RenderMan 23.0

Star Wars: The Last Jedi

Welcome to RenderMan 23.0!

Welcome to RenderMan 23. This release introduces improvements to the previous RenderMan in very significant ways.

Please dive right into the release notes below for more detailed information on the latest version of your favorite renderer!

New Features in 23.0

True Interactive Rendering — Use RenderMan like never before with the ability to continuously render while you work. See changes immediately and make decisions sooner while refining your artistic choices. RenderMan has the ability to update while artists model scenes, complete layout, perform look development, author light path expressions, and much more. We've worked hard to remove many of the restrictions that require users restart a render or wait for feedback and continue to find more ways to expand supported features.

Interactive edits are now possible on displays

- Dynamic editing of AOVs and LPEs. Add them, remove them, or edit them – all without restarting your Live Renders. This is perfect for lighters and TDs tweaking shots
- Dynamically change resolutions without having to restart your render – especially useful when using viewport renders with tools like Maya and Houdini
- Quicker artist feedback via a new capability to control how the first iterations are updated to the screen

Interactivity in heavy scenes

When making rapid edits to heavy scenes and this setting is non-zero, the renderer will try to reduce tearing by updating the whole screen progressively with a dissolve-like effect using a new option we call "decidither" but is exposed as *Interactive Refinement* in bridge products where possible to allow updates to the technique without changing the artist workflow.

Enhanced adaptive sampling and a new default

Important improvements to how RenderMan performs adaptive sampling have been made.

- The default approach has changed to how the adaptive sampler works. In the past, we used the notion of contrast changes in the pixel to determine when to stop sampling. This had the advantage that it was more memory efficient. However, it had the disadvantage that it would sometimes do the wrong thing in areas of significant change on a surface. The new approach uses statistical variance to determine when to stop sampling the pixel. The new approach will give more consistent levels of noise across an image (and across a variety of images) compared to previous versions of RenderMan. This means that you will need to put fewer renders back on the farm because they contain noisy regions
- Pixel variance tests are done in a perceptual space. This improves reliability of the sampler and makes it behave closer to what the human eye sees. This will result in more even noise across areas in the image with varying exposure
    - Testing in a perceptual space has allowed us to remove darkfalloff as a parameter that controls the new RenderMan 23 adaptive sampling metrics (though it does remain for our compatibility metric, "contrast-v22")
- The default value of the PixelVariance option is changing. For the old RenderMan contrast based metric, 0.005 was the default. 0.015 is the default for the new variance metric
- New controls for "exposure bracketing". To provide more flexibility during compositing, you can tell RenderMan to sample more in darker and lighter regions of an image so that you can change the exposure of the image without bringing more noise into the image. The units are in stops.

- If you load an old scene, double check the value of PixelVariance and adaptivemetric. 0.015 for "variance", and 0.005 for "contrast-v22". You will need to experiment with values other than these to find a good balance of render time and quality, you can find out more about this under Sampling
- You can choose the "contrast-v22" adaptive sampling metric to revert RenderMan back to its old behavior. This metric is deprecated and will be removed from a future release.

All of these enhancements will likely mean that you will need to rethink your adaptive sampling defaults if you tweaked them in an older version. Because minor tweaks may be made between the RenderMan 23.0 beta and the release, you may want to revisit your defaults again at that time.

Important Differences

- The plugin API has been updated, so all plugins need to be recompiled in order to function in version 23.x
- RenderMan has been updated to support VFX Reference Platform CY2018
    - C++14 is now the minimum requirement for developers writing plugins
    - GCC6.3 is now the minimum supported Linux compiler
    - OpenVDB has been updated to 5.2
    - Alembic has been updated to 1.7.11
    - Support for python3 bindings are available for 3.4 and 3.5
- RMANCONFIG option and its behavior have been removed
    - It is no longer possible to use the "RMANCONFIG" environment variable to override the configuration defaults file (${RMANTREE}/etc/rendermn.ini). Configurations are now customized using the "RMAN_CONFIG_OVERRIDE" environment variable. This variable specifies a directory where additional configuration files can be found, e.g. a site-specific rendermn.ini file
    - The search order for the "rendermn.ini" defaults file is as follows:
        - 1. ${RMANTREE}/etc/rendermn.ini
        - 2. ${RMAN_CONFIG_OVERRIDE}/rendermn.ini
        - 3. ${HOME}/.rendermn.ini
        - 4. ${HOME}/rendermn.ini
        - 5. ./rendermn.ini
- The RDIR option behavior, becomes RMAN_CONFIG_OVERRIDE
- The renderer no longer internally converts separate "facevarying float s" and "facevarying float t" primvars into a fused "facevarying float[2] st". As a result, any discontinuities in facevarying s will no longer automatically be propagated to facevarying t and vice versa. If your application still emits separate "facevarying s" and "t", for predictable texturing on subdivision surfaces, it is highly recommended that your application outputs "facevarying float[2] st" instead

More information can be found in the Developer Section of the documentation. These changes may impact your scripts, plugins, pipeline, and more, changes to your workflows in 23 have been kept to a minimum to allow easier migration.

Miscellaneous Changes

- We now support changes to min/max samples and variance during interactive rendering
- PxrBlackBody now supports Rec709 and ACEScg color spaces
- A crash caused by degenerate VDB volume data has been addressed
- Fixed a rare bug where the renderer could crash during expansion of procedurals if the procedural had an invalid transformation matrix
- The attribute for indexofrefraction has been removed, this value is taken from the material
- We no longer ship the experimental PxrUPBP integrator

Known Rendering Differences

- Fixed a bug in path-traced subsurface scattering with PxrSurface whereby the subsurface IOR was being ignored for part of the computation. With this fix, you may see slight differences in look the more that the ior you use departs from the old (incorrect) value of 1.5
- Spherical RiPoints using pointfalloff now have better lighting behavior when it comes to interpenetrating points. The calculation when the ray origin is inside a sphere has been amended to provide a more predictable lighting result

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

- PxrUnified integrator is currently experimental as it does not yet support all the standard rendering features.
- Meshlights cannot be instanced
- Load on demand procedurals are not supported anymore, all procedurals are now loaded immediately
- We do not read point data from OpenVDB files

- PxrSurface back diffuse color is not output to the albedo color AOV
- 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.
- Using the ' **.** ' character in the handle for an OSL shader could cause unpredictable results during re-rendering.
- 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.
- Deformation motion blurred volumes don't currently work with densityFloatPrimVar or densityColorPrimVar. You will need to use a PxrPrimVar node connected to densityFloat and densityColor instead.