

LArSoft and Vertical Drift geometry : towards a drift along Y axis

T. Kosc – 10th October 2022

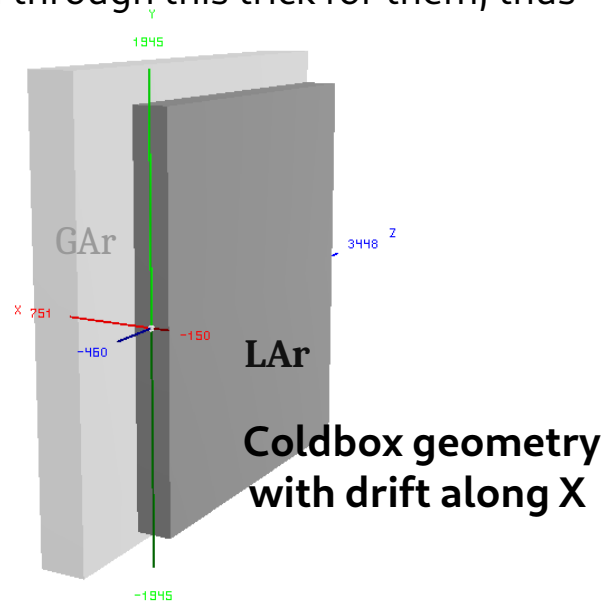
Meanwhile ...

- I was a bit off during september due to my newborn child Maël !
- Back to business only last week (but I was at CERN anyway).
- I gave this talk in protoDUNE-VD meeting last 23th September : <https://indico.fnal.gov/event/56317/>
- Wenqiang created a dedicated slack channel, please let us know if you want to join the effort.

