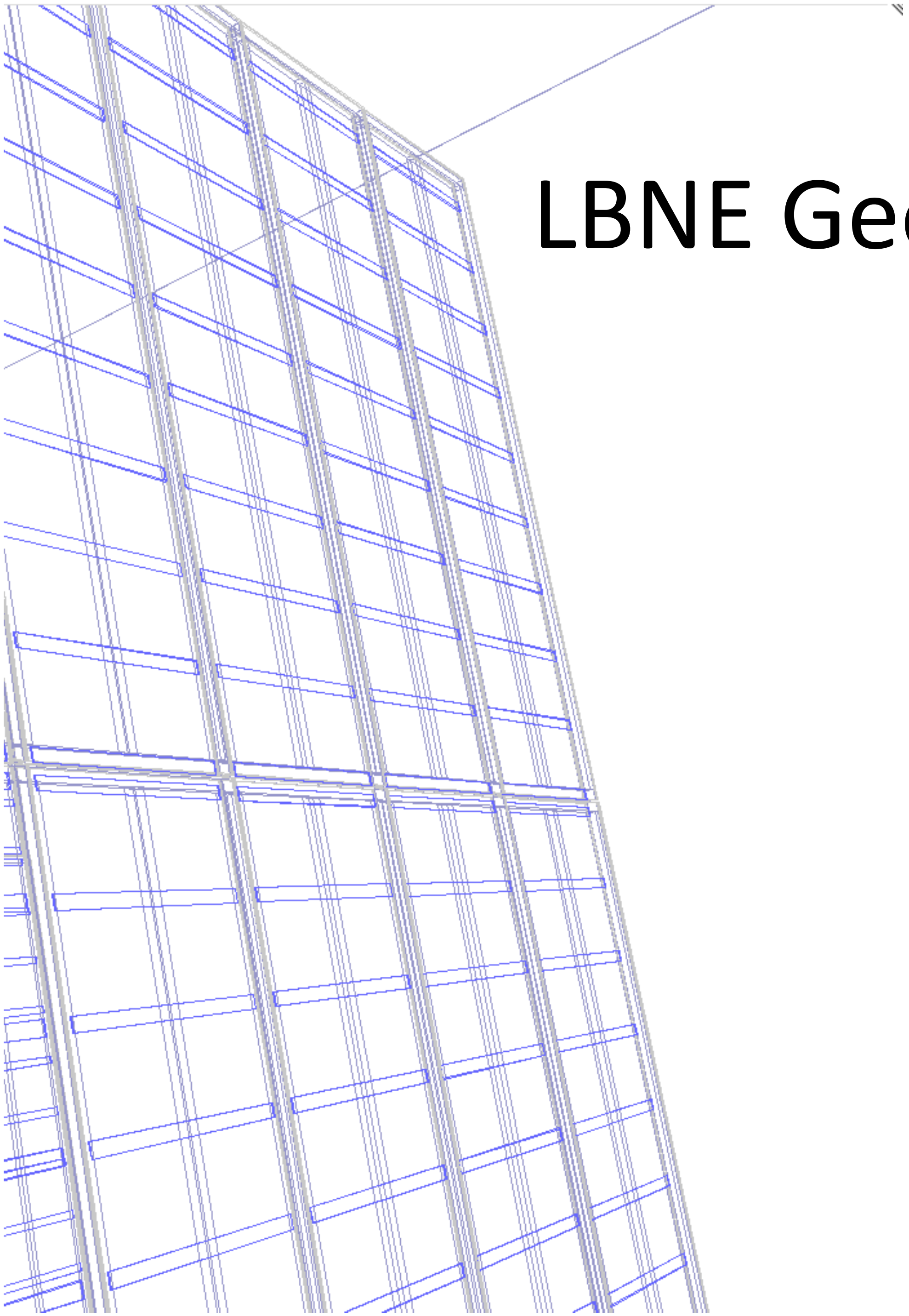


LBNE Geometry in LArSoft
Update

Tyler Alion
Jae Kim
Brian Rebel
Mike Kirby
Tom Junk

2/13/13

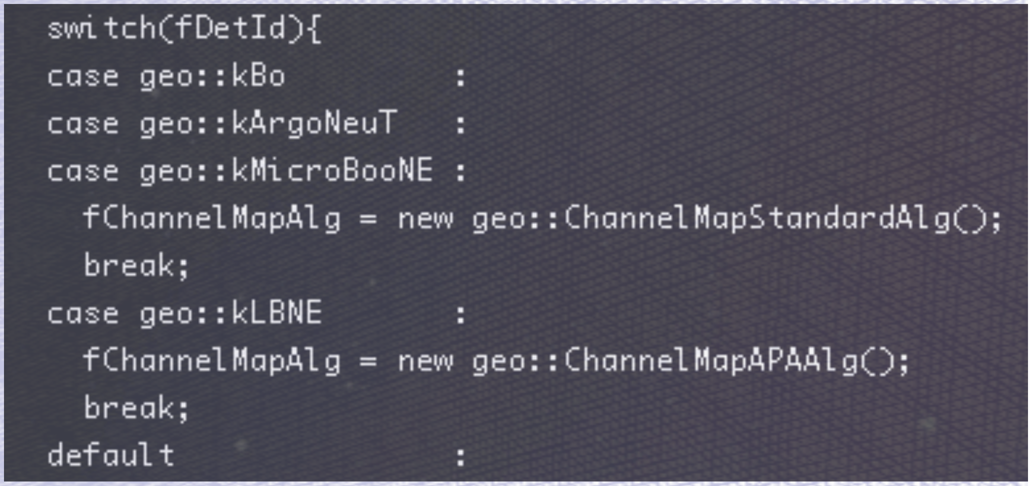
Outline

- Developments
  - Wire Wrapping is functional and public!
  - Volume object sorting
  - Plane/View_t Conventions
  - Geometry Testing
- GDML development
  - APA Frames added (around paddles)
  - Readjusting volTPC dimensions

geo::ChannelMapAlg

*Certain methods cannot be written in a unified way such that the method works the same for both configurations.

- ChannelMapAlg contains: Initialize – Uninitialize – ChannelToWire – PlaneWireToChannel – NearestWire.
- Geometry constructor calls LoadGeometry
  - Loads proper gdml file,
  - Calls InitializeChannelMap,
  - fDetId chooses mapping.



```
switch(fDetId){
case geo::kBo          :
