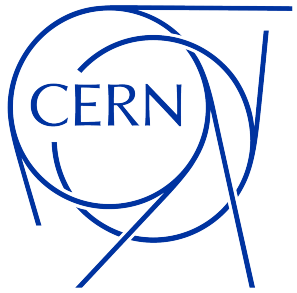
DD4hep implementation of the IDEA Endcap Calo with the capillary tubes technology

Lorenzo Pezzotti

Hidra Meeting
24/7/2024

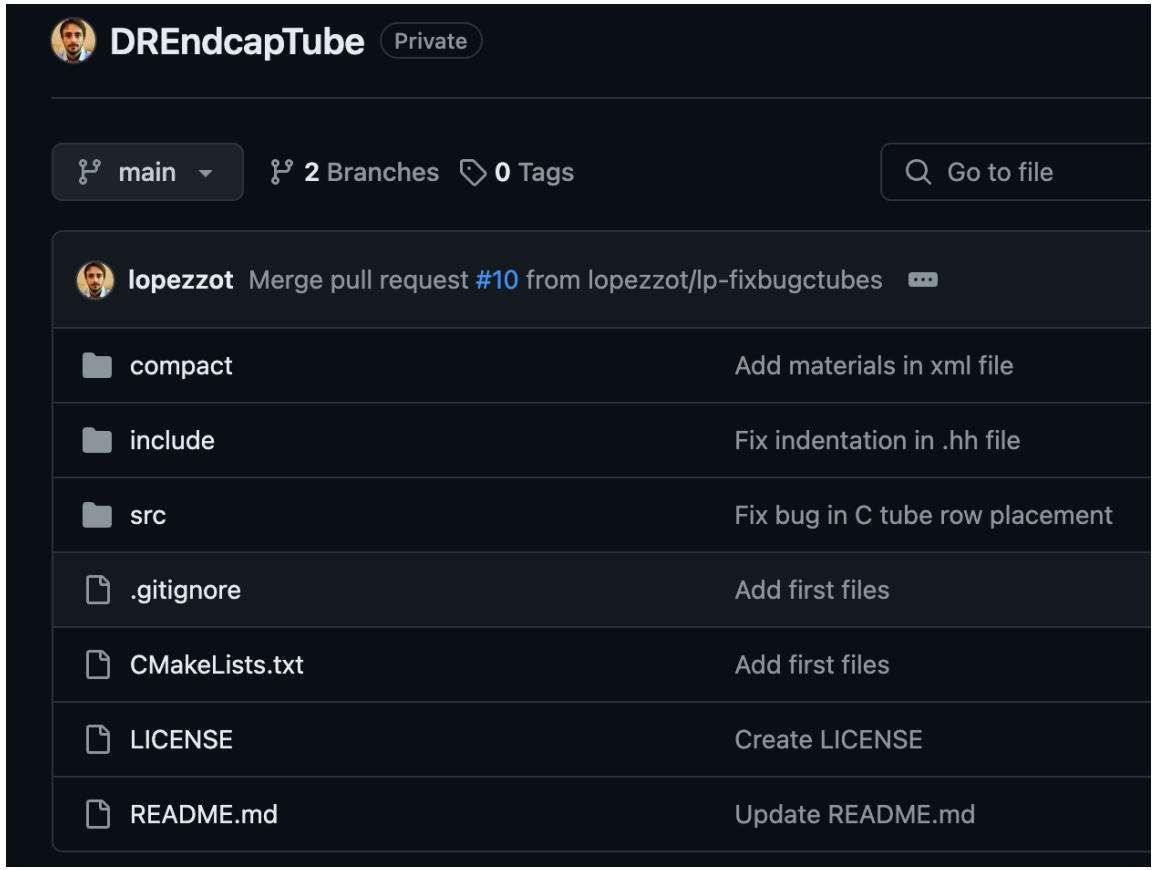


Activity purpose

- The goal of this activity is to develop a DD4hep representation of the IDEA dual-readout endcap calorimeter using the capillary tubes technology proposed by INFN
  - This geometry implementation is complementary to the calorimeter barrel one recently designed by Andreas Centeno [FCC-sim-presentation]
  - The code must be modular enough to also work with the IDEA crystal option, i.e. IDEA with a dual-readout crystal electromagnetic calorimeter before the solenoid
  - This geometry is intended to be ported to FCC sw as soon as possible

The code for the projective endcap

- DREndcapTubes
  - 700 C++ lines + 300 XML lines
  - A single translation unit, plus an header-only class with inline methods → easily portable to any sw
  - Currently in prerelease v0.1: geometry builds correctly but all the validation is yet to be done
  - On Mac M1 chip the DD4hep `create_detector` method builds the geometry
    - in ~20 s taking ~1.1 Gb
- Some history:
  - A couple of methods were taken from the 4th-concept dual-readout calorimeter simulation at the ILC
    - All the rest, especially the tube placement code, was reimplemented from scratch