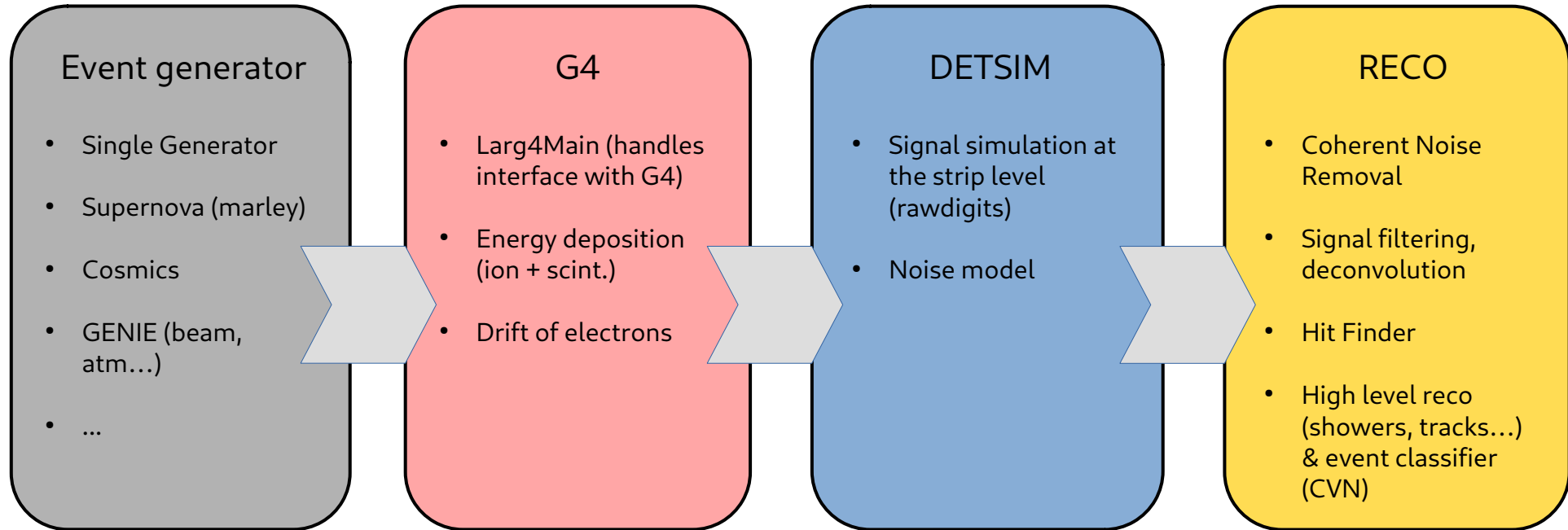


Motivations

- Up to now the vertical drift simulation workflow is done by artificially rotating the VD detectors by 90° to have the drift axis along x.
- Not a major issue for whoever is used to this trick, but an issue for newcomers and larsoft beginners (think of event displays and event generators). There exists much time spoiled at going through this trick for them, thus there is room for saving time for future newcomers.
- Natural that LArSoft evolves since dual phase and then vertical drift LArTPCs made the technology evolve. LArSoft should follow !
- It's possible that the task isn't that hard to overcome.
- This talk will get a bit technical from time to time, sorry about that. Details are given to facilitate further efforts, I'll try to keep the global picture in mind during the talk.



I- Simulation workflow



- My goal was to have something working nice up to detsim and then get in touch with pandora people to discuss current limitations. However things are moving faster than I thought.

What is targetted

- ProtoDUNE-VD : compare sim workflow of driftX and driftY and make sure results are the same. Easy for detsim (check rawdigits), harder for reco.

Work to be done


- **GDML** file with y-axis as the drift axis. ----- **T. Kosc**
- **Generators** ? There should be no problem. Problem appeared when we rotated the detector by 90° , so in principle they should go away once we rotate it back. . ----- **T. Kosc**
- **G4** step should already work fine. ----- /
- **DETSIM** (Nitish and Wenqiang to implement driftY in wirecell). ----- **Nitish & Wenqiang**
- **RECO** : more tricky. Pandora is a multi-algorith software, implies a multi source of possible bugs (cosmic reco, beam reco, neutrino reco...). Need for a robust testing campaign. ----- **???**

Few things from the past

I- Some previous work (1)

- **Christophe Alt** worked on drift direction in **larsim/ElectronDrift/SimDriftElectrons_module.cc** : the module can handle drift direction in $\pm x$, $\pm y$ and $\pm z$.

```
auto const TPCcenter = GetCenter();  
auto const PlaneCenter = Plane(0).GetBoxCenter(); // any will do  
auto const driftVector = PlaneCenter - TPCcenter; // approximation!
```



```
int driftcoordinate = std::abs(tpcGeo.DetectDriftDirection()) -  
  
int transversecoordinate1 = 0;  
int transversecoordinate2 = 0;  
if (driftcoordinate == 0) {  
    transversecoordinate1 = 1;  
    transversecoordinate2 = 2;  
}  
else if (driftcoordinate == 1) {  
    transversecoordinate1 = 0;  
    transversecoordinate2 = 2;  
}  
else if (driftcoordinate == 2) {  
    transversecoordinate1 = 0;  
    transversecoordinate2 = 1;  
}
```

- There exists a method in **geo::TPCGeo::DetectDriftDirection** which detects the drift direction by comparing the positions of the center of the TPC and the center of the first plane. So in principle the G4 step is already fine (event display check to make sure energy is deposited as expected).

I- Some previous work (2)

- Etienne Chardonnet worked on protoDUNE Dual Phase (which used a vertical drift) reconstruction by adding a parameter into Pandora xml configuration file.
- I'm not sure how deep in Pandora this parameter propagated, it's likely that it only reached the cosmic reconstruction.