case geo::kArgoNeuT    :
case geo::kMicroBooNE  :
  fChannelMapAlg = new geo::ChannelMapStandardAlg();
  break;
case geo::kLBNE        :
  fChannelMapAlg = new geo::ChannelMapAPAAlg();
  break;
default                :
```

*APAAlg defines the above methods with code re-written to know about LBNE geometry differences

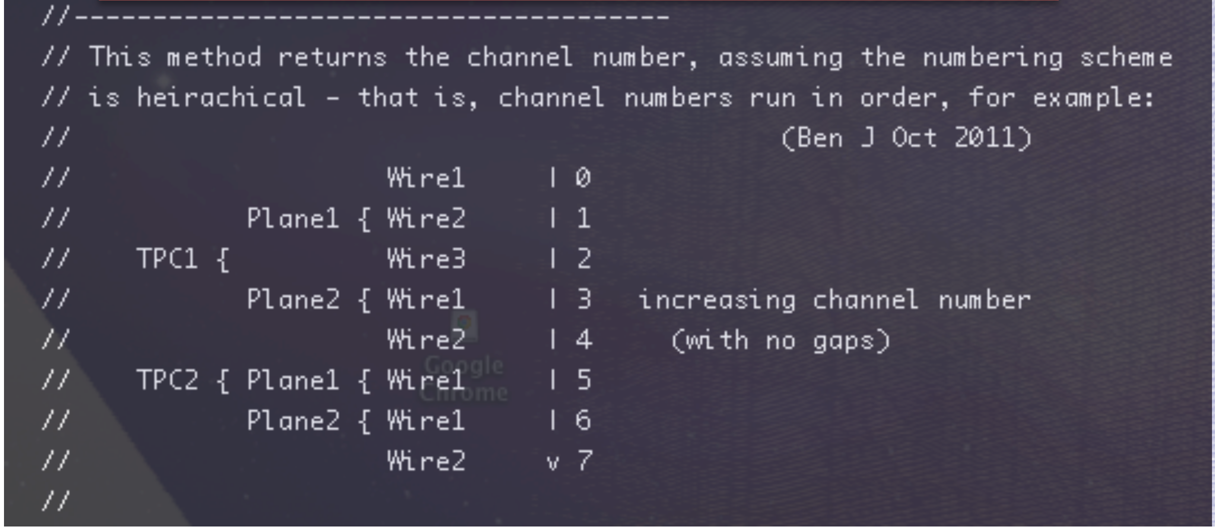
*StandardAlg contains the existing methods for the rest of the detector IDs

geo::ChannelMapAPAAlg

- LBNE readout channels live in TWO TPCs, where StandardAlg assumes *only* one TPC **
  - different Initialize()
  - ChannelToWire and PlaneWireToChannel build off of this

- Nchannels now returns an appropriate private data member
  - Other useful data members, though retrieval is limited by dependence on the existence of a parallel data member in StandardAlg

