Simulating flashers with clsim
by
Suruj Seunarine
Madison Bootcamp 2016

In the meta-project, named combo, on the bootcamp 2016 VM there is a project named clsim. This is a gpu based simulation for tracking photons in ice. For flasher simulations the relevant python scripts are in (or in the equivalent directory the simulation meta-project that you are using.

/home/icecube/i3_software/combo/source/clsim/resources/scripts/flasher

In there you will find four python scripts for doing all four steps of flasher simulations using clsim as described below.

1) Set up the flasher you want to simulate in the script generateTestFlashes.py Define the string number, DOM number, the flasher you need. Use the bit map with 0-5 for the tilted LEDs and 6-11 for the horizontal ones. cDOMs have only horizontal LEDs. See

https://wiki.icecube.wisc.edu/index.php/Flasher_LED_mask and
https://wiki.icecube.wisc.edu/index.php/Flashers

for more details on flashers.

```
tray.AddModule(clsim.FakeFlasherInfoGenerator, "FakeFlasherInfoGenerator",
        FlashingDOM = icetray.OMKey(57,30),
        #FlashingDOM = icetray.OMKey(36,22), # a cDOM
        FlasherTime = 0.*I3Units.ns,
        FlasherMask = 0b111111000000, # only the 6 horizontal LEDs
        #FlasherMask = 0b10101, # 505nm LEDs only (on a cDOM)
        FlasherBrightness = 127, # full brightness
        FlasherWidth = 127)      # full width
```

The script above is configured to simulate all 6 horizontal LEDs on DOM 30 on string 57. Simulate full brightness (127) and flash for the full duration (width of 127)

To run the script you will need a GCD file. You can get one here,

https://wiki.icecube.wisc.edu/index.php/GCD.

On the cobalt machines you can run the script with, for example,

python generateTestFlashes.py -n 1 -g GeoCalibDetectorStatus_IC86.55697_corrected.i3

This generates one flash (the -n 1 option) for the LEDs on the DOM configured in the script above. The output is an i3 file with the flasher information in it. The default output file name is test_flashes.i3.

2) Next you generate and propagate the photons using clsim. For this you will need to run on the gpu machines. See https://wiki.icecube.wisc.edu/index.php/Condor for how to submit gpu jobs or run interactive gpu jobs.

For interactive jobs run the script applyCLSim.py using the output file from the first step,

python applyCLSim.py -i test_flashes.i3

If you are doing this interactively exit the gpu as soon as this script is done running. This is the only step that requires a gpu.

The default output file is test_flashes_clsim.i3.

3) Now that you have photons that were propagated throughout the detector, you need to simulate the detector response. This step will result in InIceRawData being added to the i3 file.

python applyDetectorSim.py -i test_flashes_clsim.i3 -o test_flashes_clsim_detector.i3

4)Calibrate the raw waveforms to get pulses.

python applyCalibration.py -i test_flashes_clsim_detector.i3 -o test_flashes_clsim_detector_cal.i3.

The output i3 file will now have p frames with calibrated waveforms with at least OfflinePulses in them.