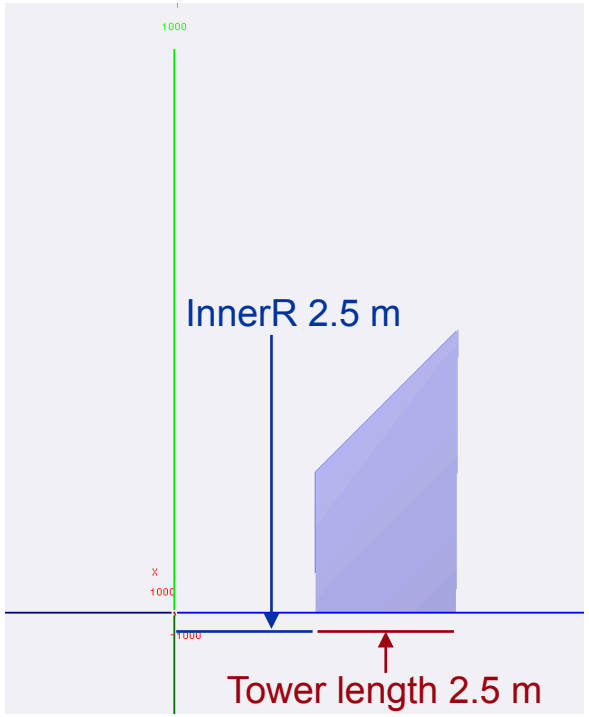


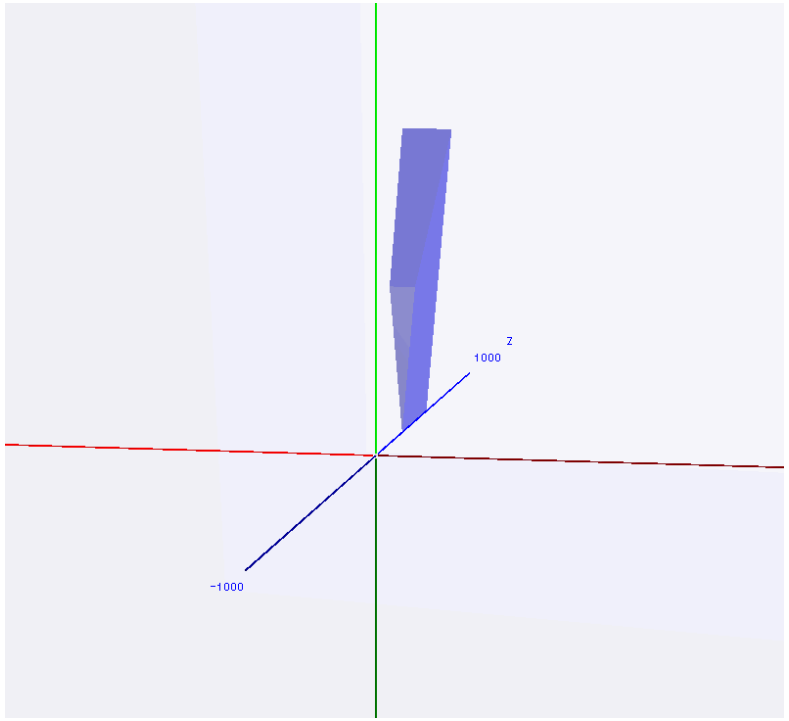
Geometry nesting

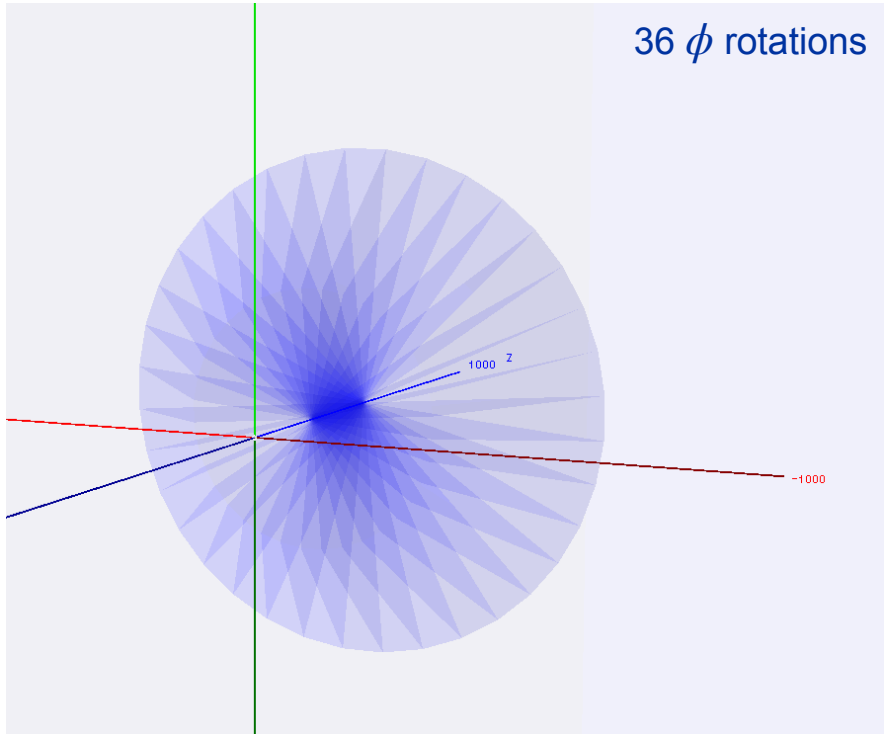
- The capillary tubes technology has no “tower” object, instead it glues together (many!) tubes to create a tower (trapezoid). However,
  - Placing in the simulation individual tubes directly in one endcap (or a ϕ-slice of it) is not the ideal solution as
  - navigating through many volumes is always a bad idea (even if we have voxelization),
  - also, having set of tubes housed in one “tower” will help the reconstruction, for instance for the calibration when each tower will be calibrated based on its signal and deposited energy from the MC truth
- Better to have “towers” of Air material and placing tubes inside each tower
  - Overall four levels of nesting are used

ϕ-slice

- A single ϕ-slice is a DD4hep EightPointSolid (equivalent to root Arb8, or Geant4 G4GenericTrap)
  - It is build according to the calorimeter inner radius (2.5 m), tower length (2.5 m), and ϕ-unit (2*π/36)
  - Assumes that the barrel and endcap region will touch at π/4
  - Is replicated around the z-axis to full coverage