** StandardAlg PlaneWireToChannel comment:

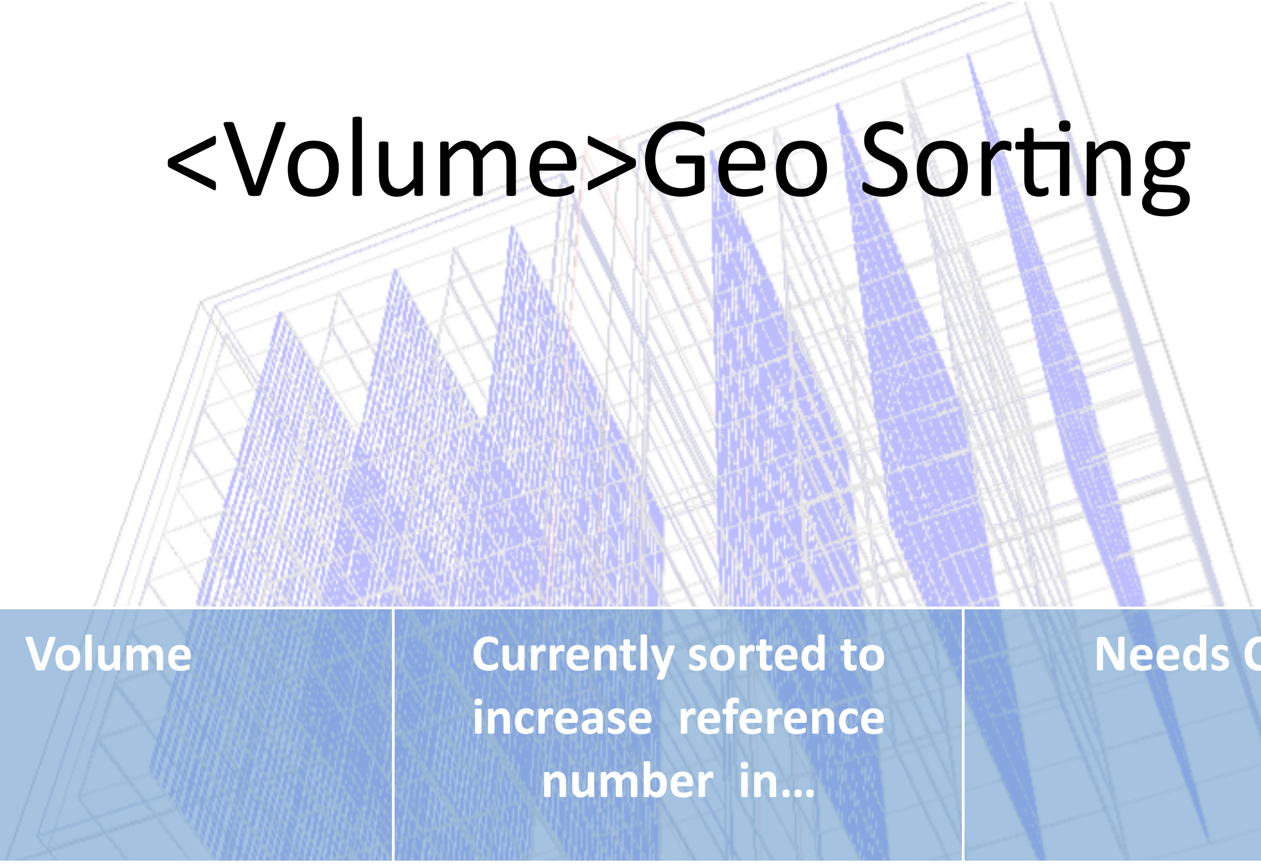


```
//-------------------------------------
// This method returns the channel number, assuming the numbering scheme
// is heirachical - that is, channel numbers run in order, for example:
//                                             (Ben J Oct 2011)
//                   Wire1     | 0
//          Plane1 { Wire2     | 1
//    TPC1 {         Wire3     | 2
//          Plane2 { Wire1     | 3   increasing channel number
//                   Wire2     | 4     (with no gaps)
//    TPC2 { Plane1 { Wire1    | 5
//          Plane2 { Wire1     | 6
//                   Wire2     v 7
//
```

