RenderMan 21.0

Welcome to RenderMan 21.0!

Welcome to RenderMan 21. This release introduces improvements to the previous RenderMan in very significant ways.

There are many additions: New lights and materials from Pixar Animation Studios. We've also added amazing new studio light filters, new patterns, and improved displacement. In addition there are changes to APIs and new APIs for the developers. Workflows have been greatly simplified and performance enhancements abound.

There have also been important subtractions:

> Please note that the Reyes rendering architecture has been removed from RenderMan 21.0. RenderMan is now based on modern and improved pathtracing technology and C++ plug-ins. Scenes using old RSL lights, displacements, and imager shaders will fail to render or render black because of this change.

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

New Features in 21.0

New Pixar Materials

New PxrSurface material used by the Pixar Animation Studios in actual production is now included! This includes solutions for all imaginable effects and are physically based but artist-friendly. You may also combine and layer these materials for added flexibility and artistic control.

New Pixar Studio Lighting

New studio lights from actual Pixar productions are also included to compliment the studio workflow and materials. Improved light filters developed in production are included with the studio lighting package. The lighting services API has also been updated and the new lights take advantage with vastly improved performance. Scenes with over 1 million lights render more quickly and easily!We also reintroduce an improved P ortal Light for difficult interior lighting situations.

New Integrators

PxrOcclusion is a non-physical integrator that can easily and quickly generate occlusion renders for adding details in post compositing work. This integrator includes many of the controls artists require for tuning the look of their results.

Denoise Improvements

Continued improvements in performance of the Denoise feature improves speed and quality. Also, using Denoise with CUDA capable GPUs could reduce Denoise processing times compared to the CPU. Using the GPU requires CUDA 7.0 (compute capability 2.0 or later) and a capable graphics card with support. GPU denoising can be enabled by default with an automatic fallback to CPU denoising if no compatible device is available. You can specify the GPU to use by index and adding it to the appropriate .json denoise configuration file.

Texture and Point Cloud Baking

It's now possible to bake patterns into 2D textures or point clouds using RenderMan. This allows artists to simplify complex networks to a single result for rendering. This can improve performance by reducing the cost in evaluating these patterns. Point clouds may be converted to ptex as a post process.

Relative Pixel Variance

It's now possible to assign a different Pixel Variance threshold to individual objects to improve render quality on especially troublesome areas visible to camera rays.

PxrDirt

PxrDirt is a new pattern made for artists wanting to create useful effects like procedural dirt maps or weathering. With useful controls and the new PxrBakeTexture pattern, artists can quickly generate and use results instead of hand painting maps.

Transparent Alpha

Shader writers can take advantage of a new feature to accumulate transparent alpha along the path. This allows compositors to use the semi-transparent alpha in later compositing. For example: A car render could composite the final render onto a background without having to "bake in" the background to the windows.

Keep Rendering

Recently introduced: the ability to mark a final render as a checkpoint. This allows the user to pick up a "finished" image and continue to render. Using either an .ini setting or a RIB option, with the later overriding the former, you can render a "completed" EXR with the option to continue as if it's a checkpoint. If neither is present it defaults to off. The final image will essentially be just another checkpoint, rather than a slimmed down image:

```
/prman/checkpoint/asfinal [bool]
Option "checkpoint" "int asfinal" [0|1]
```

Additional Features and Improvements Include

- Thin shadow light options allow realistic colored shadows for transparent and refractive objects without needing the overhead of tracing photons.
- Volumes now support deformation motion blur and rendering performance for volumes has been greatly improved.
- Automatic computation of camera lens tilt and shift effects has been added with simple controls in the PxrCamera shader.
- A Deep OpenEXR Display driver has been added.
- Native OSL support is introduced and improves work flow for pattern creation. PxrOSL has been removed to avoid confusion.
- Ptex performance has been improved especially when caching and use multiple threads.
- Statistics output has been improved with more utilization granularity and output with checkpoint renders.
- Many other performance improvements for scenes with volumes, complex transformations, high tessellation settings, and more.

Important Differences

Reyes Rendering is Removed

- RenderMan is now based on modern raytracing techniques to create a simple path to beautiful images.
- Only the "raytrace" and "bake" hiders are retained in this release.

RSL is Removed

- The RenderMan Shading Language (RSL) has been deprecated. RenderMan will print a warning if used and return black or artifact.

EXR Environment Orientation

- txmake no longer manipulates input images to conform to the OpenEXR latlong specification. An input image is converted directly to an OpenEXR format latlong texture with no image manipulation. The -extraargs exrlatlong flag is no longer needed or supported.

RiPoints as Spheres

- RiPoints without normals are now treated as spheres when ray tracing. P and N values will be as if the point was a sphere. RiPoints cannot be used for shading models which require bump mapping or tangent vectors (i.e. anisotropic specular models).

Better Defaults for Rendering

- The default hider is "raytrace", default bxdf is PxrDiffuse, and the default integrator is PxrDefault.
- Trace displacements is now *on* by default.
- The shadingrate default has been increased to improve performance since shading quality is not affected.
- A default value for darkfalloff improves rendering performance for many scenes, especially those with dark areas that were potentially over-sampled.

