LArSoft light signal simulation

Alejandro Sánchez Castillo
asanchezcastillo@ugr.es

December 2022

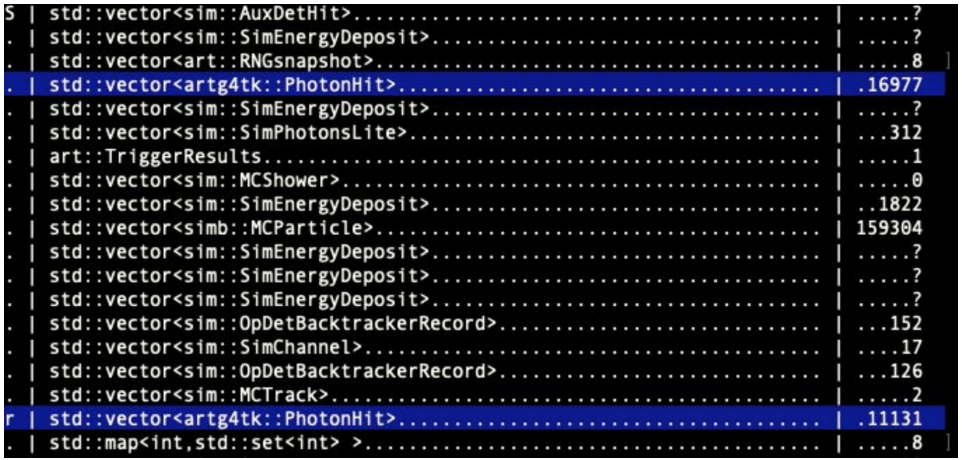
UNIVERSIDAD DE GRANADA

Reminder:

```
TrigReport ---------- Event summary ------------
TrigReport Events total = 1 passed = 0 failed = 1

TrigReport ---------- Modules in End-path ----------
TrigReport        Run    Success      Error Name
TrigReport          0          0          0 out1

TimeReport ---------- Time summary [sec] -------
TimeReport CPU = 3.707860 Real = 4.063928

MemReport  ---------- Memory summary [base-10 MB] ------
MemReport  VmPeak = 2337.51 VmHWM = 1274.04

%MSG-s ArtException:  PostEndJob 17-Nov-2022 06:25:50 CST ModuleEndJob
---- EventProcessorFailure BEGIN
  EventProcessor: an exception occurred during current event processing
  ---- ScheduleExecutionFailure BEGIN
    Path: ProcessingStopped.
    ---- OtherArt BEGIN
         ParticleList BEGIN
       sim::ParticleList::insert - ERROR - track ID=1 is already in the list
       The above exception was thrown while processing module larg4Main/largeant run: 1 subRun: 0 event: 1
         ParticleList END
    ---- OtherArt END
    Exception going through path simulate
  ---- ScheduleExecutionFailure END
---- EventProcessorFailure END
---- FatalRootError BEGIN
  Fatal Root Error: TTree::SetEntries
  Tree branches have different numbers of entries, eg sim::AuxDetSimChannels_genericcrt__G4. has 0 entries while Ev
entAuxiliary has 1 entries.
  ROOT severity: 2000
---- FatalRootError END
%MSG
Art has completed and will exit with status 1.
```

- We need to be able to run full light simulations in LArSoft.
- After the migration to the new LArG4 there was a bug preventing us from doing it.