- Uninitialize() clears memory usage of these private data members

- NearestWire (called by NearestChannel) still needs to be written for LBNE

<Volume>Geo Sorting

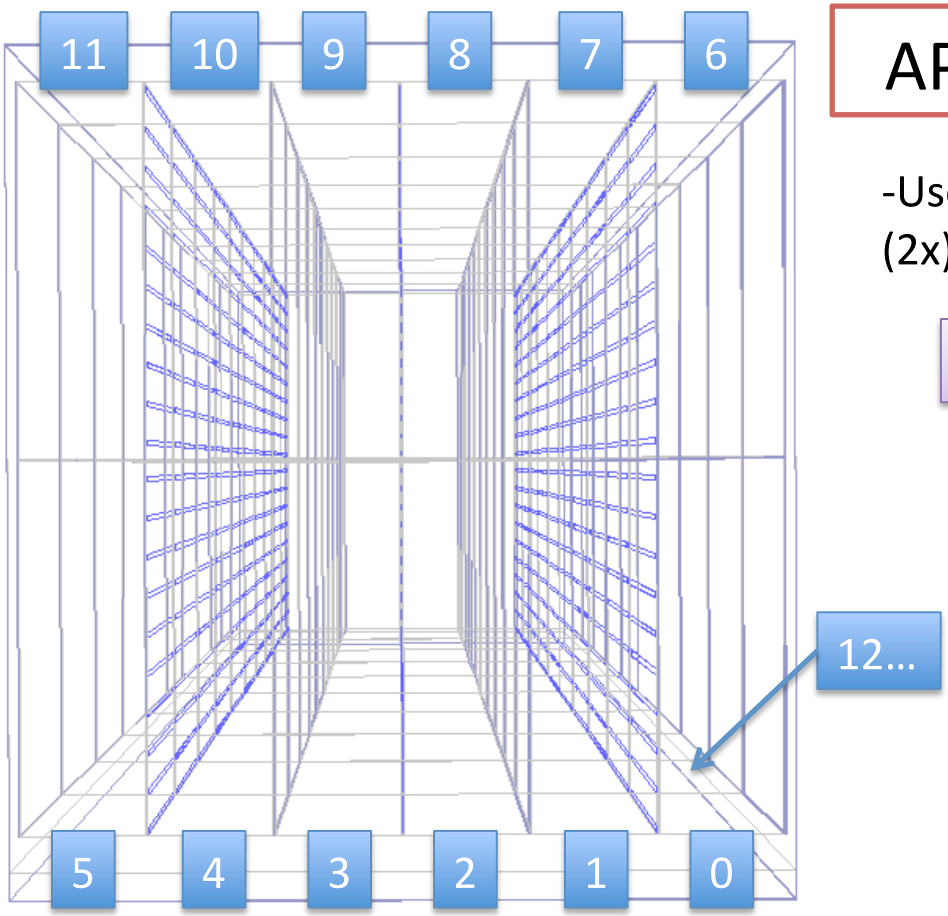


Volume	Currently sorted to increase reference number in…	Needs Change
Cryostats within World	+X	No
TPCs within a Cryostat	+X	Yes – in tpc_sort
Planes within a TPC	-X	Yes – in implementation of plane_sort
Wires within a Plane	+Z	Yes – in both method and implementation of sortByZPos

TPCs in CryostatGeo

New tpc_sort returns

```
if(xyz1[2] < xyz2[2]) return true;
if(xyz1[2] == xyz2[2] && xyz1[1] < xyz2[1]) return true;
if(xyz1[2] == xyz2[2] && xyz1[1] == xyz2[1] && xyz1[0] < xyz2[0]) return true;
return false;
```



APA x == TPCs 2x and 2x+1

-Useful to loop through APAs (x) and use (2x) or (2x+1) to refer to whichever TPC

Under this sorting convention:

-tpc%2 = 0 is a tpc number that references a TPC with drift direction kPosX

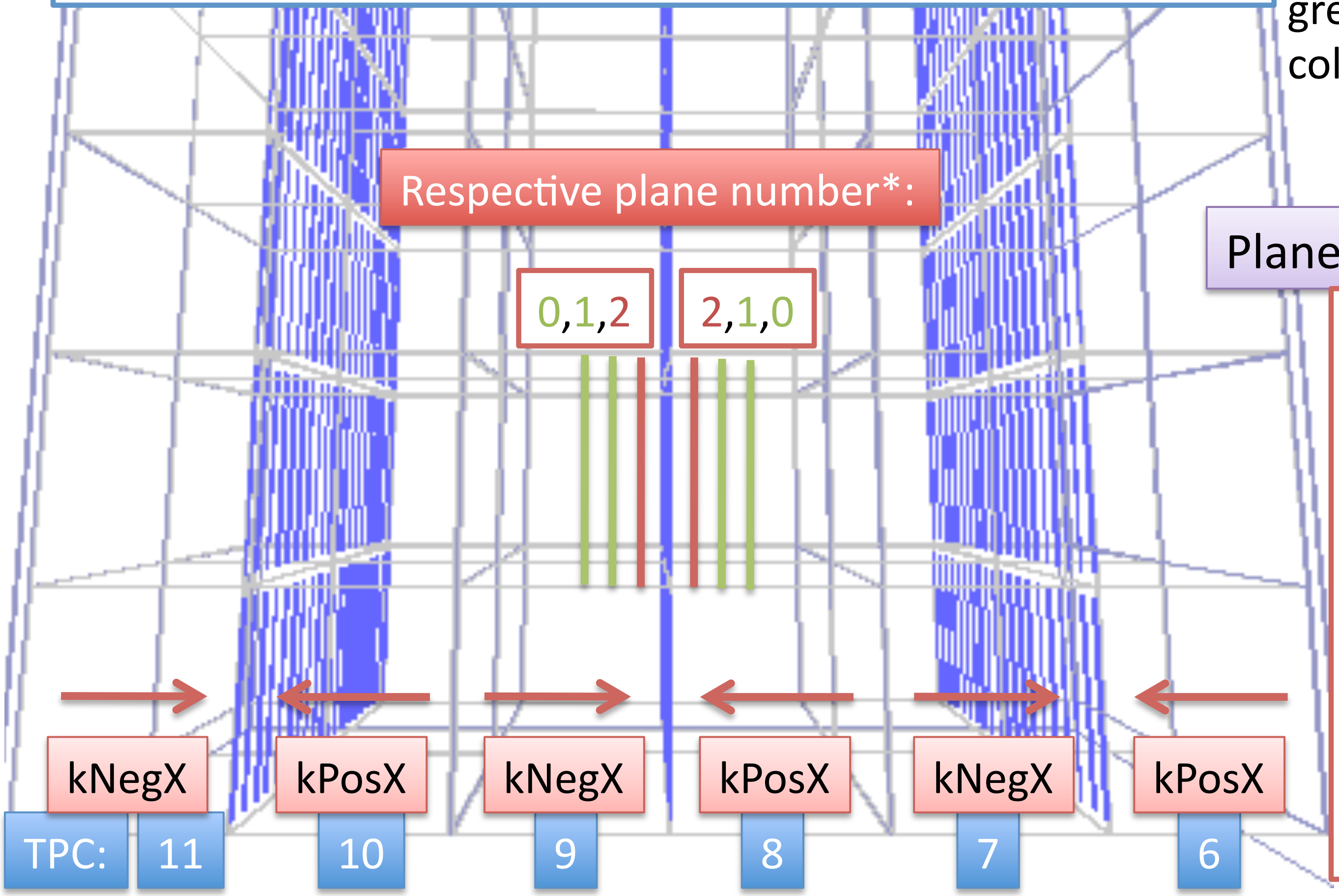
-tpc%2 = 1 references a TPC with drift direction kNegX (the rotated volTPC in GDML)

