RenderMan 21.8

Welcome to RenderMan 21.8!

This release introduces the following improvements, fixes, and miscellaneous changes.

New Features

- A new OSL pattern, PxrWireframe, has been added

Miscellaneous Changes

- Checkpoints are now written to .part files during checkpointing and only replace the prior checkpoint once all of them have been successfully written
- Improved handling of NaNs in albedo channels when denoising
- Added iesProfileNormalization parameter to PxrDiskLight, PxrSphereLight, and PxrRectLight; this causes the IES profile to affect the shaping of the light but not its overall energy output
- PxrMeshLight now combines color temperature with the light color instead of replacing it
- Improved roundcurve intersection test to avoid cases where false self-intersections could lead to shadow bias. Added an option curveOldBias to revert to the previous behavior for those who need to maintain consistency on older assets
- txinfo now explicitly indicates bit depth

Bug Fixes

- Reduced memory usage of light filter disabling when the same state is applied to multiple procedurals
- Fixed a bug that could cause checkpoint recovery to fail when the "origin" Display parameter is used
- Added lightfilter linking memory to geometry stats in expert mode
- ReadRegion() for display and sample filters now works correctly with zmin and zmax channels as well as brightness remapping
- A bug in PxrBump bump mapping on polygon mesh surfaces with vertex normals would lead to an overly bumped surface instead of smoothness across facet edges. This has been fixed
- Fixed a bug that could produce invalid EXRs when checkpointing and some buckets are empty
- Fixed bugs in the PxrDispTransform node resulting in incorrect displacement direction for Zbrush and Mudbox vector displacement in scenes with transformations with positive determinant
- Improved convergence when using many lights, and fixed a bug that could cause incorrect lighting when using PxrRampLightFilter
- Incorrect geometric normal Ng on round curves with Orientation "rh" has been fixed. Ng now agrees with N
- The k_wavelength builtin is now properly set on transmission ray hit opacity shading contexts
- The option query 'RiHider:dofaspect' now works
- Fixed bug causing a potential crash when using volumes with a very large scale transform
- Prevent light leaking from backside of PxrDiffuse and PxrDisney surfaces with bump maps. This can cause darkening on bump-mapped surfaces in places where Nn is very different from Ngn
- The univariate version of GetPrimVar with derivatives now operates correctly on S and T requests when both S and T are specified as facevarying on a mesh
- Fixed a bug causing a small chance of hanging in renders using more than 32 lights
- Camera FOV motion blur now outputs the correct global scale, it was previously inverted
- Fixed a bug that could cause checkpoint intervals, exitat times, and logging timestamps to drift relative to wall-clock time
- Fixed bug causing potential crash when using Loop subdivision meshes with the __handleid primvar
- RenderMan now correctly handles OpenVDB grids
- Fixed a bug that could cause extremely large subdivision mesh faces to exceed allowable memory usage
- Fixed a bug causing potential for incorrect rerendering results when using an attribute edit scopename not containing special regexp characters. The performance of such an edit has also been improved for scenes with many objects or lights

Known Limitations

RenderMan Pro Server

- Ptex may break on Loop subdivision meshes that are visible in render and are destroyed by a plugin using the Subdiv API
- Rendering an alembic file with any instances will only render the original geometry. The procedural does not call ObjectBegin/ObjectEnd or ObjectInstance so all instances are ignored.
- Full LPE support is only available with the PxrPathTracer and PxrVCM integrators.
- Cryptomatte
    - Crop renders write the whole image with black padding. These will still be correctly aligned with the main render.
    - Single-scatter volumes show as an opaque volume envelope. Multi-scatter volumes work correctly.
    - Checkpointing is not currently supported.
    - Cannot be used with another sample filter such as the implementation of Maya's Image plane.
- Light Filters in the PxrVCM integrator do not affect or shape photon emission.
- The Cone Angle parameter for lights is incorrect, it does not match the input angle, this will be fixed in a future version.
- The PxrAovLight does not work properly with PxrUPBP.
- Analytical lights placed inside volumes may yield artifacts when made visible to the camera. As a workaround, the light camera visibility should be turned off, and a geometry with a similar shape should be used (visible to the 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”, RiVolume
- 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.
- 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.