John Mertic <jmertic@linuxfoundation.org>

[TAC] Project Contribution Proposal: OpenImageIO

Larry Gritz <lg@larrygritz.com>
To: "tac@lists.aswf.io Calendar" <tac@lists.aswf.io>

Mon, Apr 10, 2023 at 2:04 AM

Name of the project (existing or proposed):

OpenImageIO

Requested project maturity level (select one): Sandbox or Adopted or Incubation

Incubation, on fast path to Adopted

Project description (please describe the purpose and function of the project, its origin and its significance to the ecosystem):

OpenImageIO (called "OIIO" for short) is a widely used library and utilities for VFX applications and pipelines to perform scripted manipulation of digital image files. OpenImageIO has been open source software since its inception in 2008 (the 15th anniversary of its first commit is April 21!). It is used ubiquitously in VFX, feature animation, and other areas of film production for all manner of manipulating digital images, and is embedded in many commercial products, proprietary tools in film production pipelines, and other widely used open source projects. OIIO is a popular open source project with over 1600 GitHub "stars" (which would make it the 3rd most starred ASWF project if admitted, behind only OpenVDB and OSL) and, at last count, has 190 individual contributors over the years.

OpenImageIO is the *lingua franca* for production software to deal with image data without needing to know the details of any of the file formats. At its heart is a simple, generic API for reading and writing image files in a way that is format-agnostic, and plug-ins for all the major image file formats encountered in motion pictures (such as TIFF, OpenEXR, DPX, Cineon, JPEG, DSLR raw formats, and many others). Particular attention is given to every obscure corner and edge case of these formats, robust handling of immense images and odd combinations (such as 37 channels of AOVs packed into one file, or tiled organization of the pixel data needed for efficient cached texture access), and careful shepherding of metadata.

On top of that functionality are library classes for manipulating entire images, for performing a wide variety of image processing and conversion operations relevant to film production, for cached out-of-core handling of immense image collections (such as is used for rendering texture systems, where many thousands of texture files totalling multiple TB may be referenced to render a single final film frame). The library exposes its functionality to C++ and Python, as well as providing command-line utilities for scripting. (The ASWF Rust WG also considers it a priority to provide Rust bindings.)

Uses of OpenImageIO range from the most pedestrian of image tasks, such as converting an OpenEXR file to a reduced-resolution JPEG thumbnail, all the way to handling all image input, output, and texturing for leading 3D rendering systems and other complex applications. The Python bindings and the command line `oiiotool` utility are used ubiquitously by major studios as a daily workhorse for file format conversions, resize and format conformance for ingest and delivery, color conversion (wrapping OpenColorIO functionality), and image processing. It is probably used somewhere in the generation of nearly every frame of most films that require any digital post-production.

Please explain how this project is aligned with the mission of ASWF?

As documented in the [VFX industry dependency spreadsheet](https://docs.google.com/spreadsheets/d/1EwRlz5ZYObEOdBflk8iTX5thlpTyEAfp3bxOgAfFOiU) and elsewhere, OIIO is a foundational project used directly across the VFX/animation industry, including by most (or maybe all?) ASWF member companies. OIIO is also a dependency of widely used major software packages, including Houdini, Maya, Katana, Blender, Arnold, Gaffer, Clarisse. It is also one of the earliest "VFX-originating" open source packages, dating from 2008.

OIIO is a close ecosystem cousin of many other ASWF projects. It uses OpenEXR, OpenColorIO and OpenVDB as dependencies. It is in turn a dependency of Open Shading Language, OpenColorIO, and MaterialX (and many other widely used non-ASWF projects including USD and Gaffer). Much of the actual use of OpenEXR and OpenColorIO in the real world is mediated / used through or by OpenImageIO -- many more people certainly use OCIO color conversion or EXR file reading/writing via OIIO APIs or utilities than by touching those packages' native APIs directly. Collaboration between these projects has been very fruitful historically, and I think it will be further enhanced by being under the ASWF organizational umbrella.

Amid this wide success, I fear that OIIO presents a broad risk to the industry because the growing criticality and foundational use of this this package in recent years has not been matched by an increase in active participation, steering, contribution, and diffusion of expertise across the stakeholders. I believe that OIIO is a victim of its own success: the fact that it is so widely used and seemingly so solid and production-hardened might generate the (incorrect) perception that it has all the resources it needs and is pretty much on auto-pilot. But this is far from true -- the project is much too dependent on the constant attention of its single founder. Even to the extent that he may continue to be a primary implementor of many internals, this is inhibited by the amount of time he must spend on things that could and should be offloaded or spread across multiple people.

Many routine tasks could be more widely distributed among stakeholders: issue triage, answering user questions, tending the CI and ever-changing dependencies and platforms, release management, documentation, fixing CVEs, and minor fixes. Additionally, there are several big tasks that are outside the current administrator's expertise that have long been neglected and desperately need the time and expertise of people who can take real ownership responsibility over those areas. Prominent among these are Windows, ffmpeg/movies, camera raw formats, Rust, and possibly other areas which each deserve a dedicated code owner and resident expert, but now get a tiny sliver of the maintainer's time and minimal expertise in these areas. The forward momentum of the project could increase considerably if the current maintainer weren't drowning in tasks that take time away from major feature development and new ideas. Additionally, more smart participants would lead to more and better major features and new directions than any one person would conceive of alone.

The project could also use many of the benefits conferred to ASWF member projects: neutral ownership, formal TSC membership and meetings, representation on the TAC, CLA management, access to the paid GHA runners, and general organizational help.

What does the project need from the community?