Planes in TPCGeo

Same plane_sort, new implementation:

```
if (fDriftDirection == geo::kPosX)
        { std::sort(fPlanes.rbegin(), fPlanes.rend(), plane_sort); }
else if (fDriftDirection == geo::kNegX)
        { std::sort(fPlanes.begin(), fPlanes.end(), plane_sort); }
```

Plane number increases in drift direction



Respective plane number*:

0,1,2

2,1,0

kNegX	kPosX	kNegX	kPosX	kNegX	kPosX
TPC: 11	10	9	8	7	6

*plane spacing is exaggerated, green is induction and red is collection.

Plane#/View_t convention:

-Plane 0 and 1 are assigned view kU ad kV; induction

-Plane 2 is assigned view kZ; collection

-Proposing to change the view kW to kZ since, up to some scale, it measures the z direction

Wires in PlaneGeo

New sortByZPos:

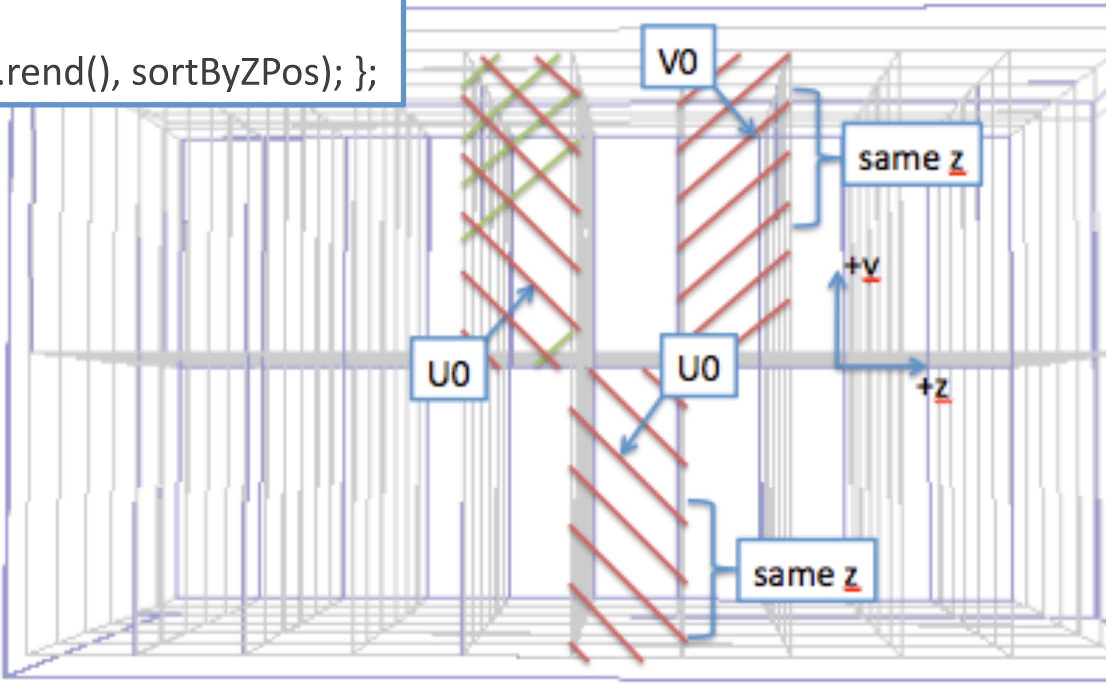
```
if( xyz1[2] < xyz2[2] ) return true;
if( xyz1[2] == xyz2[2] && w1->ThetaZ() < 0.5*TMath::Pi() && xyz1[1] > xyz2[1] )
        return true;
if( xyz1[2] == xyz2[2] && w1->ThetaZ() > 0.5*TMath::Pi() && xyz1[1] < xyz2[1])
        return true;
```

In LBNE geometry, many wires per plane have the same Z coordinate. Depending on plane, sort those in +/-Y

New implementation:

```
this->LocalToWorld(origin, planeworld);
if ( planeworld[1] >= 0 )
        { std::sort(fWire.begin(), fWire.end(), sortByZPos); }
else if ( planeworld[1] < 0 )
        { std::sort(fWire.rbegin(), fWire.rend(), sortByZPos); };
```

In LBNE geometry, bottom TPCs need to be rotated 180 around X, to appropriately represent channel readouts placement. TPCs are ambiguous to this rotation until wires are sorted, so sort the opposie way in this case.