« IsDualPhase » as new xml parameter

```
<!-- VertexAlgorithms -->  
<algorithm type = "LArCosmicRayVertexBuilding">  
  <InputPfoListName>MuonParticles3D</InputPfoListName>  
  <OutputVertexListName>CRVertices3D</OutputVertexListName>  
  <IsDualPhase>true</IsDualPhase>  
</algorithm>
```


III- What I've done so far (1) : gdml file of coldbox CRP1

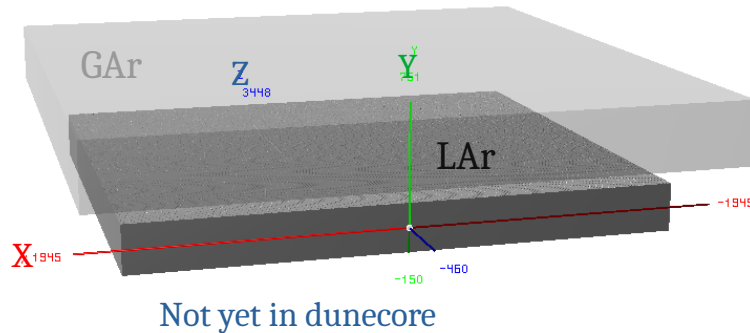
- Gdml files are generated using perl scripts. ~ 1000 lines for a 'simple' geometry as the one of the coldbox, ~ 2000 lines for far detector VD.
- Use following commands (in root session) to see progress :
`gSystem->Load("libGeom");`
`TgeoManager::Import("your_gdml_file.gdml");`
`gGeoManager->GetTopVolume()->Draw("ogl");`
- Most of the job consists in changing variables to make y-axis become the drift direction.
- Most tricky part is to position wires correctly.

```
dunecore/Geometry/gdml/generate_dunevdcbl_v2_refactored.pl
585  sub gen_TPC()
586  {
587      # CRM active volume
588      my $TPCActive_x = $driftTPCActive;
589      my $TPCActive_y = $widthCRM_active;
590      my $TPCActive_z = $lengthCRM_active;
591
592      # CRM total volume
593      my $TPC_x = $TPCActive_x + $ReadoutPlane;
594      my $TPC_y = $widthCRM;
595      my $TPC_z = $lengthCRM;
```

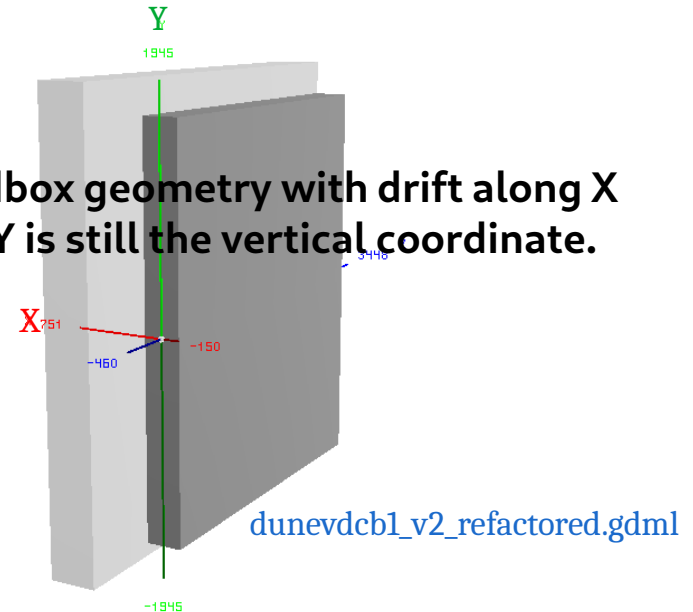
III- What I've done so far (2) : gdml file

- Targetted detector should be far detector 10 kTon (at least module 0...). For convenient reasons I decided to work on coldbox CRP1 (easier, faster at running simulations...).
- At some point I made it to have a good looking geometry file. I only have the gdml file locally, didn't push it, but I plan to do it.