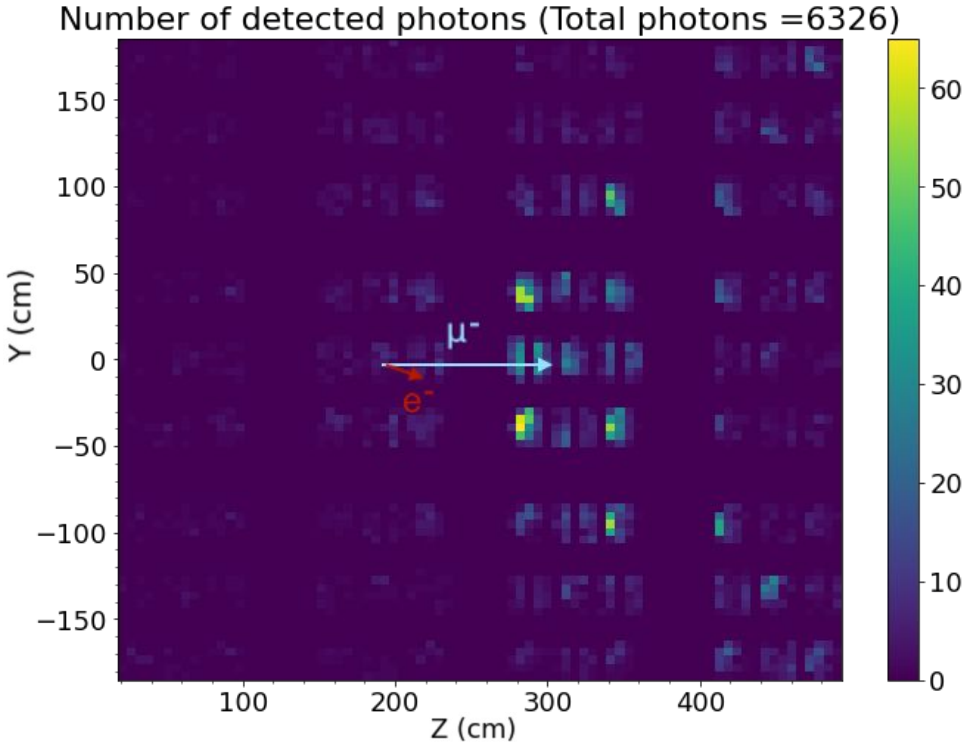
Update:



- We managed to find a provisional solution to run the first full simulations with the new LArG4.
- We can now generate, propagate and detect optical photons.

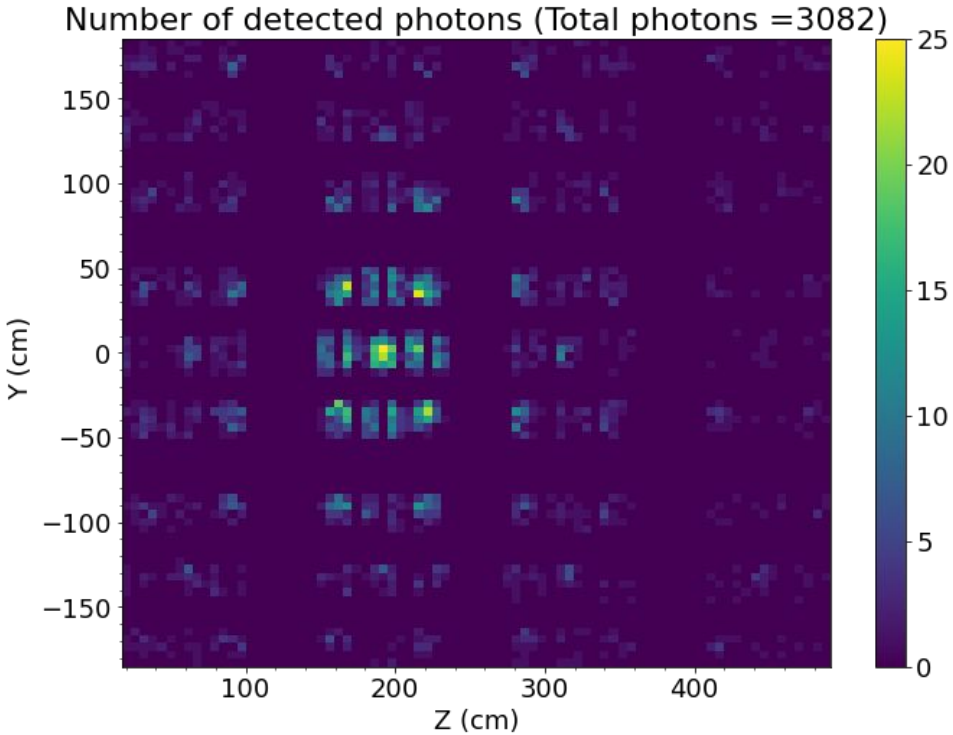
- PhotonHit is the object containing the true information of the detected photons which was previously empty and is full now.

LArSoft light simulation: Cherenkov light.



