RenderMan 22.4

Welcome to RenderMan 22.4!

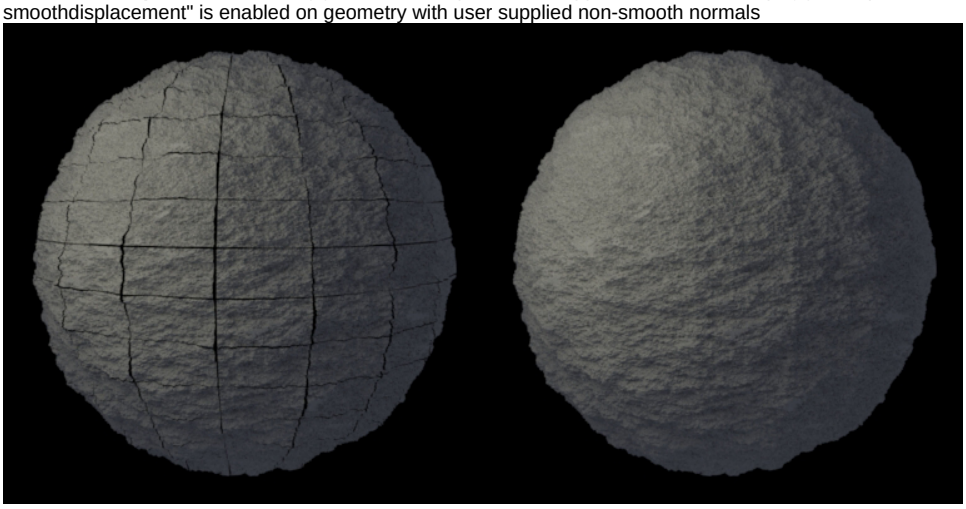
Welcome to RenderMan 22.4. This release introduces improvements to the previous RenderMan.

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

> RixRNG Generator class API change requires a recompile of Bxdfs and Integrators for 22.4

What's New

- **Further improvements to OSL performance** — A new mode of "Batched" OSL shading is available as a User Option to provide shading speedups when used on machines with Intel® Advanced Vector Extensions 512 (Intel® AVX-512). This may become defaulted to "on" in future versions. Read about it here under Performance Notes.
- **Crack Free Displacement on Polygons** — PxrDisplace now supports this crack-free polygon displacement when "int polygon: smoothdisplacement" is enabled on geometry with user supplied non-smooth normals



- **Smooth normals on displaced coarse polygon meshes** — This is implemented in the RixBump() utility function which is called by the PxrBump shader. This is basically just a new application of the classic trick of adjusting the post-displacement normal by the difference between the pre-displacement smooth interpolated vertex normal and the pre-displacement non-smooth (analytical) mesh face normal.
- **Volume Extinction Distance** — A new control for PxrVolume allows you to choose when a volume becomes opaque in scene units. This is very helpful in creating a homogeneous fog or mist for an entire scene. Below the lights become visible as the distance is increased.

- **New emissive features for PxrVolume and PxrMeshlight together** — In order to support advanced workflows where light filters or light linking are required, we now allow the PxrMeshLight light shader to be attached directly to a RiVolume for emissive effects. Next event estimation will apply for fast direct lighting. We still require a volumetric Bxdf (typically PxrVolume) to be attached. In this situation, the diffuseColor and the density parameters of PxrVolume replace the textureColor of the PxrMeshLight. All other PxrMeshLight parameters will be valid. Please see current limitations below

---

Miscellaneous Changes

- Additions to "it", the Image Tool
    - The Image Tool, "It" has a new "Commands > Standard Color Chart" menu to create a standard color rendition chart
    - The "Diff..." command in 'It' can now produce absolute valued difference images
    - Added a new Heat Map command to 'It'. This is useful to colorize float AOVs like sampleCount, difference images or CPUTime.
- Improved optimizations for large OSL networks that share computations across scatter, opacity, and volume evaluation for bxdf's that correctly categorize their inputs using extended RixSCAccess modes
- Updated PxrSurface to take advantage of extended RixSCAccess modes
- New optional iridescenceBumpNormal and glassBumpNormal parameters have been added to PxrSurface
- Exposed default (Ri) error handlers via RixInterface
- Restored C compatibility of ndspy.h
- Reduced overhead of accessing textures via OSL
- Iridescence now participates in the specularEnergyCompensation (its Fresnel, ie EdgeGain vs EdgeFace, is accounted for in the substrate's compensation, exactly in the same way and at the same level as specular and rough specular)
- Checkpoint recovery is now more verbose by default. Set /prman/recover/verbosity to 1 in rendermn.ini to return the old level. The new default is 3
- Added built-in "__faceindex" AOV
- PxrVolume now supports the user lobe "DiffuseAlbedo"
- PxrVolume has a new extinctionDistance parameter. It is useful to set a very low density in scene-wide volumes and only works with homogeneous float or color density
- Oren-Nayar (i.e. diffuse with diffuseRoughness > 0) is now more energy conserving
- PxrOcclusion now obeys Ri:Matte objects
- Improved shadow results from PxrShadowDisplayFilter
- The PxrDirt pattern and the PxrOcclusion and PxrDirectLighting integrators now ignore volumes entirely
- Displacement bound is applied more strictly producing more obvious and easier-to-debug clipping artifacts when misconfigured

