3.3.9.0

Milestone 3.3.9

IMPORTANT NOTE: This release has some rather significant changes to the SSS pointcloud generation code. Apart from the performance improvements and new sampling patterns, the sss_max_samples parameter has been removed and there is a great risk of old scenes performing very slowly or consuming large amounts of RAM because of this, i.e. pointclouds suddenly generating tens of millions of points instead of 100k points (which was the default value of sss_max_samples). If you were to encounter this, it is advised to increase the sss_sample_spacing until the number of samples is similar to the previous sss_max_samples setting.

Enhancements

- **Faster abort during SSS pointcloud generation**: The renderer will now poll for a render abort condition during the generation of the SSS pointcloud, which increases interactivity when computing dense pointclouds. (trac#2315)
- **Multi-threaded SSS pointcloud generation**: The blue noise SSS point cloud generator has been multi-threaded at the object level. This is achieved by spatially dividing the object and allowing different threads to generate points over each slice of the object. At low sampling rates, output may be different compared to previous releases, but quality remains the same and the results are deterministic as well as independent of the number of threads. (trac#2211)
- **Midpoint-based SSS pointcloud distribution**: Added a new sss_sample_distribution parameter to objects that allows the user to choose between the current blue_noise (default) sample generation method and two new midpoint-based generation methods called polygon_midpoint andtriangle_midpoint which place SSS pointcloud samples on the midpoints of either the mesh's polygons or triangles. The midpoint-based sample generation methods generate samples much more quickly than the classic blue noise method, but shading quality will be determined by the number of faces on the mesh. (trac#2214)
- **Pref-based SSS pointcloud distribution**: A mode called blue_noise_Pref has been added to the sss_sample_distribution enum that causes the SSS pointcloud generation code to use the varying user-data field called Pref (if it exists) instead of the polymesh's P vertices. This should result in fixed sampling patterns as long as the Pref data and mesh topology remain constant during animation. (trac#997)
- **Alpha composition for atmospherics:** Atmospheric shaders may now optionally use the sg->out_opacity field of the shader globals to indicate to the trace methods exactly how opaque the atmospheric effect is. This information, if available, is later used to compute the sample's opacity and alpha. (trac#2314)
- **Plugin symbols can be exported globally**: Shared libraries loaded via AiLibraryLoad() (which includes custom Arnold shaders and procedurals) can optionally export their symbols globally instead of locally. This global export capability works in Linux and OSX only, and it is triggered by renaming the shared library file from xxx.so to xxx.sog. (trac#2323)
- **-ibump option in kick**: Similar to how -idisp ignores displacement, we have added an -ibump option to the kick command-line, which directly sets the ignore_bump option in Arnold. (trac#2316)

API additions

- **AiShaderGlobalsGetVertexNormals/UVs()**: These new functions allow you to get the vertex normals and UV coordinates for the current triangle. See the Doxygen docs for more details. (trac#2331, trac#2329):

```
AI_API AtBoolean AiShaderGlobalsGetVertexNormals(const AtShaderGlobals *sg, AtInt key, AtVector n[3]);
AI_API AtBoolean AiShaderGlobalsGetVertexUVs(const AtShaderGlobals *sg, AtPoint2 uv[3]);
```

Incompatible changes

- **Renamed autobump to disp_autobump**: The autobump parameter, which was found in all geometric primitives, has been moved to the polymesh node, which is the only geometric primitive that support displacement mapping anyway. In addition, it has been renamed to disp_autobump, to better reflect that this is a displacement-related feature, along with existing disp_* parameters. (trac#2312)
- **Removed sss_max_samples**: The polymesh.sss_max_samples parameter, which placed a hard limit on the number of SSS points per object, has been removed. Because we no longer limit the maximum number of points, older scenes that were overdialed with a low sss_sample_spacingmay now take a long time to render, unless the number of points is readjusted to reasonable levels by increasing the spacing. A more detailed reasoning behind this change is as follows: when we first designed the SSS system, we thought it would be a good idea to have a safety valve to avoid memory explosion when assigning a subsurface material to a very large mesh. Instead, this parameter has caused a great deal of confusion with users wondering why the sss_sample_spacing has no effect sometimes, or by thinking they are setting the number of points directly (which they aren't when the spacing is non zero). (trac#2320)

Bug fixes

Ticket	Summary	Component	Owner	Priority	Version	Created
#2234	account for OIIO texture cache memory usage in the render statistics	arnold	marcos	major	3.3	2 months
#2325	'-set' option in kick doesn't override node parameters when using negative values	kick	oscar	major	3.3	3 days
#2327	'-set' option in kick doesn't override 'options.AA_samples'	kick	oscar	major	3.3	3 days
#2328	autobump doesn't filter texture maps	arnold	marcos	major	3.3	3 days