- Simulate only Cherenkov light of an electron (0.3 GeV) and a muon (0.03 GeV).
- High granularity, as we store the position where the photon hits the detector.
- We can differentiate PMTs and XArapucas.

LArSoft light simulation: scintillation light.



- Simulate only scintillation light of a low energy electron (0.03 GeV).
- Isotropic emission as expected.

Next task:

Public Member Functions

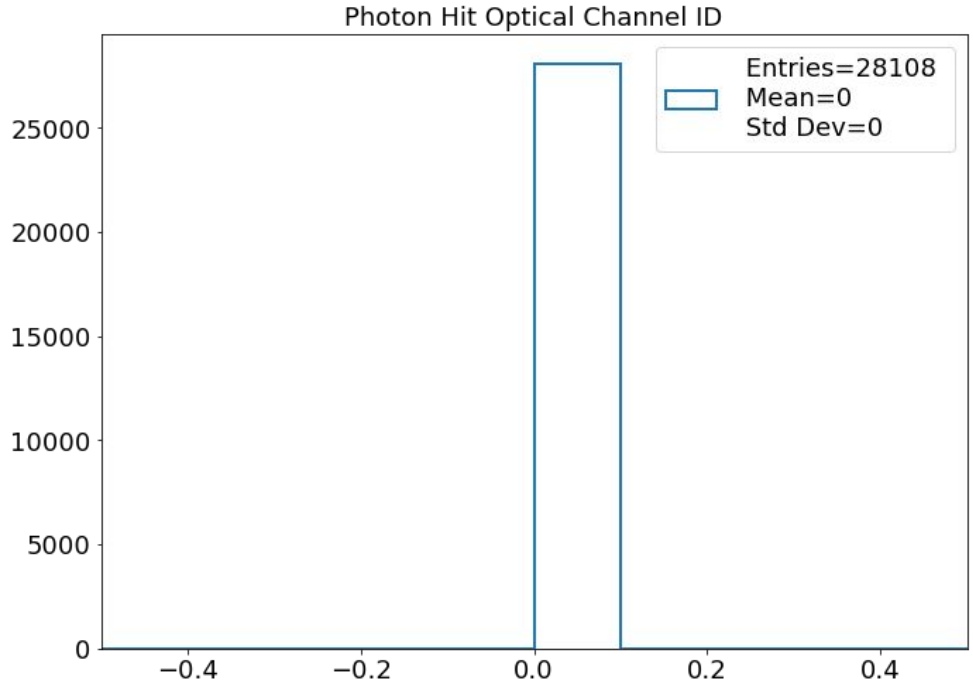
	SimPhotonsLite ()=default Default constructor (do not use! it's for **ROOT** only). More...
	SimPhotonsLite (int chan) Constructor: associated to optical detector channel `chan`, and empty. More...
SimPhotonsLite &	**operator+=** (const **SimPhotonsLite** &rhs) Add all photons from `rhs` to this ones, at their original time. More...
SimPhotonsLite	**operator+** (const **SimPhotonsLite** &rhs) const
bool	**operator==** (const **SimPhotonsLite** &other) const Returns whether `other` is on the same channel (`OpChannel`) as this. More...

Public Attributes

int	**OpChannel** Optical detector channel associated to this data. More...
std::map< int, int >	**DetectedPhotons** Number of photons detected at each given time: time tick -> photons. More...

- LArSoft does not use true hit information to digitize the signal of the optical channels.
- We need another object: SimPhotonsLite.
- This object contains the number of detected photons per time tick for each optical channel.
- Need a new module to translate from PhotonHits to SimPhotonsLite.

New hiccup:



- The current geometry makes copies of a generic PMT/XArapuca geometry without differentiating their IDs.
- We need their IDs to translate from PhotonHits to SimPhotonsLite.
- We have to change the geometry file and define one by one each PMT/XArapuca.

