scikit-image

image processing in python

Gregory Lee

Quansight
July 15, 2021

Recent Highlights

0.18 release highlights (Dec 2020)

Extended 3D support (**skimage.filters.rank**, **structure_tensor**)

Improvements to region properties: multichannel support and user-defined properties

New features:

- rolling ball background subtraction
- iterative Lucas-Kanade optical flow
- Hausdorff distance
- etc.

New sample images (cells_3d, eagle, human_mitosis, vortex)

0.18.2 started providing ARM64 (AArch64) wheels

Recent Highlights

Highlights of upcoming 0.19 release (Summer 2021)

Improved single-precision support

User-configurable **channel_axis** for multichannel images

nD support in several geometric transform classes

Substantial performance improvements:

- Biharmonic inpainting, masked slic superpixels, nD resize, Canny features, label2rgb, etc.
- More functions release the GIL: e.g., all of **skimage.filters.rank**, **skimage.restoration**

New functions:

Normalized mutual information, new illuminants, Fourier-based filtering (Butterworth), blur metric

New gallery demos: 3D microscopy examples using interactive Plotly plots

Improved Contributor & Maintainer experience

Streamlined release process (automated wheel build and upload on tagging)

Thank you John Lee, Vinicius Cerutti, and Tania Allard!

Benchmarking on GitHub Actions using Airspeed Velocity (asv)

See upcoming Quansight Labs blog post by Jaime Rodríguez-Guerra

Documentation built and visible in each PR

Acknowledgements: Funding for this was provided by our CZI grant:

https://chanzuckerberg.com/eoss/proposals/gpu-acceleration-rapid-releases-and-biomedical-examples-for-scikit-image

1.0 Release Plans

Refined API with parameters names more uniform across subpackages

Less magic/surprises

- Don’t automatically rescale the range of inputs
- Don’t try to guess if images are color (the user must specify the **channel_axis**)

Remove **skimage.viewer** (encourage use of third-party tools like Napari)

More details at: https://bit.ly/skip-3 (Please provide feedback!)

We will provide a migration guide well in advance of release.

Upcoming Plans: Backends for GPU and distributed arrays

scipy.fft implements a backend mechanism

Allows use of cupyx.scipy.fft, PyFFTW, mkl-fft, etc.

SciPy is working on extending this approach to **scipy.ndimage**, **scipy.linalg** and **scipy.special**. (e.g., https://github.com/scipy/scipy/pull/14356)

We are also starting to explore a similar approach to backends for scikit-image to allow GPU-based or distributed computation via tools like cuCIM and/or Dask.