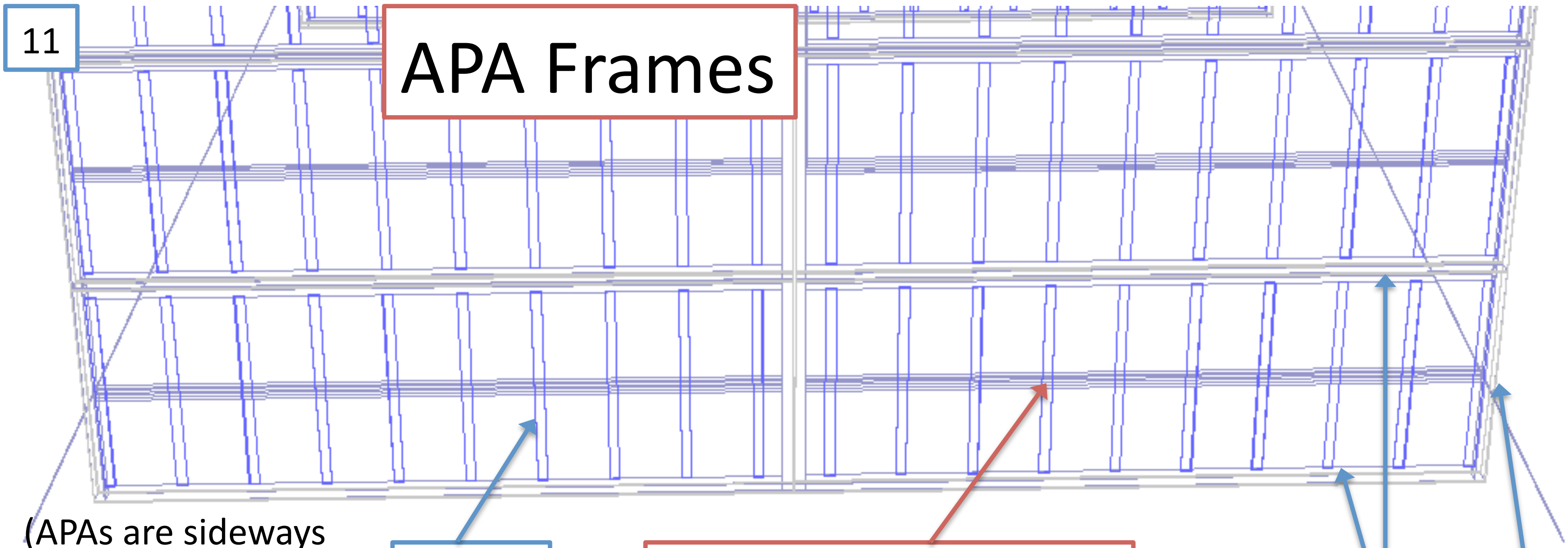
Geometry Testing

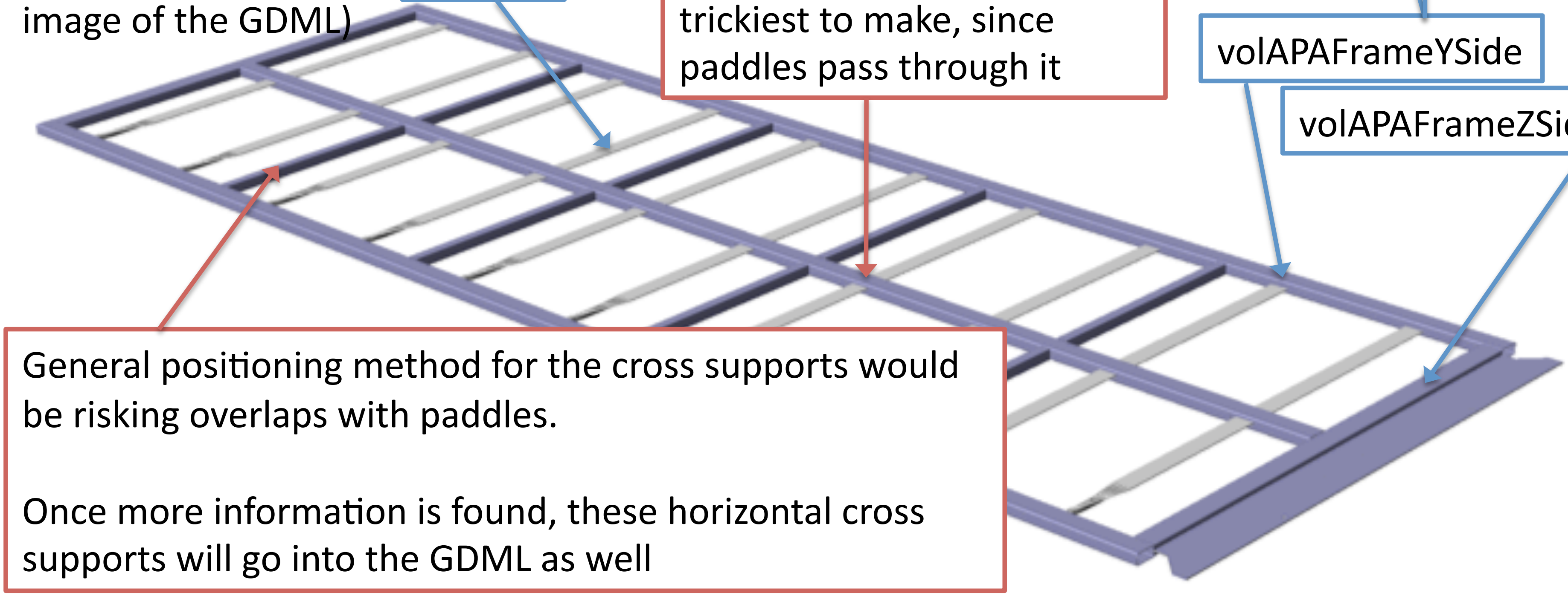
- All geometries pass all tests
  - New sorting and plane/view convention does not change MicroBooNE/ArgoNeuT/Bo functioinality
- Added overlap checker
  - Some overlaps for current geometries
- May write more APA-specific tests as needed
  - Correct drift directions