New hiccup:

```
<volume name="volPMT6">
        <materialref ref="LAr"/>
        <solidref ref="PMTVolume"/>
        <auxiliary auxtype="SensDet" auxvalue="SimEnergyDeposit"/>
        <auxiliary auxtype="StepLimit" auxvalue="0.01" unit="mm"/>
        <auxiliary auxtype="Efield" auxvalue="0."/>
        <physvol>
                <volumeref ref="vol_PMT_Back"/>
                <position name= "pos_PMT_Back " unit="mm" x="0" y="0" z="-51"/>
        </physvol>
        <physvol copynumber="6">
                <volumeref ref="volOpDetSensitive"/>
                <position name= "pos_PMT_Underside " unit="mm" x="0" y="0" z="-48.5"/>
        </physvol>
        <physvol>
                <volumeref ref="vol_PMT_in"/>
                <position name= "pos_PMT_Underside " unit="mm" x="0" y="0" z="-48.5"/>
        </physvol>
</volume>
<volume name="volPMT7">
        <materialref ref="LAr"/>
        <solidref ref="PMTVolume"/>
        <auxiliary auxtype="SensDet" auxvalue="SimEnergyDeposit"/>
        <auxiliary auxtype="StepLimit" auxvalue="0.01" unit="mm"/>
        <auxiliary auxtype="Efield" auxvalue="0."/>
        <physvol>
                <volumeref ref="vol_PMT_Back"/>
                <position name= "pos_PMT_Back " unit="mm" x="0" y="0" z="-51"/>
        </physvol>
        <physvol copynumber="7">
                <volumeref ref="volOpDetSensitive"/>
                <position name= "pos_PMT_Underside " unit="mm" x="0" y="0" z="-48.5"/>
        </physvol>
        <physvol>
                <volumeref ref="vol_PMT_in"/>
                <position name= "pos_PMT_Underside " unit="mm" x="0" y="0" z="-48.5"/>
        </physvol>
</volume>
```

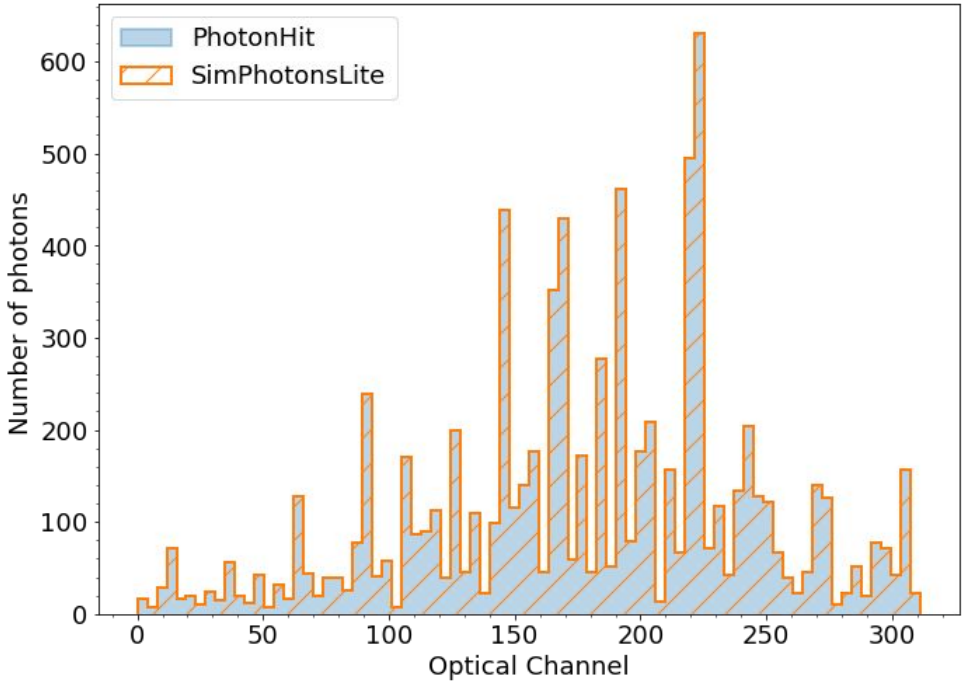
- New geometry file containing 312 independent optical channels with their corresponding IDs.
- We are ready to implement the new module that translates from PhotonHits to SimPhotonsLite.