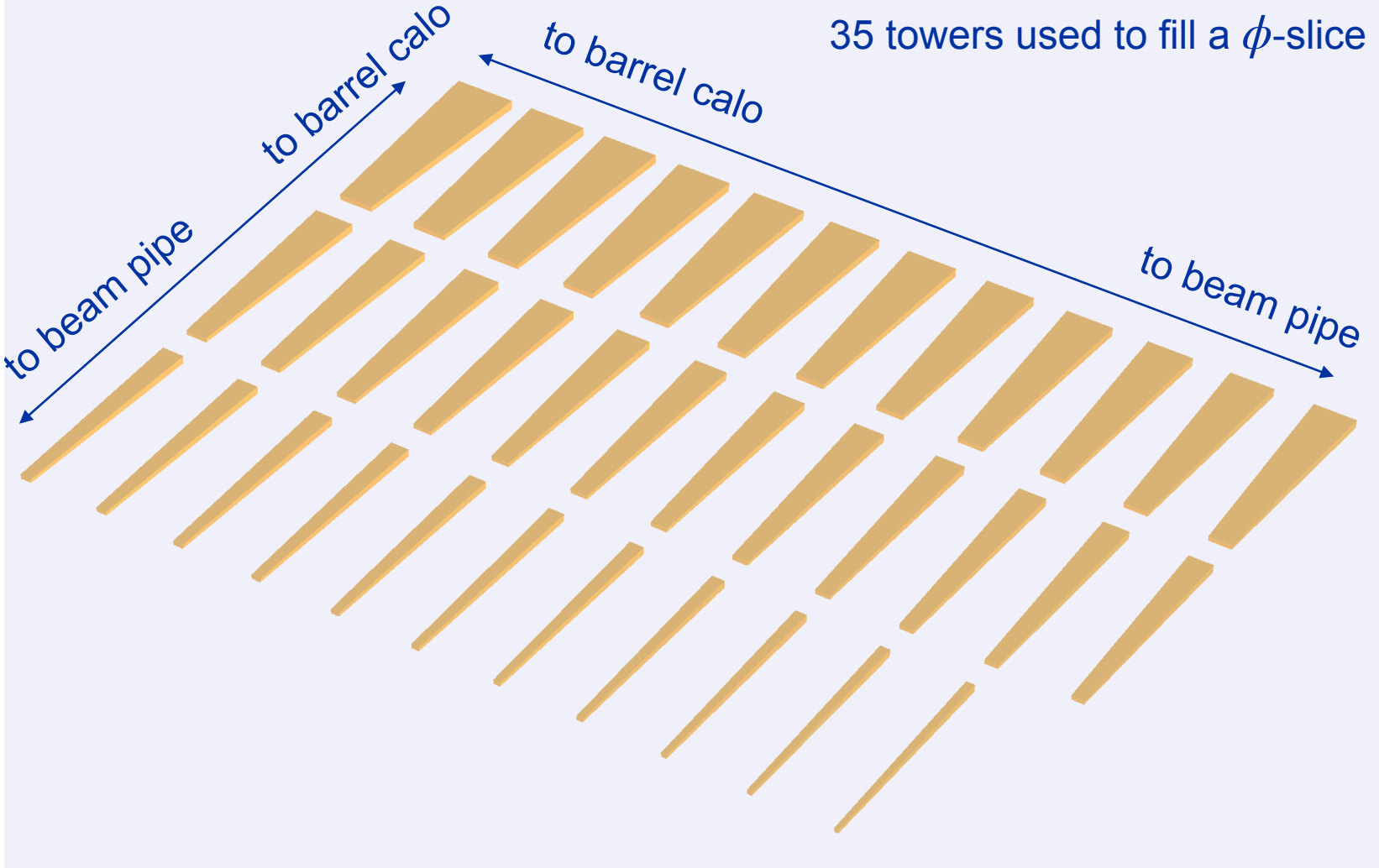






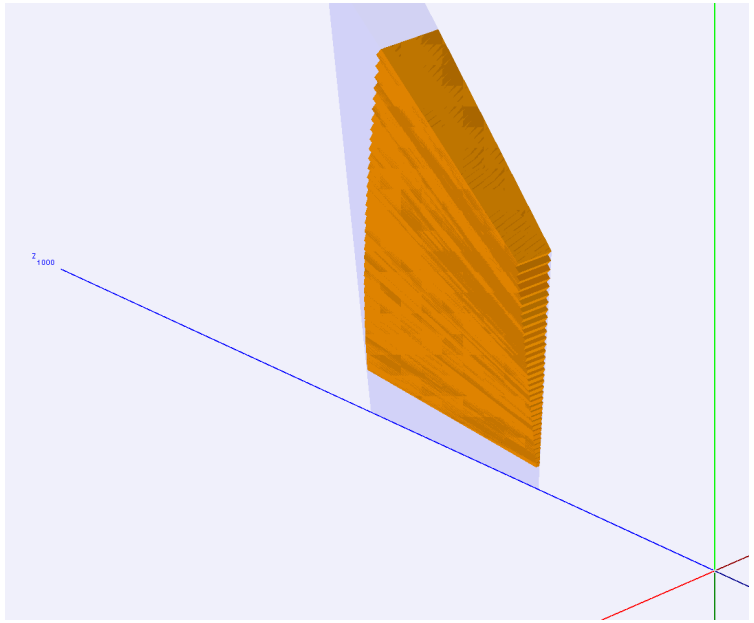
Towers

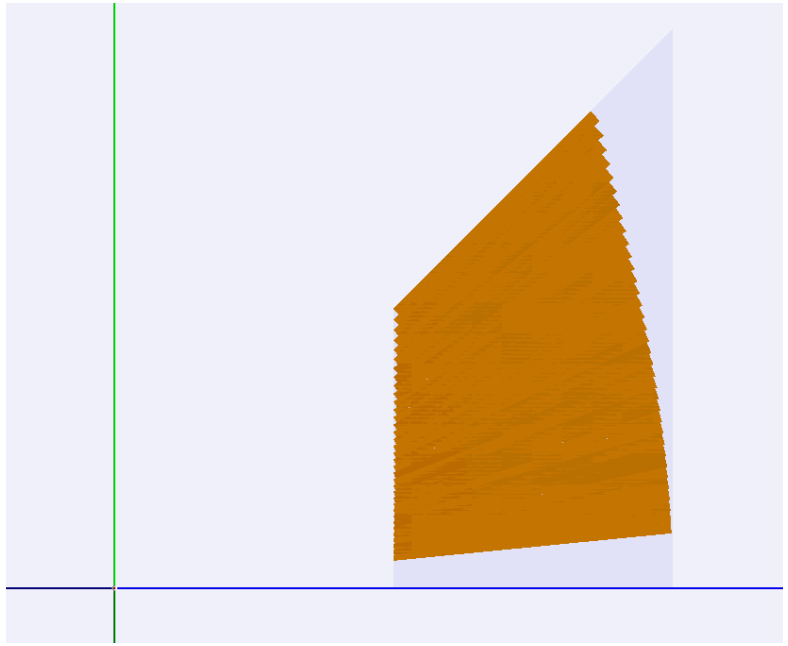
- Towers are Trap (equivalent to Geant4 G4Trap) that define a θ-region inside the ϕ-slice. The center of each tower points to the IP.
- 40 towers are considered, each covering a region of $\Delta\theta = \frac{\pi/4}{40}$
- As towers must fit inside the ϕ-slice their dimension changes with θ

