5.0.1.0

Enhancements

- **Improved mesh light sampling**: A new sampling algorithm has been implemented for mesh lights which can greatly improve their rendering quality when shading points near the light's surface, particularly in volumes. (#6005, #6062, #6074)
- **Indirect sample clamp**: The new `options.indirect_sample_clamp` parameter works similar to the existing `AA_sample_clamp`, but only affects indirect light, so that specular highlights from direct lighting are preserved. This makes it possible to clamp fireflies more aggressively without affecting the final render significantly. It is set to 10.0 by default. Lower values result in more aggressive noise reduction, possibly at the expense of dynamic range. (#5978)
- **Sharper quad and disk light spread**: Due to a more mathematically robust implementation, quad and disk lights now allow for more focused beams when their `spread` parameter is at zero. (#5980)
- **Faster BVH builds**: The BVH ray accel data structures are now faster to build, specially on machines with many cores. On a dual-socket 56-thread machine we've seen up to 2x faster builds and on a quad-socket 64-thread machine it's up to 3x faster. (#3474, #5951, #5962, #6001)
- **Polymesh as volume container**: The `polymesh` node now can be a volume container when `step_size` is greater than zero, and in that mode it will interpret the assigned shader as a volume shader. This is similar to other volume containers like sphere and box. A built-in channel named "density" is provided for shaders to query, which will be 1.0 on the inside of the polymesh, and 0.0 outside. To facilitate positional displacement in the volume shader, which is commonly used to add detail such as noise, there is an additional parameter `volume_padding` that will provide extra room. Note that large volume padding values, particularly when combined with high polygon counts (including through subdivision) will hurt performance, so only use the amount of subdivision and padding you need. (#6039, #6057, #6059, #6060)
- **Padding in volume-capable primitives**: In addition to the `polymesh` node, now `points`, `box` and `sphere` when in volume mode can have `volume_padding` applied so that positional displacement can be added in the volume shader. (#6055, #6059)
- **Iridescence in `standard_surface`**: Iridescence effects can be controlled by the new parameters `thin_film_thickness` and `thin_film_IOR` in the `standard_surface` shader, affecting the specular, transmission and coat components. This uses a new per-microfacet Airy formula, supports artistic Fresnel or complex IOR, and gives much more accurate results than the older `thin_film` shader. Typical thickness values are in the 0nm to 2000nm range. The effect will disappear above 3000nm. (#5750)
- **Exact Fresnel equation**: The exact Fresnel equation is now used for both microfacet BRDF and microfacet refraction BSDF instead of Schlick's approximation. (#5371)
- **Coat AOV**: New built-in AOVs were added for the `standard_surface` shader: `coat`, `coat_direct`, `coat_indirect` and `coat_albedo` AOVs. The builtin `specular` AOVs now only contain the base specular layer. (#6014)
- **Skydome Light AOV**: The skydome light is now output to direct light AOVs by default again as in Arnold 4. It can still be output to the indirect light AOV by enabling the `skydome_light.aov_indirect` parameter. (#6022)
- **Global AOV shaders**: A new list of shaders can be defined in `options.aov_shaders` that will be evaluated after the regular surface shader. With this it's possible to add shaders to set specific AOVs without modifying the original shader tree. Shaders intended for this purpose should add a boolean metadata named `aov_shader` on the node itself, as a user-interface hint. If `options.atmosphere` or `options.background` are set, these global AOV shaders will also be run for atmosphere and background contexts. (#5942)
- **Normal map shader**: A `strength` parameter was added to increase or decrease the effect of the `normal_map` shader, with the default 1.0 applying the normal map unmodified. The `normalize` parameter was removed and the output normal is now always normalized. (#5971)
- **Camera barrel distortion**: A new `persp_camera.radial_distortion` attribute has been added to specify the camera's quadratic radial distortion, with negative and positive values resulting in pincushion and barrel distortion respectively. (#2385)
- **Camera FOV EXR metadata**: As a convenience, a new field `CameraFov` has been added to EXR metadata to explicitly store the render camera's FOV. This is a lot easier than having to derive it from the film aperture and focal length with the formula `fov = 2 * Rad_To_Deg * atan(CameraFilmApertureHorizontal / (2*CameraFocalLength))`. (#6026)
- **Half-float EXR layers**: Output layers in files that support layers (such as regular or deep EXR files) can be individually set to type `HALF` by adding an optional `HALF` modifier to the corresponding output string. For instance: "my_aov RGB filter driver HALF". (#3839)
- **Custom EXR layer names:** Output layer names can be customized in file formats that support layers (regular or deep EXR) by adding an optional custom layer name after the driver name in the `options.outputs` string: (#6047)

```
outputs 2 1 STRING
 "RGBA RGBA gaussian_filter driver_exr beauty"
 "RGBA RGBA variance_filter driver_exr beauty_variance"
```

