RenderMan 24.0

Soul © Disney/Pixar

Welcome to RenderMan 24.0

RenderMan version 24 provides major upgrades to look development, with many tools that improve artist workflows and complement the creative process. Here are some of the highlights:

Major Features in RenderMan 24.0

- *XPU™* — Pixar's hybrid CPU + GPU rendering technology is a next-generation rendering engine, rewritten for speed and efficiency on film production assets. This first phase of XPU is focused on accelerating look development for shading artists. XPU is only available in the Commercial version of RenderMan
- *MaterialX Lama* — A state-of-the-art material layering system developed at Industrial Light & Magic introduces a modular approach to building material networks and includes new work on dispersion and energy conservation
- *Stylized Looks™* — Move beyond physically based shading and lighting into a world where you can easily create a variety of styles for your projects. You can non-destructively control outlines, create sketch patterns, and develop a wide range of unique looks, including Anime, by using the same familiar toolset. Stylized Looks is available only in the Commercial version of RenderMan
- *OpenColorIO* — Robust support for the industry standard ACES color management system and other color spaces in all bridge products, including the interactive RenderMan Image Tool
- *Live Statistics* — Watch your rendering resource usage live, thanks to a completely redesigned statistics system that prioritizes interactivity and extensibility
- *New Patterns* — We are continually working to provide more artistic options for your look development. New with this release are production-proven technologies Hex Tiling Manifold and Phasor Noise
- *Light Baking* — Greatly enhance your real-time and VR rendering needs by baking lighting to 2D texture maps or Point Clouds
- *OSL Patterns* — We have converted the great majority of C++ patterns to OSL. This conversion allows the sharing of code between RIS and XPU, which provides confidence that the renders from RenderMan XPU are representative of what you will see in RenderMan RIS. C++ patterns are still supported, but they will only work in RIS
- *Better Sampling* — Blue noise dithering results in a perceptually cleaner image sooner
- *Updates to Artist Tools* — Support for Autodesk's Maya, Foundry's Katana, and SideFX's Houdini and Solaris (including support for LPEs and AOVs in Solaris).
- *Preset Browser* — Now supports Presets for MaterialX Lama and Stylized Looks and ships with examples for both.
- *RenderMan for Blender* — And last but not least, a new plugin for cutting edge RenderMan features in the open source content creation tool. Rewritten to take full advantage of the architecture of version 24.

RenderMan XPU™

Pixar's solution for CPU + GPU rendering uses heterogeneous computing resources to accelerate production path-tracing. Starting from a system architecture that can leverage the latest developments in many-core CPU and GPU hardware, separately or in combination, RenderMan XPU™ is being built to handle the scale and complexity of Pixar's feature animation projects. Its first use in the studio has been focused on detailed look-development tasks, with more roles and capabilities to come.

For further details about RenderMan XPU™, see the following documentation sections:

- XPU - for an overview of XPU in RenderMan 24
- Shader & Look Development with XPU - for details on lookdev with XPU
- Features & Limitations - to understand the differences between XPU and RIS and for known limitations in XPU

The Artist Experience

RenderMan version 24 provides major upgrades to Look Development, with many tools that improve artist workflows and complement the creative process. By collaborating closely with Industrial Light & Magic and Pixar Animation Studios, we bring you innovations that have been developed for some of the most ambitious VFX and animation work in the world.

We've made the non-photorealistic toolset an integral and seamless part of RenderMan, allowing for new creative possibilities.

Other new features for artists include:

- *Dispersion* - The new layered materials system supports a sophisticated prismatic fringing effect for refractive objects
- *USD* - hdPrman includes dynamic rendering of LPEs and AOVs in all compatible Hydra viewers, such as Houdini's Solaris and USDView
- *Bump Roughness* - An innovative system developed at Pixar Animation for rendering micro details such as scratches efficiently
- *Bloom* — Add gleams and blooms directly to your live renders with this physically-based tool for RenderMan's Image Tool (IT).

The new features are complemented by an all-new artist-friendly stats system.

Additional Features

RenderMan version 24 also includes:

- OpenVDB — New support for OpenVDB 6.2.1.
- OpenEXR — Updated to version 2.3.
- Updated API — New changes to plug-in APIs and developer resources.