Coldbox geometry with drift along Y

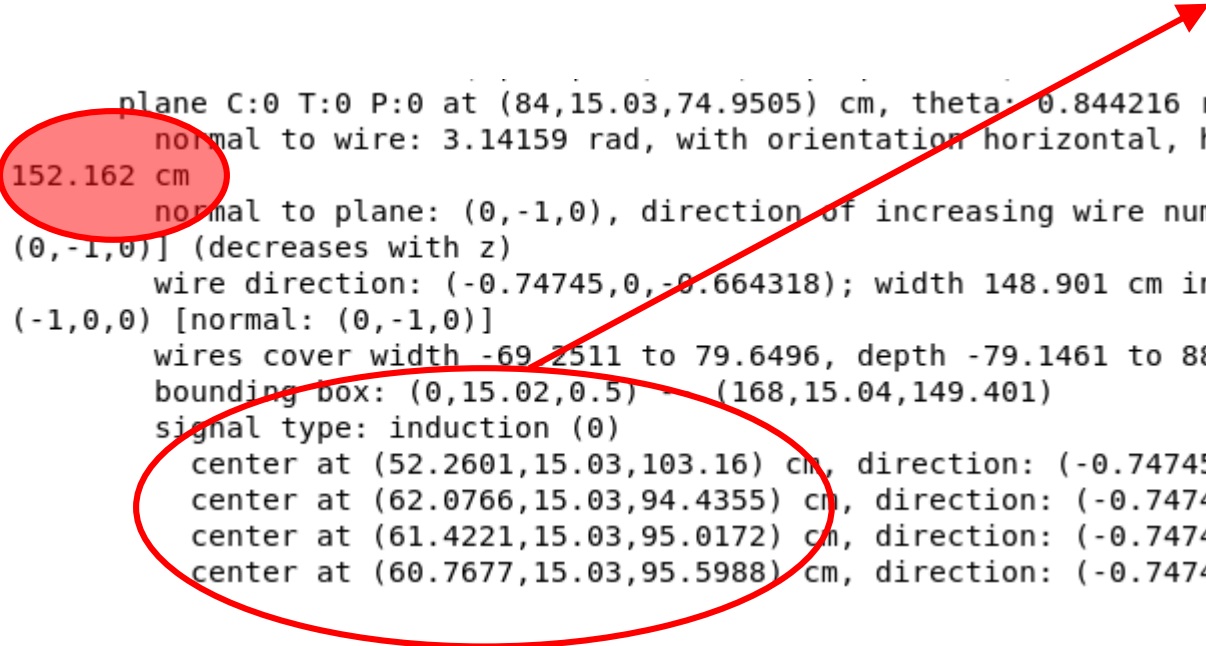


**Coldbox geometry with drift along X
but Y is still the vertical coordinate.**



III- What I've done so far (3) : from gdml to LArSoft

- One needs to make sure the **gdml** is handled correctly in LArSoft !
- Used `geo::GeometryCore::Print()` to check geometry with max. verbose. Found out that **wires were shuffled**, suspected an incorrect sorting due to hardcoded sorting coordinates. HD \rightarrow (y;z) but VD \rightarrow (x;z) .



```
plane C:0 T:0 P:0 at (84,15.03,74.9505) cm, theta: 0.844216 rad
normal to wire: 3.14159 rad, with orientation horizontal, has 256 wires measuring V with a wire pitch of
152.162 cm
normal to plane: (0,-1,0), direction of increasing wire number: (0.664318,0,-0.74745) [wire frame normal:
(0,-1,0)] (decreases with z)
wire direction: (-0.74745,0,-0.664318); width 148.901 cm in direction: (0,0,1), depth 168 cm in direction:
(-1,0,0) [normal: (0,-1,0)]
wires cover width -69.2511 to 79.6496, depth -79.1461 to 88.388 cm
bounding box: (0,15.02,0.5) - (168,15.04,149.401)
signal type: induction (0)
center at (52.2601,15.03,103.16) cm, direction: (-0.74745,0,-0.664318)
center at (62.0766,15.03,94.4355) cm, direction: (-0.74745,0,-0.664318)
center at (61.4221,15.03,95.0172) cm, direction: (-0.74745,0,-0.664318)
center at (60.7677,15.03,95.5988) cm, direction: (-0.74745,0,-0.664318)
```

