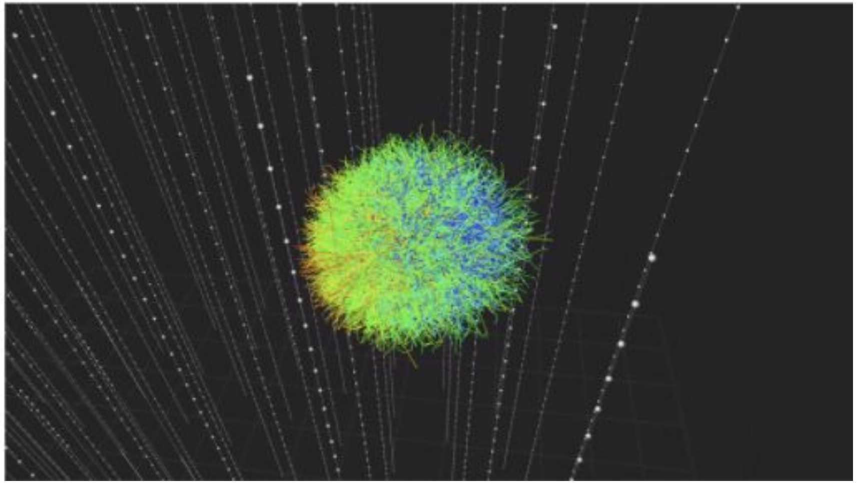
Clsim Overview

Photon Propagator Workshop
10/18/21

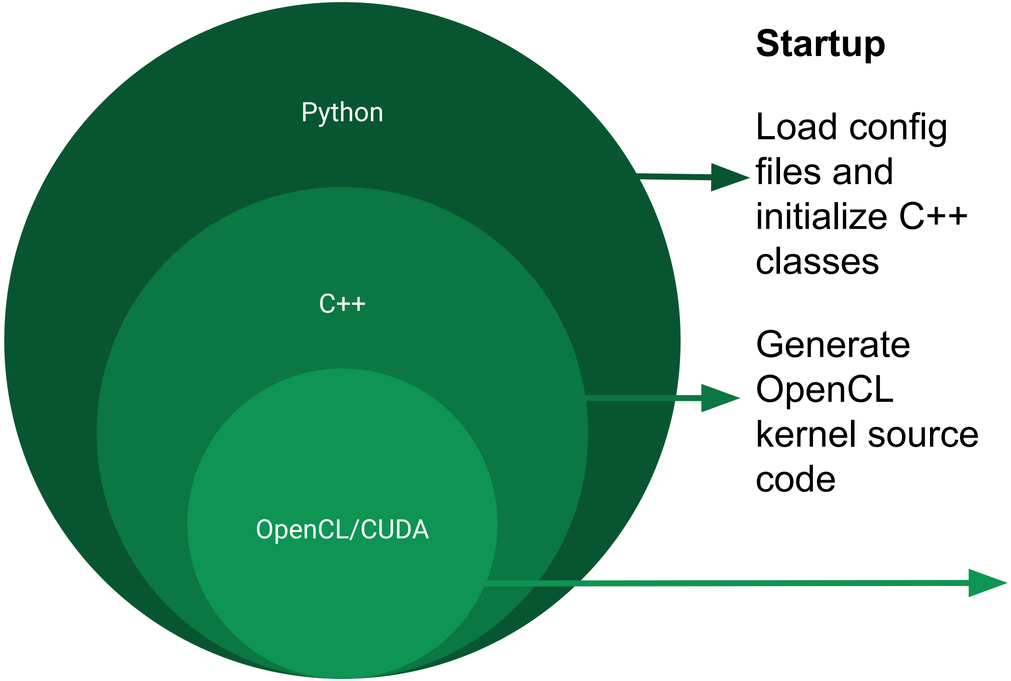
Alexander Harnisch



Outline

- High level code overview
- Feature overview compared to PPC
- NVIDIA collaboration
- Ice model implementation in clsim - A recipe
- Clsim incomplete tasks

High Level Code Overview



Startup	Runtime
Load config files and initialize C++ classes	Pybindings for C++ interface
Generate OpenCL kernel source code	Clsim server client runtime model
	Propagate photons