VFX Reference Platform 2019 — All plug-ins are now updated to conform to the standard.

Known Rendering Differences - RIS 24 vs RenderMan 23

As with every RenderMan release, we continue to push the capabilities of the renderer. At times, this can result in minor look differences from one major version to another. We have added backward compatibility flags and switches where possible.

- OSL pattern differences to the old C++ implementation – particularly with noise patterns. The default implementation of various noises in OSL is different than it is in C++: if your renders use these noises you will see different results. We have provided a backward compatibility switch on the relevant OSL patterns. We don't recommend using the switch unless you must because performance will suffer
- OSL noise differences. In RenderMan 24.0, we have upgraded to OSL 1.10. Several of the noise patterns have changed implementations in this new version of OSL, causing look differences.
- The PxrPathTracer and PxrUnified integrators now treat subsurface scattering the same as diffuse with respect to the "maxIndirectBounces" integrator parameter and "trace:maxdiffusedepth" attribute. A value of 0 for maxIndirectBounces means direct diffuse and "direct" subsurface scattering; 1 means 1 bounce of indirect diffuse and 1 bounce of subsurface scattering, etc. This results in a minor change in look between R24 and R23, as well as a minor performance decrease. We believe that the new behavior is more intuitive for artists and TDs. If you want to have the same look as R23 with the same performance characteristics, you can set an Attribute on any subsurface object in your scene to set "maxdiffusedepth" one less than it was previously.
- PxrSurface now supports two lobed Henyey-Greenstein for single scattering. Existing scenes will need to migrate the "g" parameter to the "g0" parameter.
- PxrMix parameter "mix" is now "mixer". This change was necessary because the nodes are now OSL, for which "mix" is a reserved keyword. The parameter "clampMix" changed similarly to "clampMixer".
- The OSL texture3d() function no longer transforms the P parameter from world space to object space.
- PxrFlake: Fixed moire artifacts and lower the cost of multiple flake layers (octaves). The pattern has changed but has the same characteristics as the previous version.
- Blue Noise: In RenderMan 24, we are introducing a new tiled blue noise sample scheme that builds on top of our existing pmj02 samples. This new sampling scheme shows most visual improvement for low sample counts and simple sample domains such as strong motion blur, strong depth of field, direct illumination from large area lights and dome lights, ambient occlusion, and volumes. When the image is fed to a denoiser, the new scheme can also help the denoiser better distinguish between noise (that should be smoothed) and actual image features (that should be preserved). The new scheme does not give lower error or improved convergence for any given pixel, but the visual quality is improved when the pixels are considered together. An option is provided to revert back to the old random number sample sequence if preferred.
- The default "pixelfiltermode" setting for the Hider is now "importance" instead of "weighted". This was done to align RIS with XPU. If you are not using the denoiser and are using RIS, you might consider changing the mode back to "weighted": often it will converge quicker than if "importance" is used. However, use of "importance" is required for the denoiser.

- The default for glassIor initialization in PxrSurface now matches the Args file. If you see differences with shading, you may need to explicitly set the IOR to 1.0 within the shading network.
- For volumes, Oi used to return a single float value to an AOV when a color is expected. The transparencyAccum of PxrUnified is now a color. This allows PxrUnified to correctly output a colored Oi when given a volume with colored extinction.
- The PTM occlusion term is no longer automatically applied to the denoising albedo.

Pipeline and Coding Details and Differences

This section contains a listing of the items that have changed within RenderMan RIS and our Bridge Products that affect your pipeline and any code that you may have surrounding RenderMan.