lbne10kT.gdml

- Under repair:
  - Hillside, Service building (redefine Arb8 solids)
- Under construction:
  - SiPMs (more discussion needed)
    - Paddles do not yet represent the twisted ganging, to feed into SiPMs
    - That would be messy, could we just leave the paddle perfectly rectangular and place the SiPMs *as if* the paddles *were* twisted into them? Then use a function in LArSoft to account for anything needing accounting for.
  - Field Cage
    - This has been made, it just needs to be generalized to span the whole Cryostat
- Overlaps
  - Many on the order of 10E-14, inexplicable.
  - Several other extrusions and overlaps which aren't so negligible – they are currently being fixed.

APA Frames





(APAs are sideways
In this generated
image of the GDML)

Paddles

APA Frame Y support is the trickiest to make, since paddles pass through it

volAPAFrameYSide

volAPAFrameZSide

General positioning method for the cross supports would be risking overlaps with paddles.

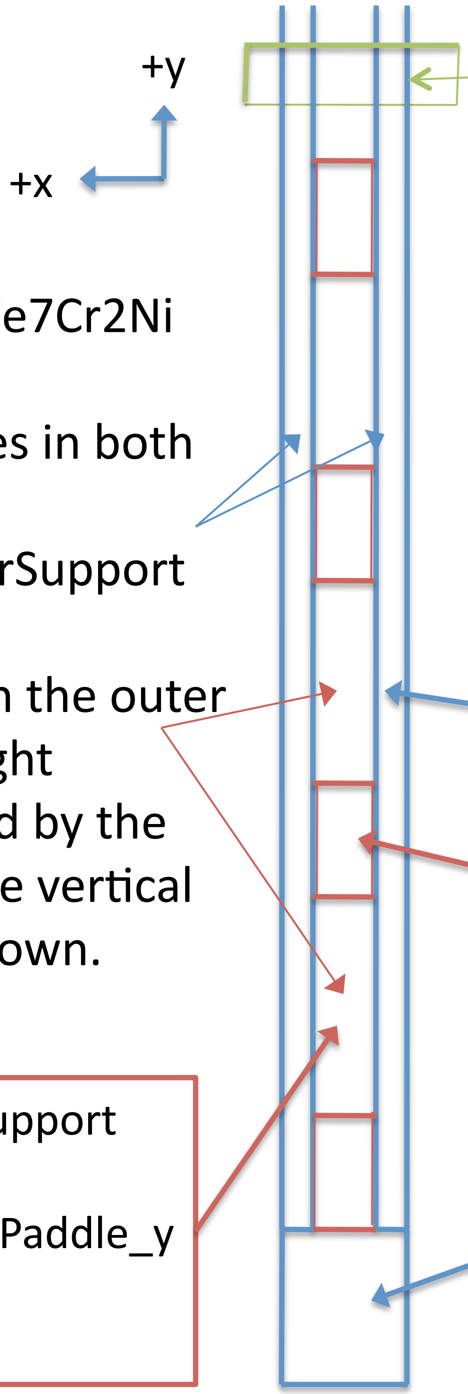
Once more information is found, these horizontal cross supports will go into the GDML as well

APA Frame

Material:
STEEL_STAINLESS_Fe7Cr2Ni

-Blocks Light paddles in both directions with volAPAFrameYOuterSupport

-The space between the outer supports and the light paddles will be filled by the inner supports, once vertical spacing is better known.



3 types of volumes make up the center vertical frame support.