Clsim Server Client Model

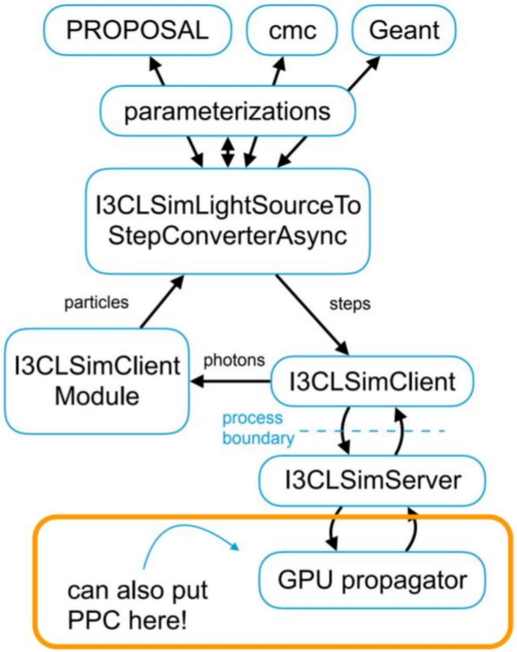


Figure stolen from Alex Olivas's slides

- Clsim is much more than just the propagator
- Server client model creates bridge between photon generation (client) and propagation (server)
- PPC can be used as propagator within clsim, but only with great care
  - Configuration does not work with the C++ interface, instead a temporary PPC config folder is created by the Python initializer which potentially erroneously changes settings without warning the user

Incomplete Feature overview

Feature	PPC	Clsim
SPICE Ice models	✔	✔
BFRv1	✔	✔
BFRv2	✔	❌ Missing absorption anisotropy
Server-client Icetray integration	❌ Only when run in clsim	✔
Ice model LLH fitting	✔	❌
Geant4 Integration	❌❓Only when run in clsim	✔
Direct Hole Ice, Cable Shadow	✔	❌ Almost there (Sebastian)
New 2d Tilt	✔	❌
Snowstorm Compatible	❌ Missing interface	✔
OptiX for collision detection	❌	❌ Branch not integrated
Triggered CORSIKA	❌❓	✔❓ Kevin

