4.2.6.0

Milestone 4.2.6.0

Enhancements

- **Volume step size in object space**: Previously, volume step sizes were specified in world space, now they are in object space. This means that if you scale down a volume object, the step size will scale down along with it, maintaining the same sampling quality and speed. This affects both the step_size parameter on nodes, and the automatic step size provided by volume plugins. (#4609)
- **Volume low light threshold**: Volumes now take better advantage of options.low_light_threshold, giving around 5% render time reductions in various scenes. (#4610, #4622).
- **Volume stats**: We now report the average number of ray marching samples taken, as well as the average number of volume intersection segments found, per ray type. For shader call stats, "atmosphere" has been renamed to "volume". (#4607)
- **Built-in _self trace set**: Rays belonging to the built-in _self trace set will only trace against the current object. This trace set only supports inclusive mode. (#4628)
- **Updated maketx**: Updated the OpenImageIO maketx utility to 1.5.13. (#4591)
- **Searchable SDK**: The Doxygen SDK documentation for the Arnold core API now has a search engine and index for easier navigation. (#3848, #3218)

API additions

- **AtString string optimization**: Building off the internal work in 4.2.5, we now expose our AtString class for use in situations where a string is built once and used many times. AtString allows for very fast constant time string comparisons. The tradeoff is that creating an AtString from achar* string is an expensive operation. For this reason, AtString should be created in a preprocess step, such as node_update or with a static const AtString and not in performance critical code such as shader_evaluate. Many API functions that used to have a const char* as an argument now also accept AtString. Converting your code to use these overloaded functions should improve performance. (#4578, #4603, #4612)
  Here is an example of how the MayaRamp shader is made 1.07x faster with the below change to use AtString.

```
    AtPoint2 uv = {0.0f, 0.0f};
-   if (!AiStateGetMsgPnt2("maya_ramp_uv_override", &uv))
+   static const AtString maya_ramp_uv_override("maya_ramp_uv_override");
+   if (!AiStateGetMsgPnt2(maya_ramp_uv_override, &uv))
    {
```

- **Update method in volume plugin API**: The volume API now provides an update method, much like the node_update method for nodes. This method is optional and does not need to be provided. If it is provided, it can be used to quickly update volumes during an IPR session, without the volume being created and destroyed every time the scene changes. (#4614)

Incompatible changes

- **Raised Linux minimum requirements**: Running Arnold requires a system with at least *glibc 2.12* and *libstdc++ 3.4.13* (*gcc 4.4.7*). This is equivalent to RHEL/CentOS 6. (#4562)
- **Volume step size in object space**: Due to this change, volume objects with scaling in their transformation will now render with a different effective step size. If this was manually compensated for by changing the step_size on volume objects or step_scale on OpenVDB volumes, that must now be undone to avoid too long render times or too low detail. (#4609)
- **Volume shader and sg->P**: When evaluating a volume shader, sg->P can now be located at any point in the volume segment defined by sg->Ro and sg->Rl, rather than always being located at the center. We have not encountered any shaders that make this assumption and needed changes, but if any shader does it needs to be updated. (#4587)
- **Tagged unpremultiplied 8-bit TIFFs**: TIFF output images are now marked to have "unassociated alpha" in their header metadata when driver_tiff.unpremult_alpha is set to true. Applications that take this tag into account will now display Arnold "unpremultipled" images correctly. Note that this only affects 8-bit output images. (#4588)
- **Removed shader_nan_checks**: The NaN checks after each shader evaluation have been optimized so that they can always be enabled with no degradation in performance. From now on you will always get information about which specific shader node causes a NaN, not just in which pixel. (#2407)
- **Removed obsolete options**: A few global options that were rarely used and never exposed in our DCC plugins have now been removed: error_color_bad_mesh, AA_motionblur_pattern, enable_fast_lights, enable_fast_importance_tables, mindist_ulp_count, bad_bounds_slack,max_shader_messages. (#4597, #4600)
- **Procedural plugin abort handling**: Previously, if a procedural plugin had started running and the render was aborted or interrupted, the plugin's cleanup callback was not run. Now, the cleanup will always be called once the procedural starts running even during an abort. Not all nodes may have been requested yet, however, so the plugin writer should keep in mind that the AtProcCleanup callback may get invoked before the AtProcGetNode callback has been invoked as many times as expected. (#4625)
- **Restore previous signal handlers when crashing**: Arnold used to replace the preexisting signal handlers with its own handlers while it was running and after running it set them to the default signal handlers. If the parent process (e.g. Houdini) had set its own signal handlers, these handlers were permanently lost. Now we restore these handlers when Arnold finishes rendering. (#4621, #4629)

Bug fixes

Ticket	Summary
#4595	XSI/C4D hang on exit after rendering
#4449	Bump evaluation can request non existing UDIM tiles
#4552	crash when setting AOVs in background or atmosphere shaders
#4577	EXR driver append mode broken for some image resolutions
#4585	Crash with procedural DSO in second Arnold session
#4587	Volume artifacts at large step sizes
#4598	NaNs in alpha channel of float image textures without alpha
#4601	overrides on options not working
#4602	Matte objects still set AOVs when output type is RGB instead of RGBA
#4615	uninitialized shaderglobals privateinfo are common problems
#4621	restore previous signal handlers when arnold uninstalls its signal handlers
#4625	Render abort/interrupt leaves procedurals no chance to clean up
#4629	Pass signals from a crash to parent process