- VFX Reference Platform 2019. RenderMan 24 supports the VFX Reference Platform 2019. If you haven't already upgraded your systems with the requirements for the VFX Reference Platform 2019, you will need to do so in order to run RenderMan 24
- OSL 1.10: We have upgraded our OSL to version 1.10. Note that in some instances, the look will be affected if you have written your own OSL patterns because some OSL noise functions have changed.
- OSL shaders now support connecting scalar values to individual elements of an array shader parameter in the same way C++ shaders do
- OSL manifolds now only take combined 2D primvars (st versus s, t) for increase performance in XPU.
- Allow explicitly setting dynamic array parameters in OSL shaders
- To take advantage of the new Color Management and Bump to Roughness features, you will need to expand your txmake workflows when converting your textures for RenderMan
- _MAPID_ and atlasStyle will become deprecated in future releases. Please use <UDIM>, <u><v> and <U><V> for all texture atlas specifications
- The PxrPrimvar built-in names "curvature_u" and "curvature_v" for principal curvatures are now "curvatureU" and "curvatureV" to better follow naming conventions.
- Dynamic String Construction within any in-house patterns: XPU does not support dynamic string construction within Patterns, so if your Studio has any in-house Patterns that do this, you will need to pre-compute any strings you need.
- Dynamically Building Filenames for Textures: While XPU does not support dynamic string construction, we are adding the capability to specify primvar references within a filename to have the system dynamically reference a filename for you. This feature will be in a dot release for XPU, but is available in RIS in R24.0. Use the <primstr:varname> token for this.
- The contrast-v22 and variance-v22 adaptive sampling metrics continue to be deprecated. They will be removed in a future RenderMan release.
- In PxrUnified, volume render Oi returns single float value to an AOV when a color is expected. The transparencyAccum of PbsWavePayload is now a color. This allows PxrUnified to correctly output a colored Oi when given volume with colored extinction. Previously, the monochromatic transparencyAccum was promoted to colored Oi and this is wrong. It is impossible to correctly composite such volumes with only the alpha channel a correct colored Oi is required.
- RfK: The "prmanPattern" terminal type has been removed from material nodes.
- RfK: The widget is now automatically set to "null" for all OSL outputs, so there is no longer a requirement that shader writers add this to their output parameters' metadata
- RfK: Alembic_In_Prman has been removed as the multithreaded advantage it had has been supplanted by the new multithreading features in Alembic_In from Foundry
- RfH: All RenderMan node versions have changed to ::2.0. please convert your nodes with provided script.
- RfH: RenderMan errors are now warnings in Houdini
- RfH: Automatic conversion of PxrHSL, PxrVary, PxrSurface, PxrMix from old scenes to the new parameter names in RenderMan 24 rely on the 456.py script on houdini scene load rather than Houdini's opalias. This is to allow us to change parameter names and keep the old values. If you already have a 456 in your pipeline, you will need to add the update methods in order to convert those shaders.

More Details

More New Features in RIS

- PxrSurface
  - Now supports two lobed Henyey-Greenstein for single scattering. Existing scenes will need to migrate the "g" parameter to the "g0" parameter.
  - The new per-instance Attribute "grouping" "int id" [0-15] can now be used to greatly reduce or entirely eliminate biasing artifacts when using the subsurfaceResolveSelfIntersections flag on PxrSurface in conjunction with path traced subsurface. Objects that wish to participate should simply be tagged with a mutually unique set of grouping ids. Upon doing so, a new self-intersection avoidance scheme should help to eliminate biasing issues that lead to highly incorrect subsurface illumination results near close intersections between the geometry, and as a consequence the need to tune or set the trace bias attributes should be greatly reduced.
  - Avoid excessive opacity calls when a utility pattern is connected to PxrSurface where we've seen a more than 2x speedup in our production scenes when large numbers of assets were wired in this way.
- A new PxrRadialDensity pattern defines a spherical density gradient in a volume container.
- Color management support. Added basic ACEScg support. There is now a scene-wide control defining the primary colorspace (Rec709 or ACEScg). Lights, light filters and nodes with color space controls (PxrBlackBody, PxrHairColor, PxrTexture, etc) will obey this setting when set to "Scene". Patterns can override the source and destination colorspace if need be.
- You can now output AOV's in OSL via the closure debug("AOVName"). No other closures are supported, but arbitrary combinations of debug() are supported to provide AOV's. Passing them via get/setmessage() is not supported.
- Bump and normal map shadow terminators. There is a new parameter added to all the BxDFs to apply a technique that improves the the abrupt shadow terminator line that appear when strong bump or normal maps are used to emulate micro-geometry. If there is no bump map applied to the BxDF the result produced is exactly the same as it was before.
- Polygon shadow terminators. We have improved shadowing artifacts on coarse polygon meshes.
- RfH: improved user friendliness of cryptomatte in RfH
- RfH: apply materials to instance points via instancer
- it: Added a simple bloom filter to It
- it: Support OCIO color spaces