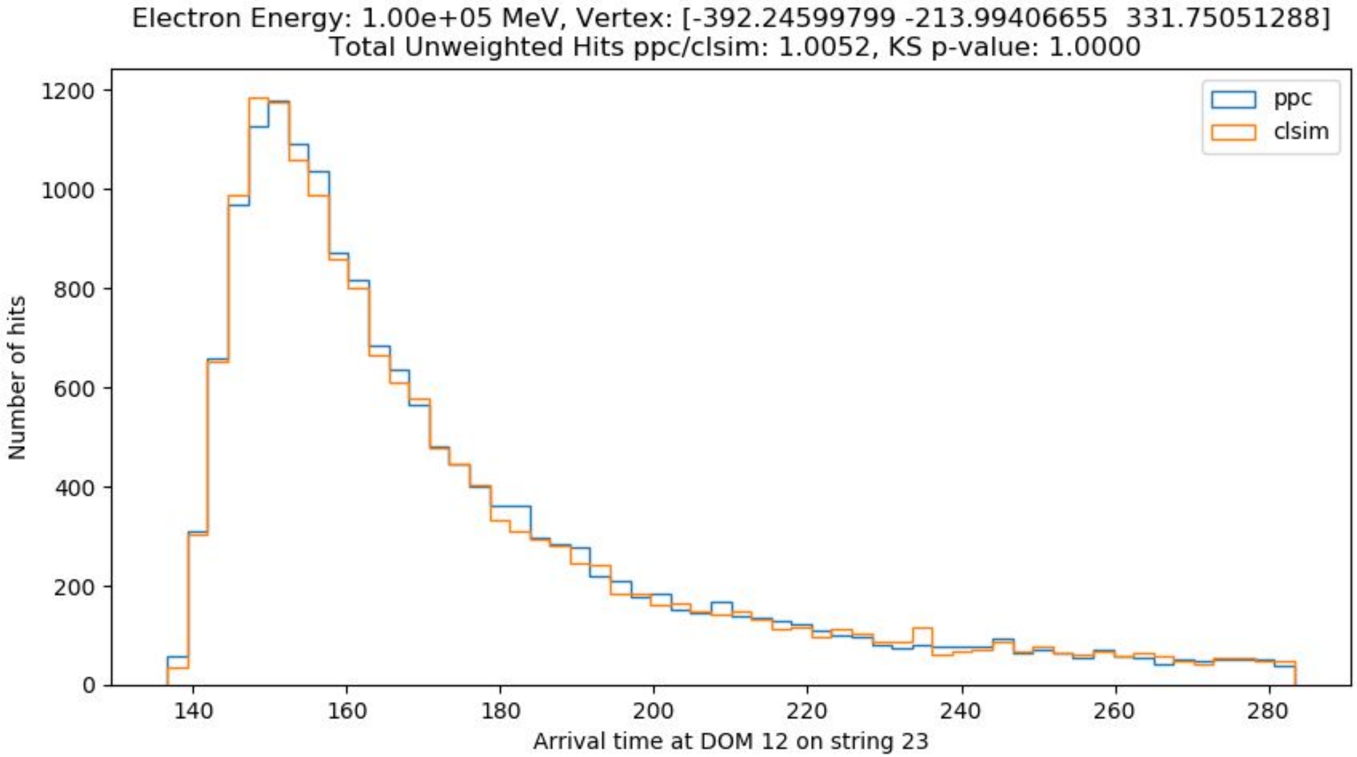
Collaboration with NVIDIA

- CUDA Kernel for clsim developed by NVIDIA master’s students
- SPICE 3.2 version integrated in main
    - **Significantly faster than OpenCL version**
    - Limited customization
- OptiX for collision detection
    - Workshop last week
    - More from Benedikt tomorrow
    - Not integrated into main yet
- Paper (peer-reviewed proceeding) submitted
    - Thanks Benedikt!

Ice model implementation in clsim - A recipe

New Validation Scripts

- New scripts to make sure PPC and clsim are doing the same thing
- In main under clsim/resources/scripts/compareToPPC
  - run propagate.py, then plot.py (ideally many times with randomized cascade position)
- Automatically generates histograms and TS test to check if the hit distributions match


Electron Energy: 1.00e+05 MeV, Vertex: [-392.24599799 -213.99406655 331.75051288]
Total Unweighted Hits ppc/clsim: 1.0052, KS p-value: 1.0000
ppc
clsim
Number of hits
0
200
400
600
800
1000
1200
140
160
180
200
220
240
260
280
Arrival time at DOM 12 on string 23

Ice model implementation in clsim - A recipe

New Model

1. The calibration group releases a new ice model in PPC
2. A developer volunteers to implement the model in clsim

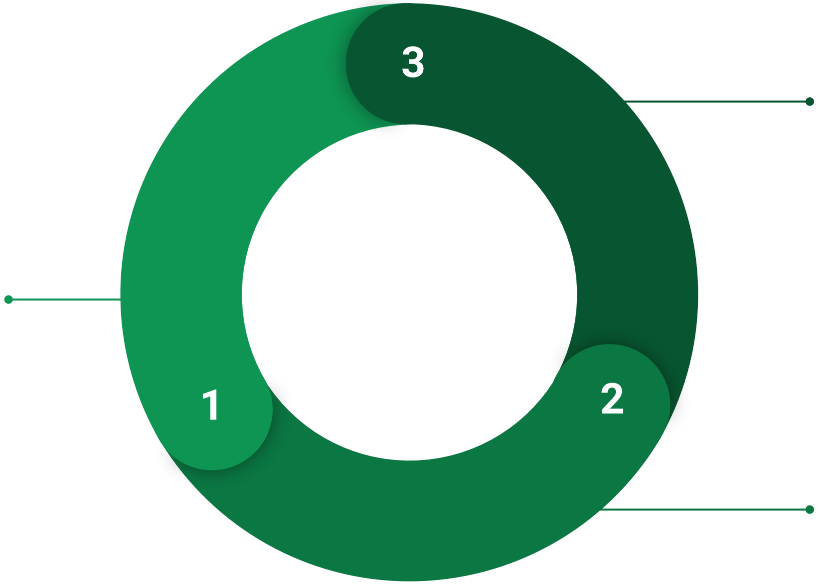
Preparation

3. Learn about the new ice model theory (somewhat optional)
4. Identify changes to the PPC source
   - Git diff is your friend
   - Dima is also your friend
5. Optional: Crank up parameters and quantify difference using the validation scripts

