Status report and outlook

Junjie Xia
Aug 23 2024, CIDeR-ML

Dataloader optimization

Recap:
At the end of the workshop, mentioned that the padding of “zero-hit” PMTs seemed to take longer than expected and was the dominant fraction of data arrangement while loading:

- Needs padding twice **per event**—for Q & T over ~10k PMTs
- The padding function took **~300 us** per hit, which was significantly longer than usual
- The 2 paddings per event took > 95% of the cached data processing time

Fixed the problem by removing an unnecessary `np.array(list)` casting:

- The new padding function takes **~30 us** per hit
- Tested over 10 photon shotgun files (30000 events), data processing time: 24.45 -> 2.03 sec

New WCSim data generation

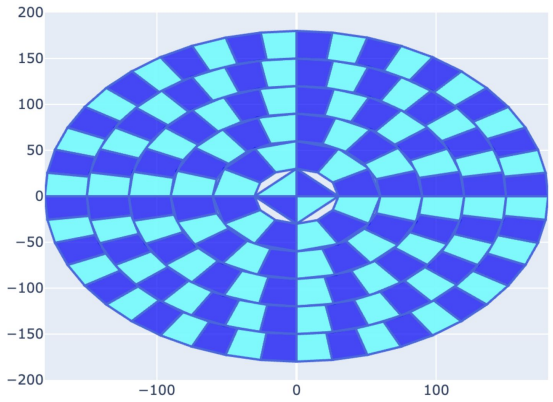
Recap:

- decided to run uniform and isotropic data generation in voxel volumes instead of on fixed grid points
- WCSim takes in the range of voxel vertices in (r, φ, z) coordinates

Expanded `wcprod` (https://github.com/seanxia8/wcprod_cider.git) to prepare database of this format (the grid point function still works), e.g.:

```
cfg=db.get_random_config('example_project_vox
_10000')
cfg
==================
{'config_id': 498,
 'table_id': 0,
 'r0': 120.0,
 'r1': 150.0,
 'phi0': 0.3490658503988659,
 'phi1': 0.5235987755982988,
 'z0': 90.0,
 'z1': 120.0,
 'file_ctr': 0}
```

To conserve the voxel volume, the range of φ depends on r and the the number of segments in the innermost circle.



Next step

1. Generate some toy WCSim voxelized data on s3df using “example_project_vox_10000” project, which means 10k photons per voxel

2. Run more diagnostics on the data loader with this toy set:

```
data =
db.get_project('example_project_vox_10
000')
print(data)
============
        Project name:
example_project_vox_10000
        Cylinder geometry
          R: 0.0 => 200.0
          Z: 0.0 => 400.0
        Gap space: 30.0
        Gap angle: 10.0
        Starting n phi: 4
        Sampling points: 2016
        Sampling directions: 1
        Sampling configs: 2016
        Photons per config: 10000
```

Next step

Data loading bottlenecks:

- The current WCSim data format doesn't support batch processing because of the varied data length of hit PMTs per event and the need of indexing each hit PMTs.

- The time to iterate over files **1 by 1, h5 dataset by dataset** scales linearly with number of files.
  - The enumeration over .h5 files has the following effects:
    1. Read in from disk and memory allocation (**O(100) msec** for an array, needs to do this 4 times per **file**)
    2. Rearrange the loaded data after padding and revise the allocated mem (**O(100) us** per **event**)

- The reason of this data format was probably to save disk->mem time from the 0 hits

- Plan to investigate the economy here: the balance among the memory usage, loading time, and the data processing. Move data augmentation to the pre-processing as much as possible.