New module:

```
void sim::HitLiteConverter::produce(art::Event& e)
{

   unsigned int nOpChannels = 312;


   std::unique_ptr<std::vector<sim::SimPhotonsLite>> photLiteCol{new std::vector<sim::SimPhotonsLite>{}};

   auto& photonLiteCollection{*photLiteCol};

   photonLiteCollection.resize(nOpChannels);

   for (unsigned int i = 0; i < nOpChannels; ++i) {
      photonLiteCollection[i].OpChannel = i;
  //    std::cout<< "Optical Channel" <<photonLiteCollection[i].OpChannel << std::endl;
    }

   // Implementation of required member function here.

   typedef std::vector<art::Handle<artg4tk::PhotonHitCollection>> HandleVector;
   auto allSims = e.getMany<artg4tk::PhotonHitCollection>();
    std::cout << "Im fine 4" << std::endl;

  for (HandleVector::const_iterator i = allSims.begin(); i != allSims.end(); ++i) {
    const artg4tk::PhotonHitCollection& sims(**i);
    for (artg4tk::PhotonHitCollection::const_iterator j = sims.begin(); j != sims.end(); ++j) {
      const artg4tk::PhotonHit& hit = *j;
      auto time = static_cast<int>(hit.GetTime());
      auto channel = static_cast<unsigned int>(hit.GetID());
      ++photonLiteCollection[channel].DetectedPhotons[time];
       }
  }
```

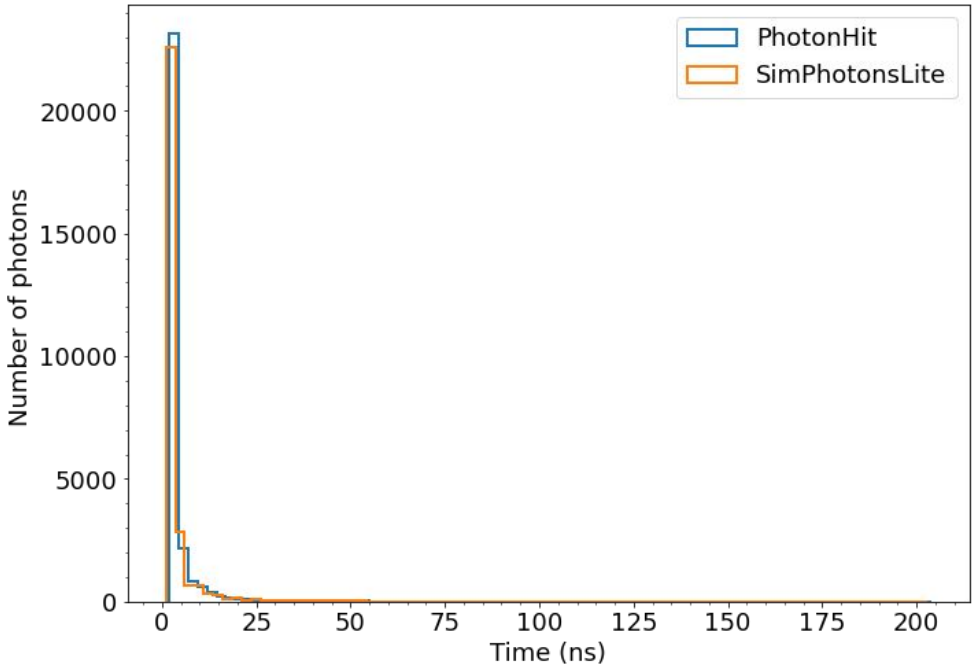
- New module: HitLiteConverter
- Reads PhotonHit information, creates SimPhotonsLite object and inserts it into the root file.

Module output:



- As expected the IDs distribution is identical (overlapped) for PhotonHits and SimPhotonsLite.

Module output:



- Time distributions are not overlapping because both objects have different sensitivity.
- SimPhotonsLite contains photons per time tick (1 TTick = 2ns).

Next tasks:

- Some minor things to deal with:
  - Find the best way to accommodate full simulations within the LArSoft workflow. We do not want to allow full and fast simulations to run at the same time.
  - Geometry changes require some tedious changes in the configuration fhicl and produce a lengthy terminal output.
- Contact Fermilab's expert to find a definite solution to the LArG4 issue.
- Check exhaustively the full simulation along with the new module:
  - Compare the result of the full simulations with the fast simulation and make sure it makes sense.