Iterative Implementation

6. Prepare first baseline by commenting out all changes to PPC until results match latest clsim version again
7. Perform iterative implementation (Next Slides)

Iterative Implementation



Baseline

PPC and clsim are doing the same thing (again). KS p-value is >0.99 and visual inspection of the histograms shows clear agreement.
Added print statements of new variables also show agreement.

Implement micro change in clsim

Make same changes to clsim until new baseline is established.
Usually much more work due to modular structure, configuration can optionally be postponed.

Uncomment micro change in PPC source

Re-activate the smallest possible coherent change in PPC. Validation scripts should show disagreement.
Optionally add helpful print statements of relevant variables.

Clsim Incomplete Tasks

- **Implement BFRv2!**
- Merge Sebastian’s direct hole ice/cable shadow

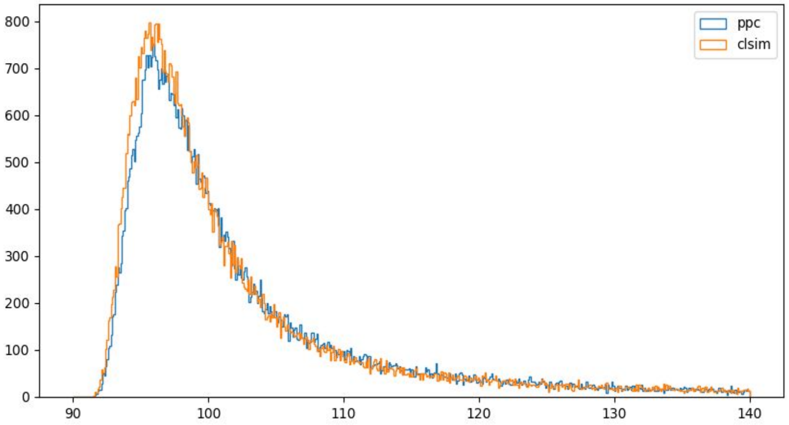
- Improve PPC integration?
  - Add more assertions/checks for ice model misconfiguration
  - Add Python/C++ interface for snowstorm?

- Integrate OptiX fork into main
- Improve CUDA version?
  - Run-time compilation?

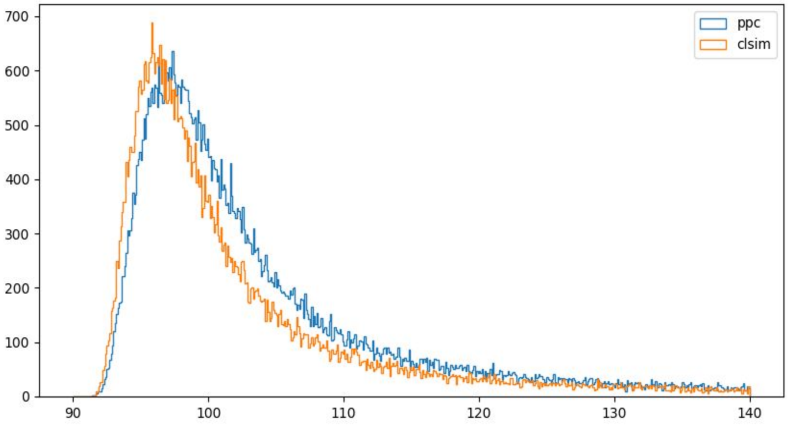
- **Decide how to move forward with development**
  - Maintaining CUDA + OpenCL branch seems unsustainable
  - Keeping up with PPC is tough, requires dedicated clsim ice model developer
  - Maybe focus efforts on improving PPC integration?

Backup

Example for Step 5



PPC with BFRv1, clsim with SPICE 3.2



PPC with **“cranked up”** BFRv1, clsim with SPICE 3.2