Contribution ID: **19** Type: **not specified**

Discussion

Wednesday 20 September 2023 14:00 (1 hour)

Submitted topics for discussion:

–

Thomas Vincent - Managing compression filters in the mid- to long-term

–

Erik Maansson - Run-length encoding for mostly black images

About ongoing work where a much simpler compression algorithm (customized run-length encoding at the application-level) may be faster than the built-in ones.

For multi-detector covariance analysis, e.g. between photoelectron energy and mass of ionic fragments of a molecule, it is necessary to save raw data (e.g. images and spectra) separately for each laser shot, rather than only saving the average over many laser shots. This can lead to higher rates of raw data than a typical computer or storage medium can handle at the required repetition rate of the experiment, unless suitable (lossy) compression is applied.

In my application, we use a CMOS camera to acquire images at 1 kHz from which angularly-resolved electron velocities can be determined. The largest square image size of 1024x1024 pixels gives 2.1 TB/s of raw data, which in my experience is too much to save locally, with or without available HDF5 compression libraries, by a single computer (via pytables in Python & numba, on an Intel Xeon E5-1620 v4 3.5 GHz from 2018). For mass spectra, sampled waveform data (ADC) is also acquired, but this is two orders of magnitude less data and therefore not setting the speed limit.

However, by knowing that our kind of image normally contains less than a few hundred bright spots (detected electrons), each covering a few pixels, it becomes worthwhile to find a more efficient (lossy) encoding that still maintains the scientifically meaningful information. After subtracting a dark image, pixels darker than a threshold value are therefore set to zero. Currently the resulting image is then passed to HDF5 for compression with LZO to about 1/10th to 1/50th of its raw its size.

The required 1 kHz continuous processing and saving rate is achieved by letting the camera bin groups of 2x2 pixels, so that the software only sees 512x512 pixels. It would be desirable to be able to use the full 1024x1024 pixels, and perhaps the standard compression algorithms are wasting CPU-time by trying to be “smart”when the main way that my mostly-black images can be compressed is to get rid of all the zeroes. I have begun implementing a run-length encoding scheme where a run of successive values below a user-chosen threshold are encoded by a negative value (the length of the run) in the array of signed 16-bit integers. Bright pixels remain as positive values. This yields several times higher compression ratios than LZO, and compiled to machine-code with numba (LLVM) it runs at a speed where it seems interesting to implement for full-scale testing in the acquisition program. HDF5’s variable-length array does not seem performant enough to store the compressed result from each individual image, so solutions concatenating the compressed form of many (or all) images will be explored.

Website

Session Classification: Day 2