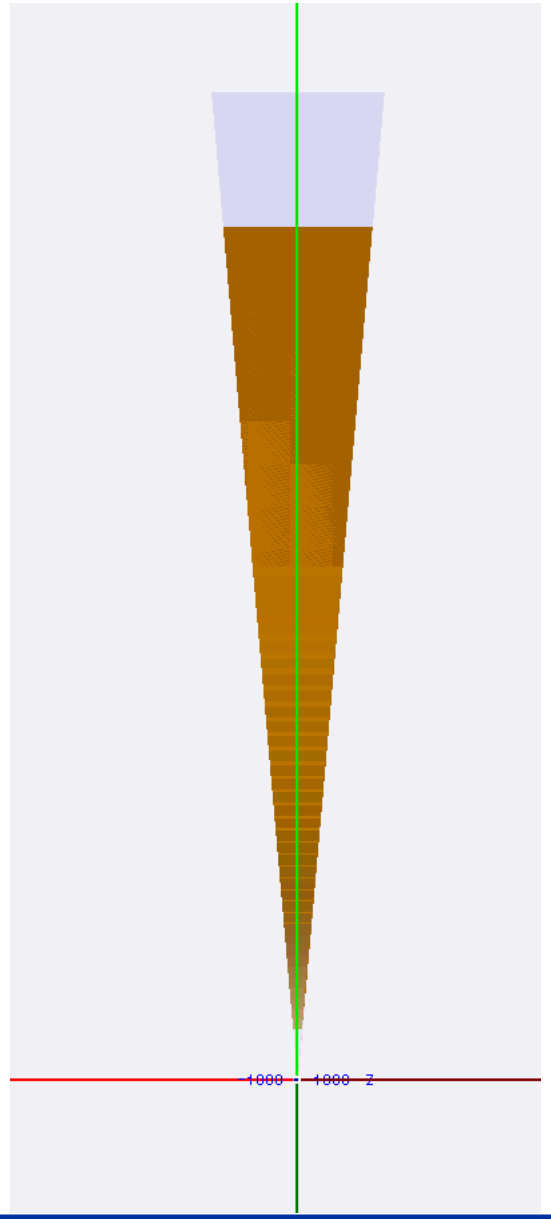


Towers

- Towers are Trap (equivalent to Geant4 G4Trap) that define a θ-region inside the ϕ-slice. The center of each tower points to the IP.
- To leave room to the beam pipe 35 towers are actually placed in a projective manner inside the ϕ-slice
- Each of the 35 is 2.5 m long, i.e. the calo containment is independent of θ







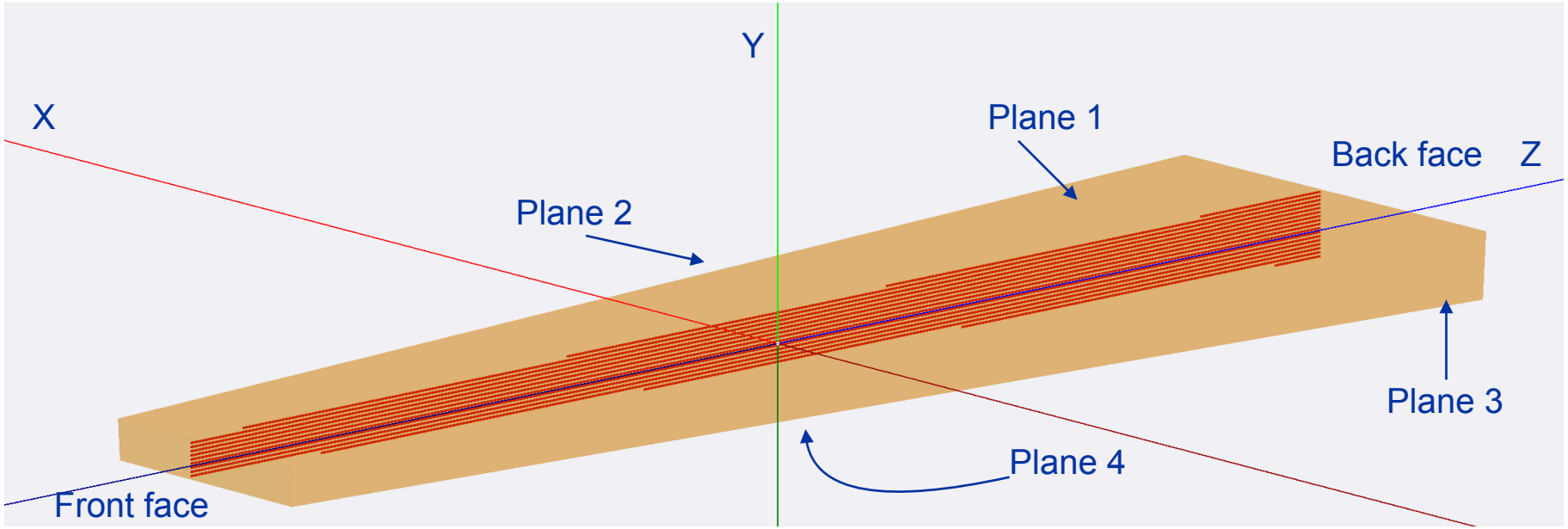
Tubes

- Tubes are 1-mm-thick radius Tubs (equivalent to G4Tubs) and house Scintillating or Cherenkov (clear) optical fibers. In the following optical fibers inside tubes are not displayed to aid visibility.
- Tubes z-axis is always parallel to the tower axis, i.e. they are projective pointing to the IP
- Tubes are placed starting from the back face of the tower (the biggest one) and,
- the tube length is changed towards the edges of the Trap to get the trapezoidal shape (examples are showed in the following)

Tubes placement (y-direction)

A tower back face

A tube column (of Scintillating fibers) whose x-y position is distributed over the the back face of a tower

