LArSoft - Bug #7460

NaN's in the simulation

12/09/2014 08:48 AM - Andrew Blake

Status:	Closed	**Start date:**	12/09/2014
Priority:	Normal	**Due date:**	
Assignee:	Gianluca Petrillo	**% Done:**	90%
Category:	Simulation	**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour
Occurs In:		**Co-Assignees:**	
Experiment:	LArSoft		

Description

When running the LBNE 35t simulation and cheated reconstruction in LArSoft v03_04_04, I observe the following kind of exception in every few events:

```
%MSG-w HitCheater:  HitCheater:hitcheat 30-Nov-2014 14:05:25 CST run: 1
subRun: 0 event: 21
caught exception
---- Geometry BEGIN
Can't find Cryostat for position (nan,nan,nan)
---- Geometry END
when attempting to find TPC for position move on to the next sim::IDE
%MSG TimeModule> run: 1 subRun: 0 event: 21 hitcheat HitCheater 0.132391
```

I traced the problem back to the SimChannel::AddIonizationElectrons(...) method. Sometimes this method receives an input of zero electrons, which can generate divisions by zero when calculating the average position. Possible fixes are: (a) return from the method straight away if the input is zero, (b) protect against divisions by zero, (c) throw an exception if the input should never be zero. In the case of (c), there might be a deeper problem in the simulation that needs to be addressed.

To study the problem (sorry, not very sophisticated), I compiled a cout statement into the above method, printing out a warning message whenever there are zero electrons. I then ran the standard prodsingle_lbne35t.fcl script from lbnecode. I saw several of my warning messages in every event! It takes two instances to generate a divide-by-zero error.

Associated revisions

Revision 930e1684 - 12/09/2014 08:13 PM - Gianluca Petrillo

Fix to Issue #7460: do not add ionization contributions with no electrons to SimChannel's

History

#1 - 12/09/2014 09:06 AM - Brian Rebel

I am surprised that there is ever a case where no electrons are passed to the AddIonizationElectrons method. The reason being that any energy deposition creates ionization electrons. I guess it is possible that those electrons are attenuated away before reaching any wire planes, but that seems unlikely.

Can you provide the exact command line you used to produce the problem?

#2 - 12/09/2014 10:29 AM - Andrew Blake

Here's the command line:

lar -c prodsingle_lbne35t.fcl

I compiled a print statement into the SimChannel::AddIonizationElectrons(...) method:

if (numberElectrons == 0) std::cout << " *** THIS IS BAD *** " << std::endl;

It crops up in every event!

#3 - 12/09/2014 02:20 PM - Gianluca Petrillo

- Status changed from New to Assigned

- Assignee set to Gianluca Petrillo

I can reproduce the problem with the 0 electrons.

#4 - 12/09/2014 08:15 PM - Gianluca Petrillo

- Status changed from Assigned to Feedback

The source of the problem seems a voxel with very small deposited energy.
That may be below the threshold to produce even just one electron.
Or recombination can make it so.

I have implemented a check to avoid the NaN problem.
It is committed as commit:d0019afd77cce301bd6f832d6575dd94a33a985c in feature/Issue7460 branch.

I would still like to discuss the problem from a physical point of view: if this fractional energy is real, maybe it should be made into fractions of electrons (this is a funny concept that our code already supports).

#5 - 12/09/2014 08:16 PM - Gianluca Petrillo

- % Done changed from 0 to 80

#6 - 12/10/2014 08:50 AM - Andrew Blake

Gianluca's fix to LArVoxelReadout works as expected. Running the simulation from the Issue7460 branch, the number of electrons passed to SimChannel::AddIonizationElectrons(...) is now always non-zero, and the NaN's have gone away in the cheated hit creation.

I think I'd suggest adding some additional protection to AddIonizationElectrons as well, in case the simulation code is ever changed in a way that allows zero electrons to propagate through the SimChannels again. Perhaps just bailing out at the start of the method if there are zero electrons:

```
if (numberElectrons < std::numeric_limits<double>::epsilon())  {
// Print warning message here...
return;
}
```

#7 - 12/10/2014 08:38 PM - Gianluca Petrillo

- Status changed from Feedback to Resolved

- % Done changed from 80 to 90

- Experiment LArSoft added

- Experiment deleted (LBNE)

I appreciate the suggestion; I have implemented such a warning as temporary, and as such it is going to be added in the next release.
It will be replaced by throwing an exception as soon as I figure out which exception is proper to throw (STL, CET or art).

#8 - 03/03/2015 10:22 AM - Lynn Garren

- Status changed from Resolved to Closed