III- What I've done so far (4) : from gdml to LArSoft

- The issue comes from the GeoObjectSorter class (dunecore/Geometry/GeoObjectSorterCRU.cxx) which assumes x as the drift coordinate to sort objects, in particular wires.
- The function that sorts wires uses coordinates (y;z), but obviously that will produce bugs if y is chosen to be the drift coordinate since wires will be in the (x;z) plane.

```
//-----  
bool sortWire(WireGeo const& w1, WireGeo const& w2){  
    // wire sorting algorithm  
    // z_low -> z_high  
    // for same-z group  
    // the sorting either from bottom (y) left (z) corner up for strip angles > pi/2  
    // and horizontal wires  
    // or from top corner to bottom otherwise  
    // we assume all wires in the plane are parallel
```

- As a temporary solution (thank you Slavic !) I wrote a new channel map + geo object sorter object in which y-axis is hardcoded as the drift coordinate.
- Slavic made dunecore/Geometry/GeoObjectSorter60D.cxx (coldbox crp2 + protoDUNE VD eventually) able to detect drift direction.

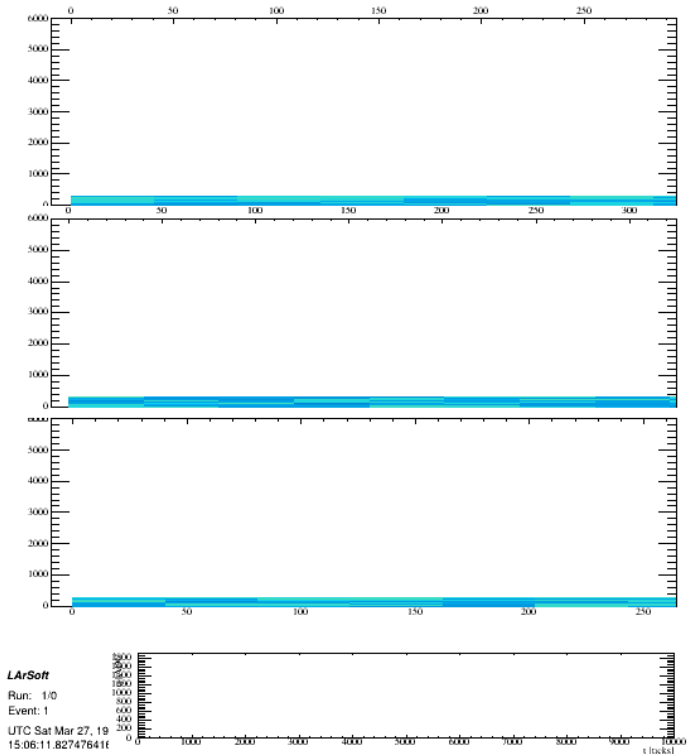
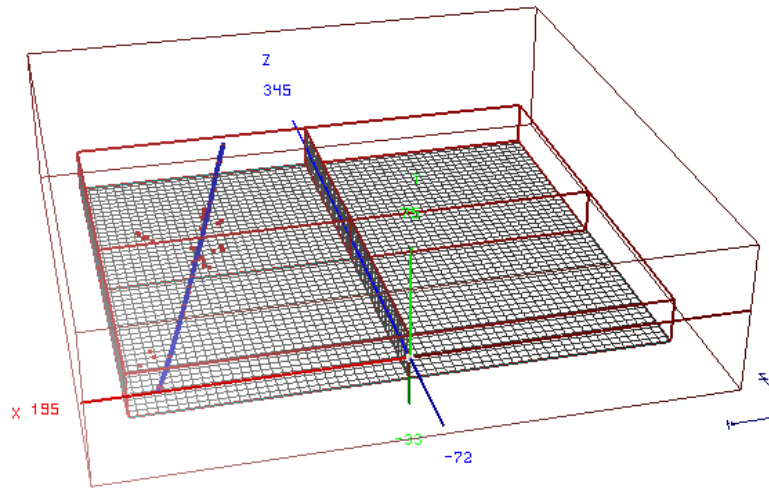
III- What I've done so far (5) : from gdml to LArSoft