In my experience, multiple subtractions from a single solid don't work well in general. So instead of making one box solid with holes, the perpendicular paddles are sandwiched between two long outer volumes, with a center volume filling the spaces between each paddle.

volAPAFrameYOuterSupport
x_dim = (2" – Paddle_x)/2
y_dim = 700cm – 2*4"
Z_dim = 2"

volLightPaddle
x_dim = 0.476
y_dim = 4"
Z_dim = 252 cm
same on top

volAPAFrameYInnerSupport
x_dim = 0.476 cm
y_dim = PaddleYInt – Paddle_y
Z_dim = 2"
same on top

APAFrameZSide
x_dim = 2"
y_dim = 4"
Z_dim = 252 cm
same on top

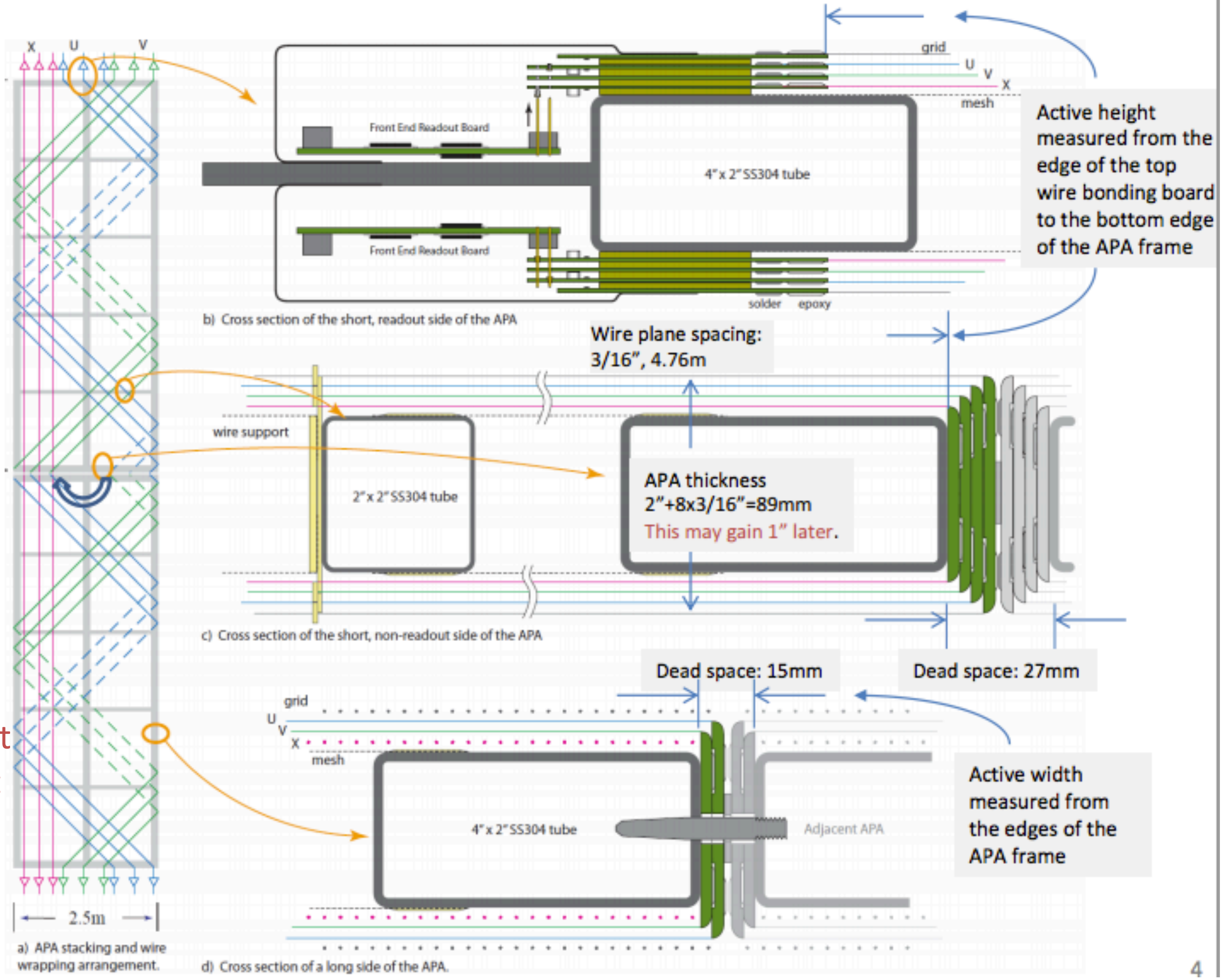
APA Dimensions

Slide by Bo Yu.

TPC dimensions are being changed to incorporate these dead spaces.

TPC volumes touch other TPCs in y and z directions, but touch their APA frames in x direction.

This is important so that the right volumes of the detector are active.



Active height measured from the edge of the top wire bonding board to the bottom edge of the APA frame

b) Cross section of the short, readout side of the APA

Wire plane spacing: 3/16", 4.76m

APA thickness 2"+8x3/16"=89mm

This may gain 1" later.

c) Cross section of the short, non-readout side of the APA

Dead space: 15mm

Dead space: 27mm

Active width measured from the edges of the APA frame

a) APA stacking and wire wrapping arrangement.

d) Cross section of a long side of the APA.

Discussion?