the CUDA programming model, allowing interactive previewing on both single and multiple NVIDIA GPU platforms. iRay balances intuitive ease of use and scene setup with the highest quality photorealistic final frame output and interactive performance.

Core Features

iRay generates photorealistic imagery without introducing rendering algorithm specific artifacts, and without requiring the use of renderer specific parameterizations. iRay progressively refines the image until maximum fine detail is reached, providing a single process which smoothly combines interactive pre-visualization and final frame rendering.

When coupled with NVIDIA GPUs, iRay progressively produces final frame photorealistic images, with changes displayed at interactive frame rates.
Interpolation techniques, which trade final quality, predictability, and simplicity of scene specification for performance, form the core of most current global illumination renderers. Unlike them, iRay rendering is based on deterministic and consistent global illumination simulation algorithms that converge without introducing persistent approximation artifacts.

Shading and Material Support

iRay uses a highly optimized BSDF and EDF shading framework as opposed to an involved collection of programmable shaders.
Lighting support in iRay includes:

- HDR environment maps
- IES profile lights
- Spot, point, and directional lights
- Area lights of all shapes
- mental ray Sun and Sky shader

Features and Workflows

- Customize UI and Defaults Switcher
- Material Editor Renderer
- MaxStart scene(s) and Render Presets
- testing Testing TESTING!
    - Make a change, render, notate/save, repeat (never to JPG!)
    - RAM player
    - Composite's Compare node
    - The Backburner Queue
    - Skip existing
    - mr DBR

- State Sets: for renderer switching

Third Party Tools

- Autodesk Material Converter ($49 Euro or ~$65 USD)
  http://www.3dstudio.nl/producten/maxscripts/autodesk-material-converter
- mental ray and iRay managers (free)

  http://www.youcandoitvfx.com/

- Allegorithmic Bitmap2Material ($169.00)

http://www.allegorithmic.com/products/b2m/overview

Combining Rendering Engines in Production

A few combinations for investigation:

1. Scanline with AO (mental ray or QS)
2. Scanline or mental ray mattes for iRay
3. QS previews for animation tests
4. Local iRay tests for network rendering with mental ray
5. Render to Texture (RTT) map baking

Links

http://www.mentalimages.com/index.php
http://www.mymentalray.com/
http://mentalraytips.blogspot.com/
http://www.irayrender.com/
http://area.autodesk.com/blogs/shane/the_iray_faq
http://jeffpatton.net/
http://www.fxguide.com/featured/the-art-of-rendering/

Additional Resources

(Book) Mastering mental ray: Rendering Techniques for 3D and CAD Professionals.
By Jennifer O'Conner
http://www.amazon.com/Mastering-mental-ray-Techniques-Professionals/dp/0470563850

(Third party plugin) Autodesk Material Converter
http://www.3dstudio.nl/producten/maxscripts/autodesk-material-converter