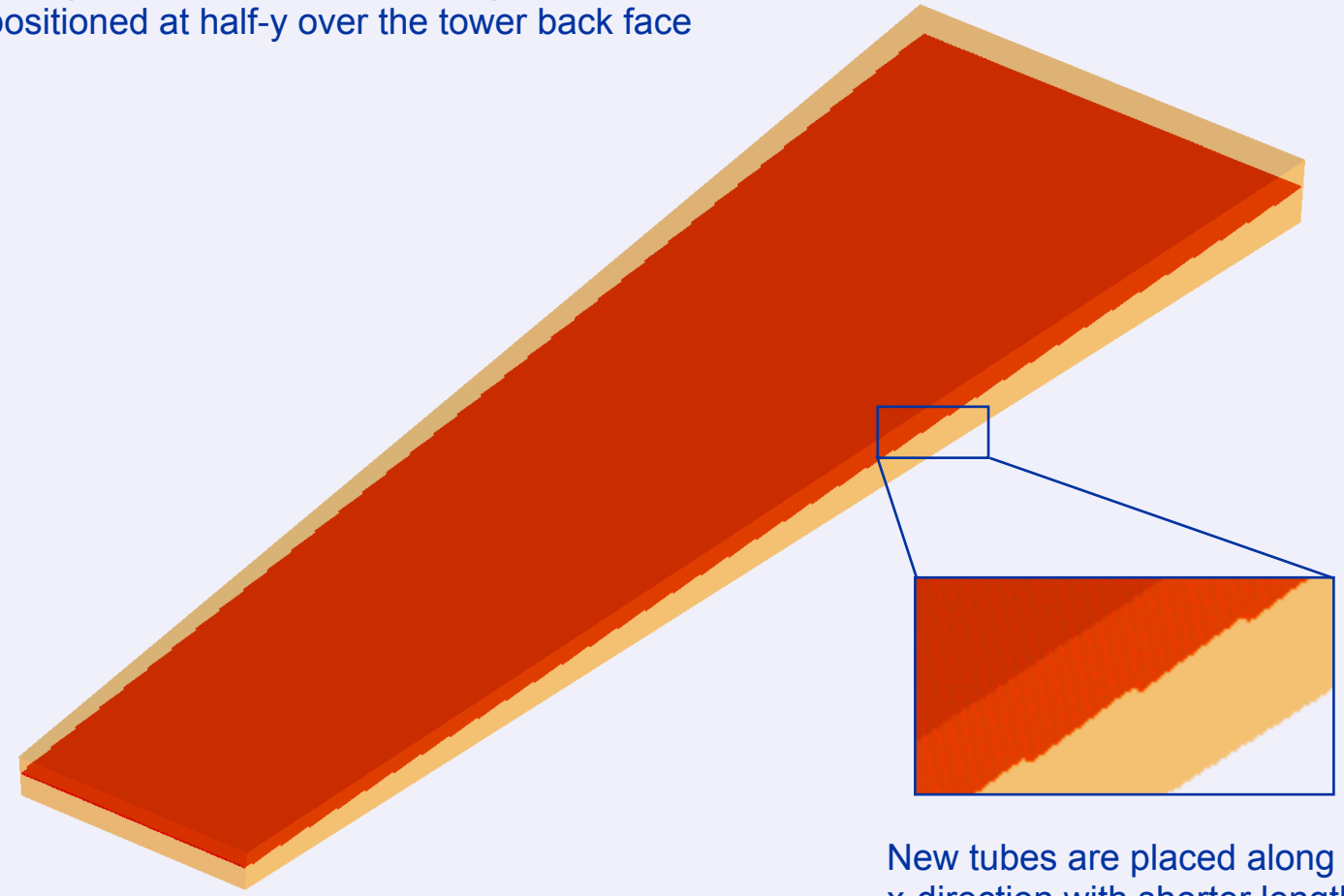


- For each tube starting at x-y from the back face, I calculate the intersection of a line parallel to the z-axis passing through x-y with each of the 4 planes in figure + the front face
- The shortest intersection defines the tube length
  - In reality some corrections are needed because the intersection happens between a plane and a cylinder (the tube)
- Tubes intersecting with the front face have the same length and are represented by the same Solid object in memory
- Tubes intersecting with any other plane are of specific length and are created on the fly

Tube radius was increased to 2 mm to help visualization

Tubes placement (x-direction)

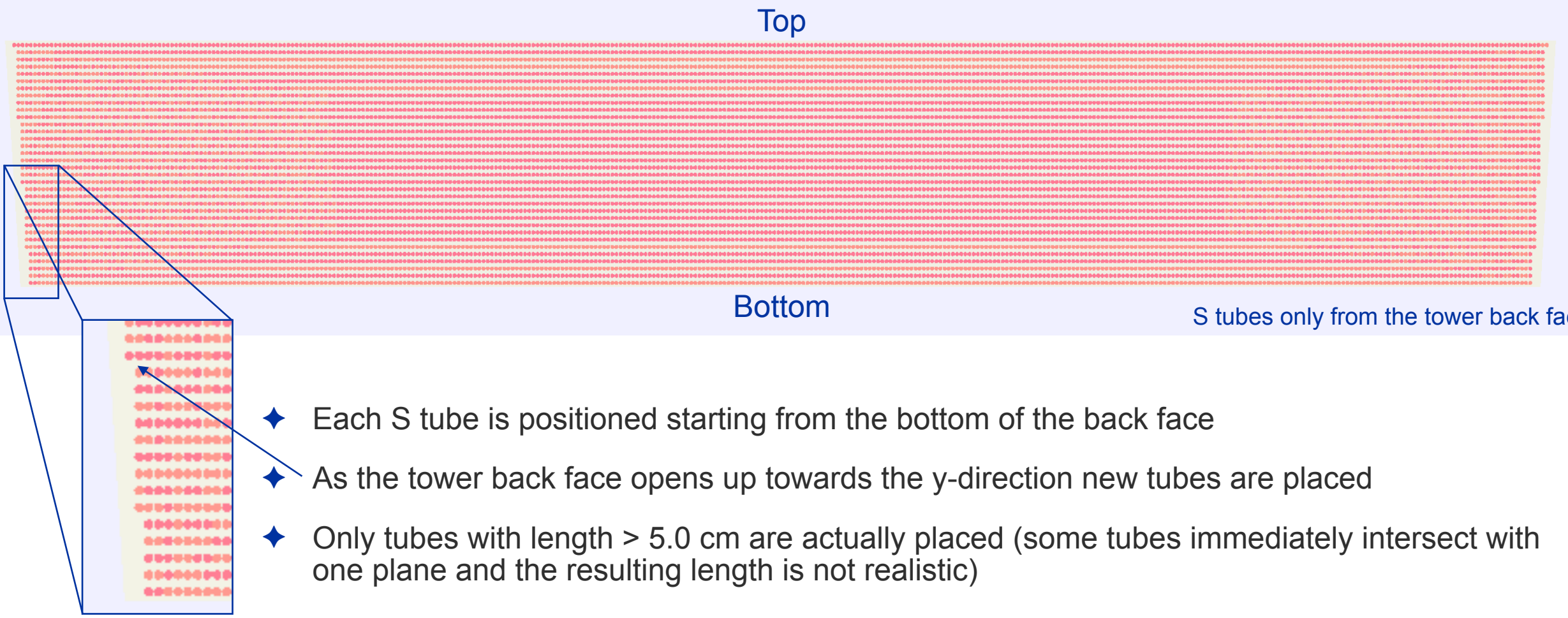
A single tube row (of Scintillating fibers)
positioned at half-y over the tower back face