- **OCIO display / view**: OCIO Display/View tuples can now be selected as an output color space following Maya's convention. This applies to any input or driver node with a `color_space` string attribute. When using the OCIO color manager, if a color space name matches the format "VIEW_NAME (DISPLAY_NAME)" the given view on the given display will be used if they both exist. Note that conversions to a given display/view might not be valid for input textures, but they will always be available for output drivers. (#6069)
- **Weighted variance filter**: The variance filter has a new `variance_filter.filter_weights` to set the weights used for each sample. The default value (`box`) matches the previous behavior. (#5954)
- **Array setting with `kick`**: Array parameters of type `float` can now be set in the command line with `kick -set`. For example, `kick -set mynode.myarray 0.1 0.2 0.3 0.4` will create and assign a 4-valued array to `myarray`. (#6043)
- **Upgraded OIIO to 1.7.15**: We have upgraded to OIIO 1.7.15 which provides for several bug fixes and enhancements. Improvements not listed in our bug fixes section are (#6009):
  - PSD files now have support for cmyk, multichannel, and grayscale color modes and now supports 32 bit per sample bit depths.
  - Log statistics no longer list as "BROKEN" files which were invalidated or that were initialized but never actually opened.
- **Upgraded OSL to 1.9.0**: Open Shading Language has been upgraded to the latest version. This brings many performance improvements and bugfixes, along with new features such as operator overloading, struct initializers, and new utility functions. (#6034)

API additions

- **ASS file metadata**: New API functions are provided to store and read metadata from .ass scene files. Metadata is added as special comments at the beginning of the file. For now, all metadata types are stored and read as strings. (#3833)
- **Non-destructive `AiASSWrite()`**: It is no longer necessary to enable `options.preserve_scene_data` before writing a scene to an .ass file in order to be able to render it or write it again afterwards. (#3182)
- **`AiMakeTx`**: `AiMakeTx()` can now be run with an OCIO config, even outside an `AiBegin()` / `AiEnd()` block. Just specify `--colorconfig <path_to_ocio_config>` as an additional flag. All concurrent jobs must use the same config and convert to the same linear rendering color space. (#5786)
- **`AiShaderGlobalsGetShader()`**: Return root shader node assigned to the object at the current shading point. (#5956)
- **`AiShaderGlobals()`**: This function now allocates a more complete shader globals, so that when used with `AiTraceProbe()` user data is accessible. (#5956).
- **`AtHPoint` C++ math operators**: In addition to `AiV4Add()`, and associated math functions, operators +, -, *, and negation have been added in order to allow for cleaner code. A `project()` member function has also been added which can be used instead of `AiV4Project(`. The `AiV4CreateVector()` and `AiV4CreatePoint()` functions have now been overloaded so that the following is also possible: `AtHPoint hp = AiV4CreatePoint(my_AtVector);` (#5918)
- **`AiMicrofacetSetThinFilm()`**: This can be used to add thin film effects to the microfacet, metal and refraction BSDFs. It works by replacing the classic Fresnel term with a new Airy reflectance term, giving more accurate results than the `thin_film` shader. The microfacet normal is used instead of `sg->N`. (#5750)
- **`AiVolumeMergeIntersection()`**: Like `AiVolumeAddIntersection()`, this is used by a volume or implicit plugin to submit a ray interval for integration or intersection. However, it will merge with other extents submitted that match, so that any overlapping intervals are not integrated and shaded more than once. Previously a plugin would have to do this merging itself. (#6058)