Diffuse and Specular Depth defaults have changed

- Originally introduced as an option, PxrPathTracer now tracks the diffusedepth and speculardepth separately from each other based on which lobe of the bxdf is sampled. Previously rays that have both diffuse and specular contributions increment both the diffuse and specular depths. The new behavior is more intuitive and produces expected results when adjusting the trace:maxdiffusedepth and trace: maxspeculardepth attributes. This comes at the cost of, in some cases, effectively doubling the number of indirect ray bounces which can cause look differences and/or increase render times.

New Ri Calls

- RiDisplace

- RiDisplayFilter
- RiLight
- RiSampleFilter

Deprecated Ri Calls

- RiAreaLightSource
- RiDisplacement
- RiImager
- RiLightSource

New Options

- Hider "bake" has been added for texture/pointcloud baking

Deprecated Options

- Hider "jitter" has been removed as part of REYES
- Camera "focusregion" has been deprecated. Its use will now trigger a warning.
- Option "shading" "int directlightinglocalizedsampling" has been deprecated as part of an improved light sampling scheme.

New Attributes

- Attribute "dice" "float micropolygonlength" has replaced ShadingRate

Deprecated Attributes

- RiShadingRate
- visibility transmissionportal
- visibility usestransmissionportal
- dice nonuniform
- stitch traceenable
- trace importancethreshold
- trace emissionbias
- irradiance maxerror
- irradiance maxpixeldist
- photon estimator
- photon maxspeculardepth
- photon maxdiffusedepth
- photon minstoredepth
- stochastic sigma
- volume refinementstrategies
- volume depthinterpolation
- volume polygonizepartialvoxels
- volume incidentfallback

Miscellaneous Changes

- Changing offsetS/T of each of the X, Y, and Z tri-planar projections in PxrRoundCube to be 0. The old default was an offset of 1.
- By default, RenderMan no longer combines hits across different instances that share the same Bxdf into a single shading context. This is the more correct behavior if shading variation is desired between instances, but may lead to a loss of efficiency otherwise. Attribute "shadegroups" "attributecombining" may be set to "permissive" to re-enable the old behavior if the instances are guaranteed to be identical in shading.
- The intensity of IES profiles is now directly proportional to the number of lumens per lamp (we ignore input watts). If the number of lumens is not specified, we default to 1700 lumens, the equivalent of a 100W incandescent light bulb. Should you wish to revert to the old intensity computation, you can disable this through a new option : 'Option "user" "int iesIgnoreWatts" [0]'
- Heatmap statistics have been improved visually as well as their numerical results.
- Prman now issues a warning and avoids a potential crash when geometry has ST primvars with the wrong type.
- Improved error reporting when a -capture output or inline archive rib buffer exceeds the maximum size supported.
- Changes have been made to improve layering workflow with PxrSurface and PxrLayerMixer.
- The memory requirements for heterogeneous volume rendering has been substantially reduced, especially when using many volume samples during single scattering.
- LocalQueue is now part of the RenderMan distribution.
- An issue with using autocrop and OpenEXR images has been fixed.
- There is now a Delete() callback in the RixLightFactory API.
- Light plugins can now safely return NULL from RixLightFactory::CreateLight method.
- New AOV "float rawID" that passes through integer id values to the display driver unchanged. For example, using: Attribute "identifier" "int id" [4294699949] on an object will result in the display driver receiving a value that is -nan if the bits are interpreted as a float, but the original 4294699949 is interpreted as an unsigned int.
- Portal Lights will now return white if the DomeLight does not have a texture specified.
- PxrDistantLight now has emissionFocus and emissionFocusTint parameters.
- Removed some obsolete Reyes stats.
- Improvements have been made to the thread scheduling for dense procedurals and meshes.
- Displacement bound stats are now available in expert mode.
- OSL shaders can register a nodeid and classification through OSL metadata
- RiVolumes that use DSO plugins no longer use filtered evaluations. This improves performance in some cases.
- Legacy photon mapping has been removed. Please use the PxrVCM integrator instead.
- OSL no longer generates a warning when it fails to find an Attribute or Option value.
- The OSL getattribute() call now supports fetching primitive variables using the "primvar" object name and special PRman built-in variables such as the tangent vector "Tn" via the "builtin" object name.
- RenderMan now supports up to 12 user lobes in LPE. Previously this was limited to 8.
- The integrators are now responsible for managing the emission from camera visible lights. Previously this was handled by a bxdf.

Known Limitations

RenderMan Pro Server

- 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.
- Instances are not supported for baking.
- 3d baking: no direct bake-to-ptex support.
- PxrBakePointCloud cannot directly render ptex.
- No RixPTC/pointcloud API (so PxrBakePointCloud cannot read ptc files).
- 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.
- Camera visibility changes are not respected during Live Rendering.
- For PxrUPBP, If the light source is inside a volume, that volume needs to be defined as Volume “box”
- For PxrUPBP, To get a volume caustic, the object casting the caustic needs to have higher intersectpriority than the volume.
- For PxrUPBP, Overlapping heterogeneous volumes are not working yet. (However, overlapping homogeneous volumes do work.) This will be resolved in the future.
- PxrPortal may yield artifacts when generating photons for PxrVCM / PxrUPBP. This will be fixed in a future release.
- 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 have mesh lights attached to them.
- 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.