It is vital that this project not be admitted to the ASWF only to continue to be nearly 100% reliant on the administrator, who would then be further burdened by an additional weekly meeting, a barrage of non-coding TSC members making further demands on the project, and other bits of overhead stemming from the organization's standards and practices. The whole point is to make the project less bottlenecked on its founder, and to achieve a much more extensive active participation from across the stakeholder spectrum. Therefore, this application is contingent upon -- and begging for -- help from the ASWF TAC and member companies to step up with:

- Assembling a TSC comprised of people who can pledge to work on the project, not merely to attend meetings and ask for more things to be implemented. We should aim for the supermajority of the TSC members to be committed to be active developers, not merely advisors.
- Member companies spending some portion of their FTE requirements on this project if they are significant users of the package.
- Recruiting of expert code-owners willing to take responsibility for major areas that require ongoing familiarity with particular issues or dependencies, including: Windows, camera raw formats and use of libraw or alternatives, video formats and use of ffmpeg, Rust bindings, build/CI issues, GPU, color management.
- A roster of very part-time (but consistent) volunteers to rotate through helping with issue triage, answering questions on the mail list, tracking down bugs when CVEs are reported, release management.

If each of the 5 or 6 biggest companies among those who are critically dependent on this project were able to allocate somebody to consistently spend a consistent 20% of their time working in the OIIO code base, that alone could double or triple the development velocity of the project and significantly mitigate risk for everybody.

Extra hands would allow us to take on some really interesting future development, including:
- A TextureSystem/ImageCache implementation for Cuda/OptiX, giving as-close-as-possible APIs and features as the current CPU-only texture system, to aid GPU porting of the many renderers that use OIIO underneath for their texturing.
- GPU-acceleration of existing features, ideally automatically when available.
- Easier build/install: A really bullet-proof build system, including on Windows; possibly pre-built binaries; PyPI distribution.
- A long-overdue overhaul of how we deal with camera raw and movie formats.
- A big 3.0 release that is allowed to break compatibility and has the opportunity to jettison a lot of old cruft.
- Rust and C bindings.

- A hard look at performance, profiling, optimization.
- Machine learning -- are there current OIIO features that should be be using it? ML-based or ML-related functionality that makes sense to be in our collection of image processing functionality? better integration of OIIO into ML frameworks to allow their algorithms improved access/support of the types of image files we use in VFX? Ready for big new ideas here.
- All sorts of other things I haven't thought of!

What is the project's license for code contributions and methodology for code contributions. ASWF maintains [recommendations](https://tac.aswf.io/process/contributing.html) for contribution and licensing for hosted projects.

BSD-3-Clause for code, and CC-BY-3.0 for documentation.

What tool or platform is utilized for source control (GitHub, etc.) and what is the location (e.g. URL)?

Code repo: https://github.com/OpenImageIO/oiio
Secondary repo for example images used for testing: https://github.com/OpenImageIO/oiio-images

What are the external dependencies of the project, and what are the licenses of those dependencies?

All the dependencies and their licenses are documented here: https://openimageio.readthedocs.io/en/master/oiiointro.html#acknowledgments

The few dependencies that are incorporated directly into the OIIO source code or repo are all compatible liberal licenses -- MIT, BSD flavors, Apache 2.0, or equivalent, and some public domain code.

The many link-time dependencies mostly use those licenses as well, and additionally include some LGPL libraries which are compatible with our license because they are dynamically linked.

What roles does the project have (e.g. maintainers, committers?) Who are the current core committers of the project, or which can a list of committers be found?

- Chief architect / administrator / founder: Larry Gritz (affiliated with Sony Pictures Imageworks, though please note that OIIO is not an SPI-owned project)
- People with committer access:
    - Alex Wells (Intel)
    - Chris Kulla (Epic)
    - Ole Gulbrandsen (Sony Pictures Imageworks)
    - Thiago Ize (Autodesk)
    - Cliff Stein (Sony Pictures Imageworks)
    - Luke Emrose (Animal Logic)
- Additional people with committer access for historical reasons, but are no longer active in the project (we should probably clean this up and restrict write access to active contributors)
    - Jeremy Selan (Valve)
    - Robert Matusewicz (Frostbite)
    - Leszek Godlewski
    - Chris Ford
- Full list of all 190 contributors: https://github.com/OpenImageIO/oiio/blob/master/CREDITS.md

What mailing lists are currently used by the project?

Mail list: http://lists.openimageio.org/listinfo.cgi/oiio-dev-openimageio.org

What tool or platform is leveraged by the project for issue tracking?

https://github.com/OpenImageIO/oiio/issues

Does the project have a OpenSSF Best Practices Badge? Do you foresee any challenges obtaining one? (See: https://bestpractices.coreinfrastructure.org)

https://bestpractices.coreinfrastructure.org/en/projects/2694

- 100% at the "passing" level.
- 55% "silver"
- 22% "gold"

What is the project’s website? Is there a wiki?

- Main web site landing page: https://www.openimageio.org
- Documentation site: https://openimageio.readthedocs.io
- Developer/GitHub site: https://github.com/OpenImageIO/oiio

What social media accounts are used by the project?

none

What is the project’s release methodology and cadence?

- Major (potentially ABI-breaking) releases annually in August/September, coinciding with VFX platform deadline for the following year. (Though this is not a project listed in the VFX Platform.)
- Minor (strictly ABI-backward-compatible) releases on the 1st day of each month for bug fixes and non-breaking low-risk feature additions.
- Ad-hoc very occasional patch releases for critical bug fixes that can't wait until the next month.

Are any trademarks, registered or unregistered, leveraged by the project? Have any trademark registrations been filed by the project or any third party anywhere in the world?

No registered trademarks.

The name and a newly designed logo are used to identify the project.

--
Larry Gritz
lg@larrygritz.com