Fixes

- Fixed possible crash related to long transformation paths
- Fixed the NP and Nl values provided to display plugins when using a worldorigin other than "world"
- Fixed a bug causing artifacts in LPE AOV outputs when rendering with more than one indirect sample in PxrPathTracer, in the presence of volumes mixed with glass or subsurface
- The built-in variable dPdtime is now correctly computed for transform blurred volumes which do not have velocity wired into their shading network
- Fixed an issue where many threads calling RiObjectBegin at the same time could lead to non-unique object handles being generated
- Fixed a bug that could lead to failure of portions of a shading network when using mixed C++ and OSL patterns
- Fixed a bug where deep images may have the negative values in the last channel in the channel list turned positive
- PxrUnified's adaptive progressive photon mapping feature, broken in previous 22 releases, now works again

- Fixed an issue where linear curves with motion blur could appear clipped or disappear completely if option curve:minwidth had a value greater than zero
- Avoid possible crash with stack allocations
- Equiangular sampling in PxrVolume is now correctly disabled when using an integrator that does not support volumetric single scattering (like PxrVCM or PxrUnified)
- Improved interpolation of normals in deforming polygons and pretessellated subdivision surfaces. Dark or inconsistent illumination could be seen on object edges when the deformation had a rotation component
- When PxrBump()'s surfaceNormalMix parameter is greater than 0.0, the resulting bump normal was not normalized. Now it is -- just like in all the other cases
- Subsurface objects which overlap volume envelopes now no longer render with artifacts (e.g. dark lines) that follow the edge of the envelope
- Fixed shading derivative calculation for volumes
- "it" now launches correctly on OS-X
- Varying opacity/presence now works on RiCurves
- Fix possible crash with long AOV names
- Fix rendering artifact in PxrDirt and PxrOcclusion with large numSamples
- Matte and Holdout attributes correctly compute alpha now with PxrPathtracer, regardless of BXDF
- Alpha for holdouts when using VCM integrator are now correct
- Fixed a bug where autocrop combined with negative crop windows could cause the EXR driver to crash
- Alembic archives now render with correct smooth normals
- PxrRoundCube was incorrectly scaling its texture filter size and was also impacted by another low-level bug in the renderer
- RixShadingContext::Transform was incorrectly returning radius^2 instead of radius

API Changes

- A RixRNG Generator class API change requires a recompile of Bxdfs and Integrators
- Added projection FOV query from RixRenderState::GetOption as Ri:ProjectionFOV
- The six RixRNG single-point sample generation functions DrawSample*D() and GenerateSample*D() now have an extra parameter 'i' which is the shading point index. (Same as in RenderMan 21, but with the index as last parameter rather than first.) Most Bxdfs and Integrators use the multi-point versions of these functions, so no change is needed there; nonetheless, this change requires a recompile
- Fields that are not valid in the RayCtxInfo struct have been removed. The RixRenderState::GetRayCtxInfo() method should be considered deprecated and will be removed in a future release
- The RixIntegratorContext::GetNearestHits() methods have been extended to allow volumes to be ignored. New versions of these methods have been added which simplify the optional parameters; the old versions should be considered deprecated
- Add RixIntegratorContext::k_MaxRaysPerBatch to API. This defines the maximum limit of numRays that can be passed to GetNearestHits and GetTransmission

Interactive/Live Rendering Limitations

- Crop window edits are restricted to fall inside the original crop window

RenderMan Pro Server Limitations

- PxrUnified integrator does not yet support all the standard rendering features
- PxrMeshlight as a volume will not update all scene lighting effects during IPR with parameter changes in 22.4 a restart is required, this will be resolved in a near future release
- 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 a new mesh light on existing geometry during IPR results in double geometry
- Motion blurred geometry does not motion blur normals when deformed
- 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 (mesh) 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

> ⚠ The Centos KDE style "Oxygen" installs a version of Qt and sets the user's environment variable QT_PLUGIN_PATH forcing "it" and LocalQueue to attempt to load an incompatible Qt library. Either avoid installing the Oxygen theme or unset QT_PLUGIN_PATH before running "it" or LocalQueue. Other KDE styles may also install this theme.