Incompatible changes

- **Indirect sample clamp**: This feature is enabled by default (at 10) and in most scenes will have little impact besides reducing noise. For fully backwards compatible renders this parameter may be set to a very high value like 1e30. (#5978)
- **Exact Fresnel equation**: Due to the change of the Fresnel term, there will be differences especially when the relative IOR is close to 1, for example, when light travels from water to ice. (#5371)
- **OSL texture color space**: By default no automatic color space conversion is applied anymore in `texture()` calls. To revert to the previous behavior, use the optional `colorspace` argument set to `auto`: `texture("filename.jpg", u, v, "colorspace", "auto")`. (#6030)
- **`bounds_slack` renamed to `volume_padding`**: In volume nodes (those backed by plugins, such as the built-in OpenVDB volume node) the `bounds_slack` parameter has been renamed to `volume_padding` to be consistent with the other nodes. A synonym exists for `bounds_slack` so it will continue to work, but will emit a warning when used to remind you to switch to the new parameter name. (#6059)
- **`max_subdivisions`**: The type of the global `options.max_subdivisions` has been changed from `AI_TYPE_INT` to `AI_TYPE_BYTE`, to match the type of `polymesh.subdiv_iterations`. Existing client code should set this global option with `AiNodeSetByte()` to avoid a (harmless) warning. (#6029)
- **`light_decay` output type changed**: The shader output type of the `light_decay` light filter shader changed from `AI_TYPE_FLOAT` to `AI_TYPE_NONE`. It doesn't actually output anything, it just modifies the shader globals as do the other light filters. This makes its output type consistent with the other light filter shaders. Any shader networks that were relying on its output should be disconnected from it. (#6076)
- **OCIO config defaults:** For well known OCIO configs Arnold will use a name-based heuristic to determine reasonable default color spaces and chromaticities. These new defaults will be reported in the log file. This applies to ACES, nuke-default, spi and filmic-blender configs. Previously `Rec. 709` chromaticities were always assumed. (#6035)

Bug fixes

- #5927 Memory pools allocations are much bigger than requested
- #5639 Mesh lights in procedurals not working properly
- #5839 Metadata files don't allow boolean values as attribute or metadata name
- #5870 skydome_light and quad_light incorrectly detect some textures as constant color
- #5880 camera_projection : use NODE instead of STRING
- #5910 Missing Python bindings for some AtNodeEntry API functions
- #5911 Nodes contained in .obj and .ply procedurals are registered in the global name scope
- #5917 Remove pykick since it's broken
- #5923 UINT parameter values are clamped to 0x7FFFFFFF when parsed from .ass files
- #5925 Properly handle cases where cpuset is smaller than the detected number of cores
- #5926 ignore_motion_blur with non-zero reference_time does not work with a polymesh with normals
- #5930 normal_map shader issues
- #5931 Standard surface coat normal not decoupled from main normal
- #5936 AiNodeClone not working for parameter overrides
- #5940 AiNodeEntryGetDerivedType wrong for builtin procedural and implicit nodes

- #5946 Inconsistent reference_time between deform and transform motion blur
- #5947 AiSetAppString() not working
- #5950 Improve precision of box and plane objects
- #5953 Wrong stats after destroying shape nodes
- #5964 LPEs not working with raw drivers
- #5969 Avoid gcc -Wall warning about strict-aliasing in AtParamValue
- #5973 Curves with varying UVs rendering wrong
- #5977 Shadow matte shader missing transparency with passthrough shader
- #5982 Memory leak due to AiShaderEvalParamArray
- #5988 Avoid gcc -Wall warning about strict-aliasing in AtArray
- #5991 OpenEXR metadata not working for 3D vectors and RGB colors
- #5992 Color Manager: support chromaticities for additional standard gamuts / white points
- #5994 Color manager: wrong chromaticities for XYZ rendering color space
- #6001 Performance regression in BVH build
- #6015 Crash with oriented curves mode when no orientations are specified
- #6028 Rendering color space: use more accurate conversion matrix
- #6029 tighten max_subdivisions from 999 to 255
- #6030 Set OSL texture() default color transform to no-op
- #6031 OSL refraction closure crash
- #6036 Missing nodes not aborting render
- #6041 Color Manager: account for white reference when chromaticities are specified
- #6044 Color Manager: test and warn for degenerate rendering color space chromaticities
- #6045 Color management: review warning and error severity
- #6049 AtTextureHandle becomes corrupt after invalidating corresponding opacity texture
- #6053 AiNoise*() returns black when number of octaves is large
- #6076 Change light_decay output type to AI_TYPE_NONE
- #5899 Missing linkable metadata for some shader parameters
- #6080 trace() in OSL doesn't give the closest hit point