- Finally the new channel map algorithm must be called in `dunecore/geometry/DUNEGeometryHelper_service.cc`

```
/*  
    // VD CRP cold box channel map  
    } else if ( ( detectorName.find("dunevdcbox") != std::string::npos ) ) {  
        channelMap = std::make_unique<geo::ColdBoxChannelMapAlg>(pset);  
        //channelMap = std::make_unique<geo::ChannelMapCRUAlg>(pset);  
*/  
  
// VD CRP cold box channel map  
} else if ( ( detectorName.find("dunevdcbox") != std::string::npos ) ) {  
    channelMap = std::make_unique<geo::MyColdBoxChannelMapAlg>(pset);  
    //channelMap = std::make_unique<geo::ChannelMapCRUAlg>(pset);
```

III- What I've done so far (6) : sim workflow status ?

- 2 GeV muon in coldbox CRP1 top electronics :
energy deposited works fine, but out of the box
detector response not yet working.



Change suggestions in LArSoft

IV- What could be done in LArSoft

- dunecore/Geometry/GeoObjectSorterCRU.h (derived from LArSoft class GeoObjectSorter). Add drift direction as second argument to method SortWires + necessary logic to handle it.

```
namespace geo{

class GeoObjectSorterCRU : public GeoObjectSorter {
public:

    GeoObjectSorterCRU(fhicl::ParameterSet const& p);

    void SortAuxDets      (std::vector<geo::AuxDetGeo>      & adgeo) const;
    void SortAuxDetSensitive(std::vector<geo::AuxDetSensitiveGeo>& adsgo) const;
    void SortCryostats    (std::vector<geo::CryostatGeo>    & cgeo) const;
    void SortTPCs         (std::vector<geo::TPCGeo>         & tgeo) const;
    void SortPlanes      (std::vector<geo::PlaneGeo>      & pgeo,
                          geo::DriftDirection_t          driftDir) const;

    void SortWires        (std::vector<geo::WireGeo>        & wgeo) const;
    void SortOpDets       (std::vector<geo::OpDetGeo> & opdet) const;

};
```

➤ larcoreobj/SimpleTypesAndConstants/

```
* @brief Drift direction: positive or negative
*
* Do not use this type to distinguish different drift axes: e.g., negative
* x drift and negative z drift are both by `kNeg`.
*/
typedef enum driftdir {
    kUnknownDrift, ///< Drift direction is unknown.
    kPos,          ///< Drift towards positive values.
    kNeg,          ///< Drift towards negative values.
    kPosX = kPos,  ///< Drift towards positive X values.
    kNegX = kNeg    ///< Drift towards negative X values.
} DriftDirection_t;
```

Add kPosY, kNegY (kPosZ and kNegZ ?)
as possible values of type driftdir

IV- What has be done in LArSoft

- dunecore/Geometry/GeoObjectSorterCRU60D.cxx : Slavic called method `geo::TPCGeo.DetectoDriftDirection()` in SortTPCs and stored value in global variable.

```
//-----  
void GeoObjectSorterCRU60D::SortTPCs(std::vector<geo::TPCGeo> & tgeo) const  
{  
    // attempt to get TPC drift direction: 1 - X, 2 - Y, 3 - Z  
    if( !geo::CRU60D::TPCDriftAxis ){  
        geo::CRU60D::TPCDriftAxis = std::abs((*tgeo.begin()).DetectDriftDirection());  
        MF_LOG_DEBUG("GeoObjectSorterCRU60D")  
        <<" Retrieved drift axis "<<geo::CRU60D::TPCDriftAxis;  
    }  
    std::sort(tgeo.begin(), tgeo.end(), CRU60D::sortTPC);  
}
```

Discussion

- Current efforts ongoing to implement correct handling of vertical drift geometry in LArSoft.
- Targetting protoDUNE-VD. Testing campaign at the simulation level still under discussion, to make sure transition from driftX to driftY does not bring bugs, especially at the reco stage.
- Wenqiang has created a dedicated slack channel to coordinate efforts.
- Any help of course appreciated !