New tubes are placed along
x-direction with shorter length

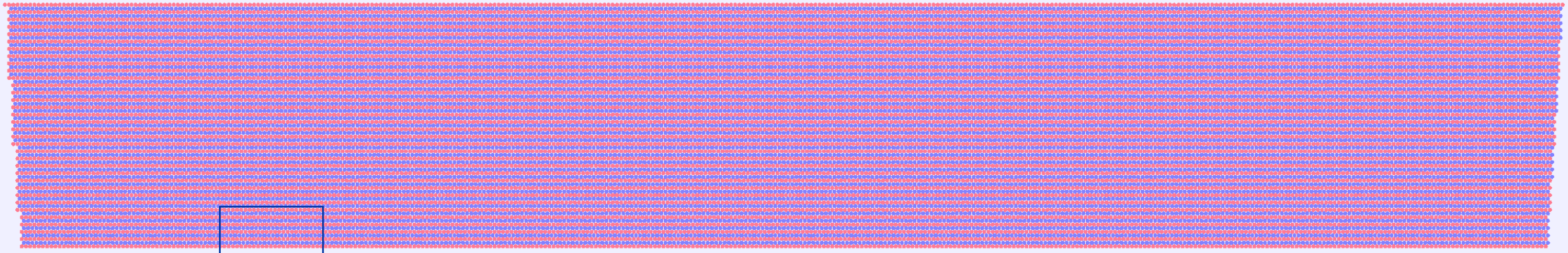
Tube radius was increased to 2 mm to help visualization

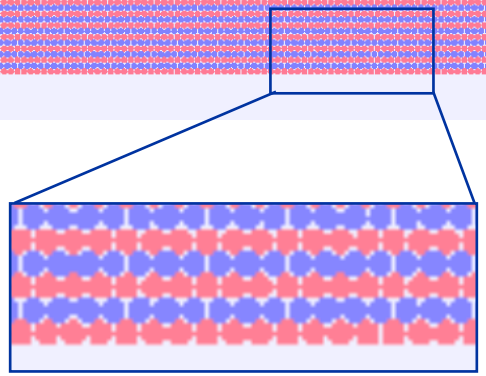
Filling the towers (Scintillating tubes only)

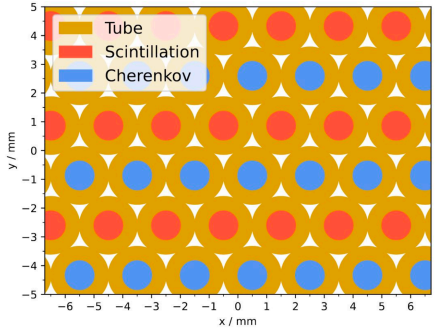


- Each S tube is positioned starting from the bottom of the back face
- As the tower back face opens up towards the y-direction new tubes are placed
- Only tubes with length > 5.0 cm are actually placed (some tubes immediately intersect with one plane and the resulting length is not realistic)

Filling the towers (all tubes)



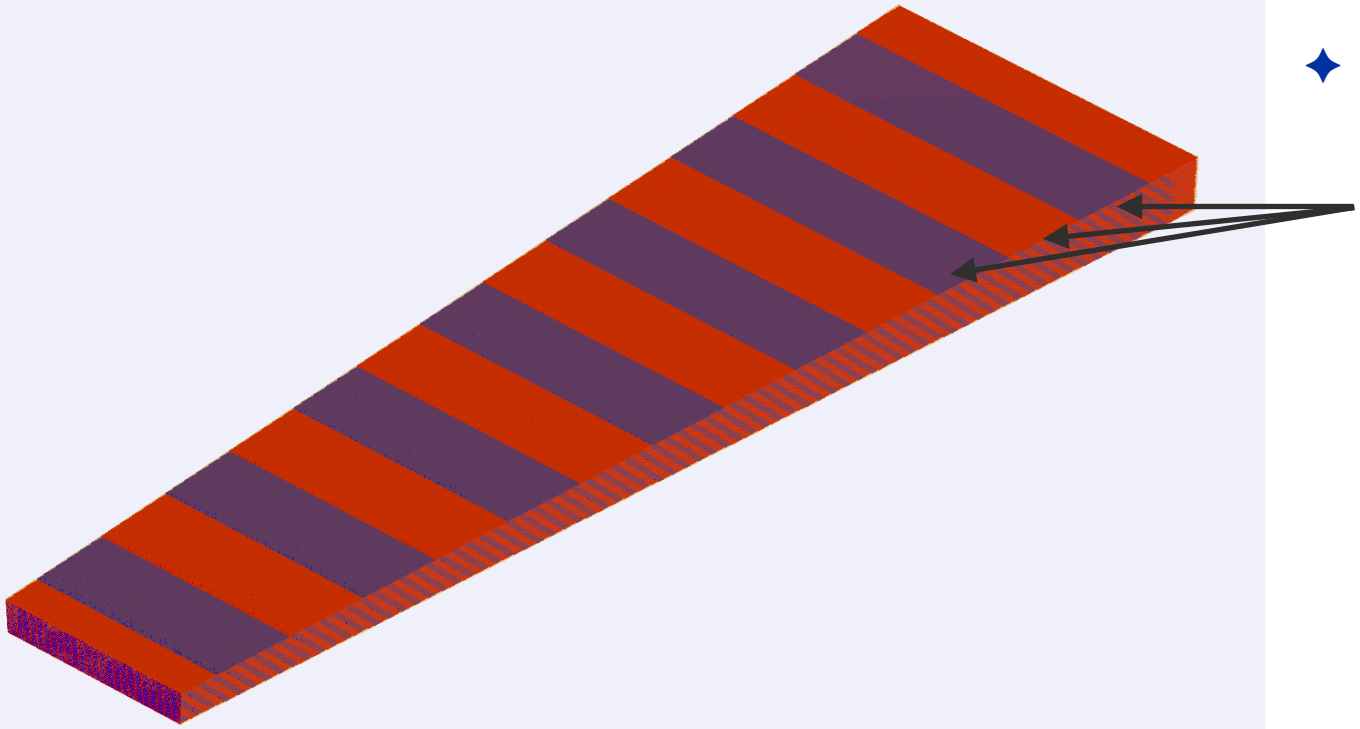




- The same technique applies to Cherenkov tubes housing clear plastic fibers
- C tubes (blue) are placed as rows in between S ones (red)

Filling the towers (all tubes)