Bug Fixes

- We now correctly combine the two shadow exclusion subsets defined by Attribute "trace" "string shadowexcludesubset" and the "string shadowExcludeSubset" parameter on Light shaders that support this functionality.
- RfH: args2hda now supports Unicode characters in parameter/shader help
- Add a Version string to the rman python module.
- PxrUnified no longer clamps "numIndirectSamples" to max 1. Though if you are using path guiding, that code assumes one indirect sample, so leave numIndirectSamples at 1 in that case.
- More informative message when the alembic procedural fails to read a RLF file.
- RfM: Color and float ramp widgets are not enclosed in an extra group anymore, if they share the same label as the enclosing group
- RfM: The Pixel Variance fields have a five-digits precision instead of 3.
- Fixed a bug preventing connections to dynamically sized array input parameters in OSL shaders
- Fixed a crash when loading OSL shaders with dynamic array parameters.
- Fixed a bug where, in some cases, 'motionFore', 'motionBegin' and 'dPcameradtime' would be emitted w.r.t to a camera different from the one being rendered
- Fixed a bug where scenes with very high light counts could sometimes get stuck for a long period of time during initial scene processing

Known Limitations

Interactive/Live Rendering Limitations

- Bucket order or size cannot be changed during live rendering
- Changes to Presence do not update when using the opacity cache option
- Motion Blur will disappear during interactive rendering with scene changes
- Objects are not re-diced during interactive camera edits
- Mesh lights cannot be interchanged as geometry without restart.

RenderMan Pro Server

- Shading
  - PxrUnified integrator is currently experimental as it does not yet support all the standard rendering features.
  - PxrSurface back diffuse color is not output to the albedo color AOV
  - <primstr:nameofvalue> substitution of data from a constant primvar or user attribute on an object for dynamic filename substitution is not yet working for the gettextureinfo() OSL call.
  - Using the '**.**' character in the handle for an OSL shader could cause unpredictable results during re-rendering.
- Performance:
  - OSL startup. We have upgraded to OSL 1.10 for both RIS and XPU in RenderMan 24. We have found that this upgrade (and its dependencies) have slowed down time to start up for some scenes.
  - Volume integration with patterns. Because of the switch from C++ to OSL patterns and the sensitivity of volume integration to pattern execution, there are performance regressions with volumes in RenderMan 24 compared to RenderMan 23 that remain to be addressed. There are also memory increases that remain to be addressed. This is most noticeable when the volume density is connected to a pattern (e.g. PxrPrimVar); the performance regression can be avoided by avoiding a pattern connection if possible.
- Interactive instrumentation/statistics:
  - Bridge product (Maya, Katana, Houdini, Blender) initial integrations are available, but are turned off by default. Artists can turn on the statistics through the buttons in the UI.
  - Web browser integration is still to come and is now likely to arrive in a future version of RenderMan.
- General rendering
  - Load on demand procedurals are not supported anymore, all procedurals are now loaded immediately
  - We do not read point data from OpenVDB files
  - Per-Instance baking is not supported, only the reference instance.
  - 3d baking: no direct bake-to-ptex support.
  - PxrBakePointCloud cannot directly render ptex.
  - Sample/Display filter plug-ins do not have access to lighting services for light dependent effects, e.g. lens flare.
  - Adding new mesh light on existing geometry during IPR results in double geometry.
  - Motion blurred polygons do not motion blur normals when deformed. Use Subdivision meshes instead.
  - When attempting to access an array primvar, you must first check the size of the array primvar and allocate the appropriate space. Not doing so may lead to a crash.
  - Points and curves cannot be used as geometric lights.
  - Analytical lights placed inside volumes may yield artifacts when made visible to the camera. As a work around, the light camera visibility should be turned off, and a geometry with a similar shape should be used (visible to camera, invisible to transmission and indirect rays), with the proper emissive bxdf.
  - Indirect Path Guiding in the PxrUnified integrator causes a crash
- Bridge Products:
  - RfH: Soloing MaterialX Lama nodes in complex shading networks can give incorrect result

RenderMan XPU

- Please refer to the XPU section of the documentation for the current list of limitations.