LArSoft - Feature #22628

Change internal representation of sim::OnePhoton

05/24/2019 01:18 AM - Gianluca Petrillo

Status:	Closed	**Start date:**	05/23/2019
Priority:	Normal	**Due date:**	
Assignee:	Gianluca Petrillo	**% Done:**	100%
Category:	Data products	**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour
Experiment:	-	**Co-Assignees:**	

Description

sim::OnePhoton is part of one of the two supported representation of scintillation light in LArSoft.
It is in fact part of the most expensive one, where each scintillation photon is saved individually as an element of a sim::SimPhotons collection.

Given that this is so expensive, it may be good to refresh and compact it.
The actions I propose are:

1. replacement of TVector3 with geo::Point_t in the two vector members
2. move of the bool data member at the end of the object

A survey in the code shows that the fixes after the first change are trivial.

Other actions that should take place at the same time:

1. ordering operator (ordering by increasing Time, then by MotherTrackID, InitialPosition (x, y, z) and FinalLocalPosition (x, y, z)
2. transformation into struct
3. removal of the empty constructor and explicit initialization of the data members

Associated revisions

Revision fe34d9da - 05/13/2020 06:21 PM - Gianluca Petrillo

Updates for new sim::OnePhoton.

Ripples from issue #22628.

History

#1 - 06/03/2019 10:33 AM - Kyle Knoepfel

- Assignee set to Gianluca Petrillo

- Status changed from New to Assigned

The proposal sounds sensible. Please proceed, place any code changes on feature branches, and plan to present at a LArSoft coordination meeting.

#2 - 05/13/2020 09:19 PM - Gianluca Petrillo

- % Done changed from 0 to 100

The following changes have been implemented:

- transformation into struct
- removal of the empty constructor and explicit initialization of the data members
- data types: two data members changed from TVector3 to geo::Point_t
- class layout: the bool data member moved from the beginning to the end of the class

I have not added the ordering operator (because I have forgotten).

The proposed changes have been implemented in branches feature/gp_issue22628 and are now under pull request.
The changes happened in:

- lardataobj: sim::OnePhoton class change *[pull request to LArSoft]*
- larsim: updates for the change (plus an unrelated optimisation) *[pull request to LArSoft]*
- ublite: updates for the change *[published as feature branch]*

The experiment code for ArgoNeuT (argoneutcode), MicroBooNE (the rest of uboone_suite), DUNE (dunetpc and protoduneana), SBND (sbndcode) and ICARUS (icaruscode) have been verified not to require any change.

Compatibility tests with data files from LArSoft v08_50_02 have been successfully performed with icaruscode as test bench.

For reference, from a ICARUS cosmic ray sample I have found the improvement in memory (VmHWM):

- from the change to the class (from TVector3 to geo::Point_t and change of layout): 8411 MB → 8062 MB
- from the side optimisation (writing of sim::SimEnergyDeposits data product): 8062 MB → 7790 MB

A serious speed up was also observed, but I honestly can't explain it, did not investigate it nor attempted to reproduce it.

#3 - 05/15/2020 02:37 PM - Gianluca Petrillo

I have fixes a major semantic issue: one of the two points subject of change, FinalLocalPosition, is not in world coordinates.
I decided to introduce a different type for that point, so that it can't be processed homogeneously with the world coordinate points (geo::Point_t).
There is some infrastructure for that already, so I have bound the new point type to the one used in geo::OpDetGeo.

As a consequence, the summary of the changes is:

- transformation into struct
- removal of the empty constructor and explicit initialization of the data members
- data types: two data members changed from TVector3 to geo::Point_t and geo::OpticalPoint_t
- class layout: the bool data member moved from the beginning to the end of the class
- addition of a sim::OnePhoton comparison operator
- geo::OpDetGeo now also uses type geo::OpticalPoint_t as local point type

The updated list of updated repositories, with branches feature/gp_issue22628, is:

- larcoreobj: optical vector type definitions *[pull request to LArSoft]*
- larcorealg: geo::OpDetGeo using optical vector types *[pull request to LArSoft]*
- lardataobj: sim::OnePhoton class change *[pull request to LArSoft]*
- larsim: updates for the change (plus an unrelated optimisation) *[pull request to LArSoft]*
- ublite: updates for the change *[published as feature branch]*

The experiment code for ArgoNeuT (argoneutcode), MicroBooNE (the rest of uboone_suite), DUNE (dunetpc and protoduneana), SBND (sbndcode) and ICARUS (icaruscode) have been verified not to require any change.

#4 - 05/18/2020 07:54 PM - Lynn Garren

I had to hunt down all 4 pull requests:

- larcoreobj PR 7
- larcorealg PR 6
- lardataobj PR 7
- larsim PR 17

#5 - 05/18/2020 08:04 PM - Lynn Garren

CI build triggered

```
trigger --build-delay 0 --cert /tmp/x509up_u1147 --version develop  --workflow default_wf --revisions "LArSoft
/larcoreobj#7 LArSoft/larcorealg#6 LArSoft/lardataobj#7 LArSoft/larsim#17 ublite@feature/gp_issue22628"
```

#6 - 05/19/2020 05:48 PM - Lynn Garren

There are two outstanding problems with this set of changes. First there are problems compiling with c7. Second, the uboone CI fails because the ublite feature branch has not been updated to match the head of develop, which means that there is a version mismatch.

#7 - 05/19/2020 08:01 PM - Gianluca Petrillo

Thank you for the feedback.

Clang error

I could not test the whole set of experiment packages at once with Clang: I am apparently failing to convince Cmake to use Clang: it seems to pick up GCC 8.2.0.
So I am sticking to testing only LArSoft and MicroBooNE code base.
A single error was reported (inconsistent use of struct and class for the same object), which is now fixed.
I have updated the feature branch in lardataobj.

MicroBooNE error

I don't fully understand the problem. I have found that my local GIT branch was not tracking a remote one, so I am not positive if the branch for ublite was actually pushed. I have pushed it now, to be safe.
Beside that, I don't know what else is needed: none of the branches is updated with the head of develop: they are based on LARSOFT_SUITE_v08_51_00 and UBOONE_SUITE_v08_51_00, respectively.

#8 - 05/19/2020 10:10 PM - Lynn Garren

The branch for ublite needs to be based off the head of develop. Otherwise the CI test simply fails to build due to a mismatch in versions of dependent products.

I used the CI to build everything for c7. A link to the CI output is in the PR comments. Thank you for fixing the problem.

#9 - 05/20/2020 01:06 AM - Lynn Garren

c7 CI tests are now OK. I updated the ublite feature branch myself.

#10 - 05/21/2020 04:01 PM - Lynn Garren

These PR's are now in larsoft v08_53_00.

#11 - 05/26/2020 01:23 PM - Kyle Knoepfel

- Status changed from Assigned to Resolved

#12 - 06/01/2020 10:35 AM - Kyle Knoepfel

- Status changed from Resolved to Closed