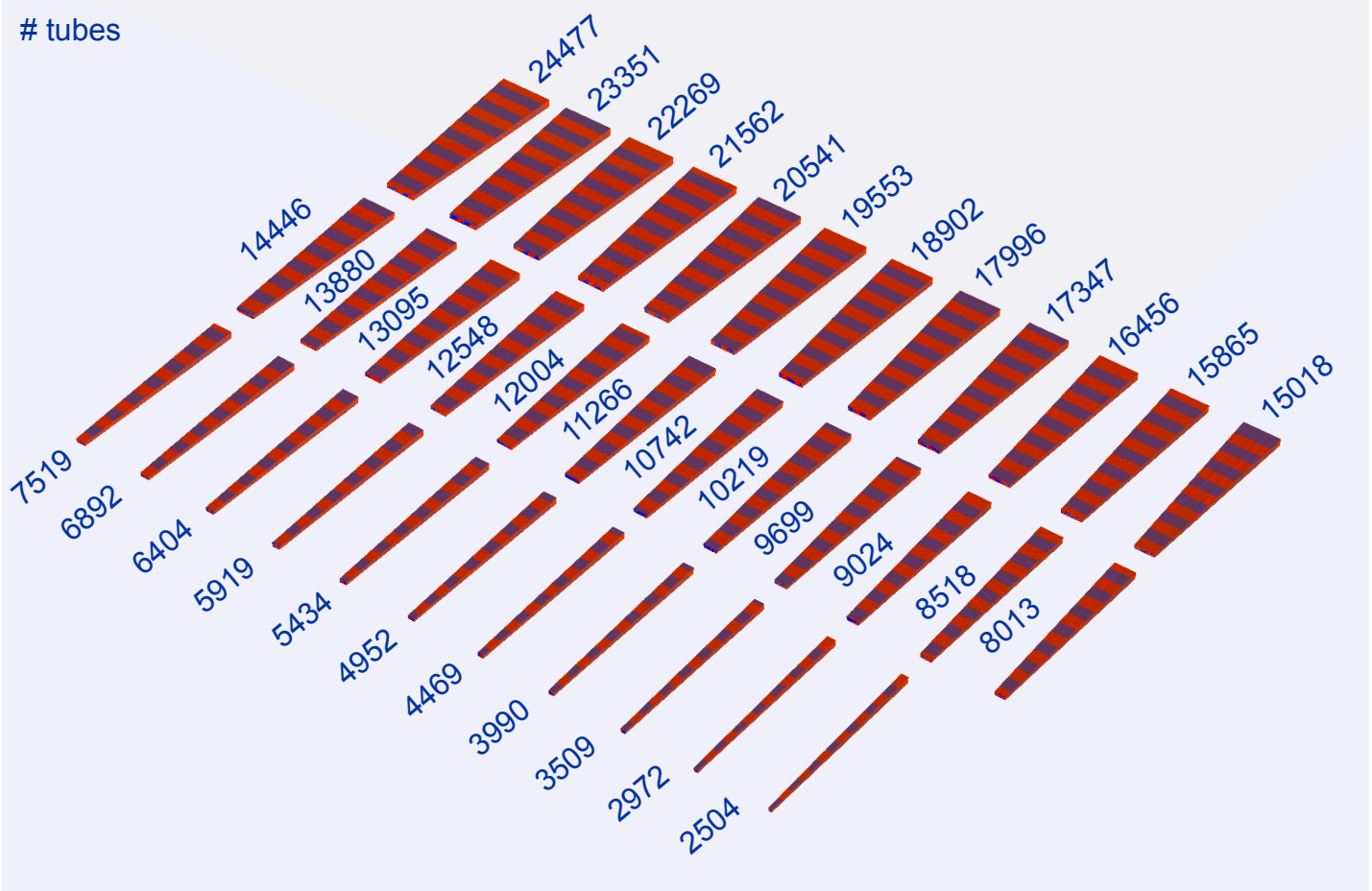
A single tower filled with 24477 tubes



- A tower fully filled with tubes (S and C)
- S and C tubes alternates in y direction and the y-position determines the tube length
  - hence the colored bands visible from the top view

Some numbers

- Considering the default parameters (36 rotations around ϕ, 35 towers along θ, inner radius 2.5 m, 2.5 m long towers)
  - tubes per ϕ-slice: 421325
  - tubes per both endcaps: 30335400*
  - total tube length: 47128.5 km
- If we reduce the tower length to 2 m
  - tubes per ϕ-slice: 346877
  - tubes per both endcaps: 24975144
  - total tube length: 33072.5 km

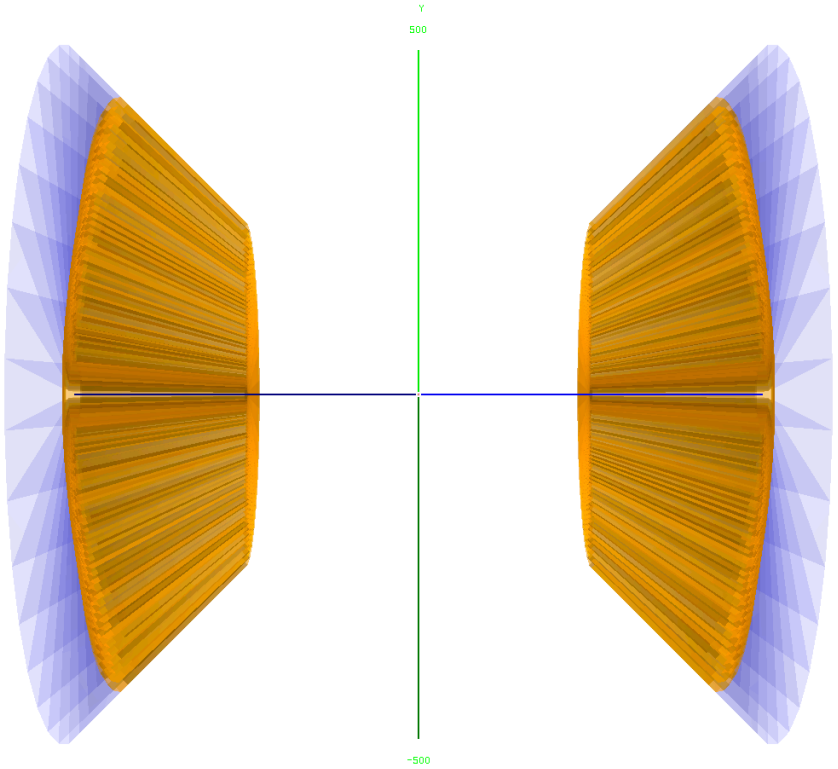


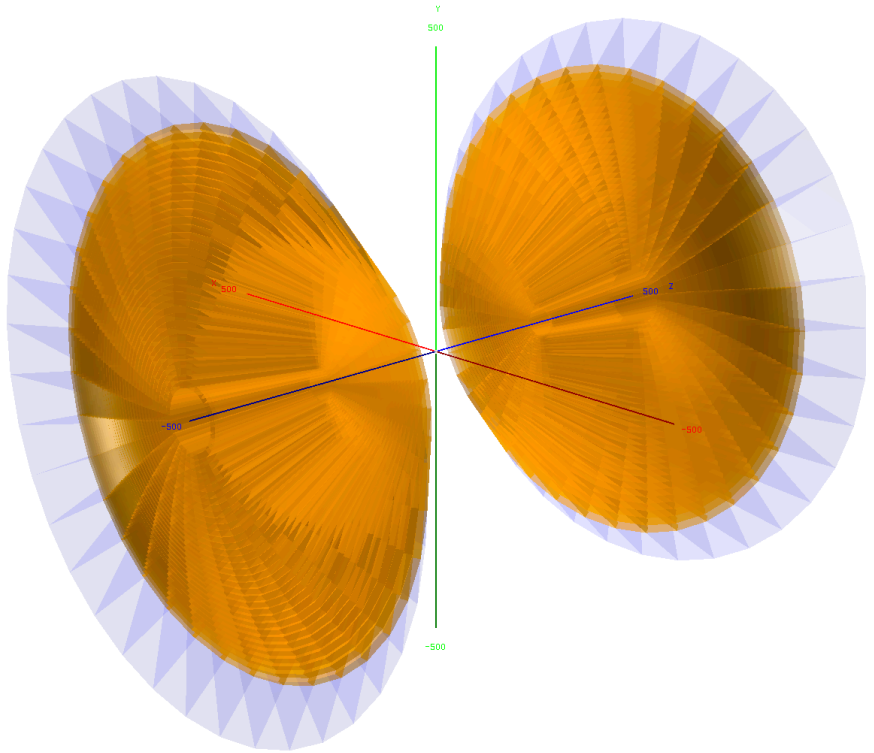
*the number of Tubs objects in memory is ~295648 (only counting for the absorber tubes)

Final geometry

- The final geometry is obtained with a simple repetition of the ϕ-slice around the z-axis (right endcap, $z > 0$), and a reflection + translation for the left endcap ($z < 0$)

? Please, let me know if there is any better way to do reflections in DD4hep

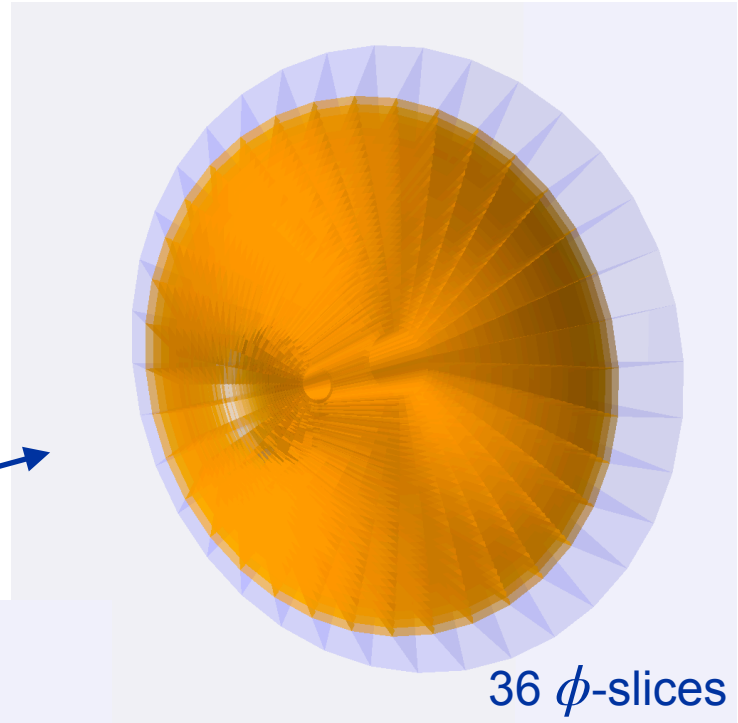


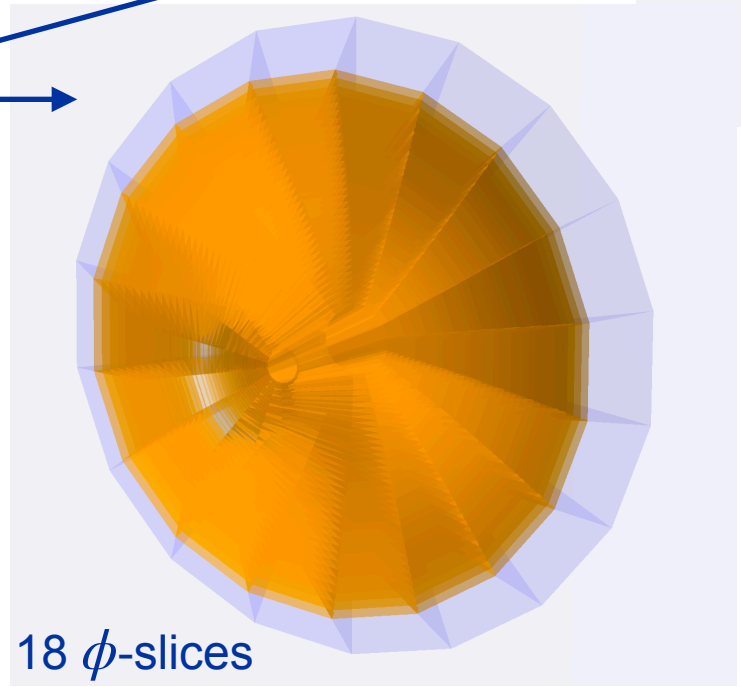


Modularity

- All the geometry custom parameters are encapsulated in the XML description file

```
<define>
  <constant name="world_side" value="6*m"/>
  <constant name="world_x"    value="world_side/2"/>
  <constant name="world_y"    value="world_side/2"/>
  <constant name="world_z"    value="world_side/2"/>
  <constant name="innerRadius"value="2.5*m"/>
  <constant name="towerHeight"value="2.5*m"/>
  <constant name="NbOfZRot"   value="36"/>
  <constant name="TubeRadius" value="1.0*mm"/>
  <constant name="CladRadius" value="0.5*mm"/>
  <constant name="CoreRadius" value="0.45*mm"/>
</define>
```



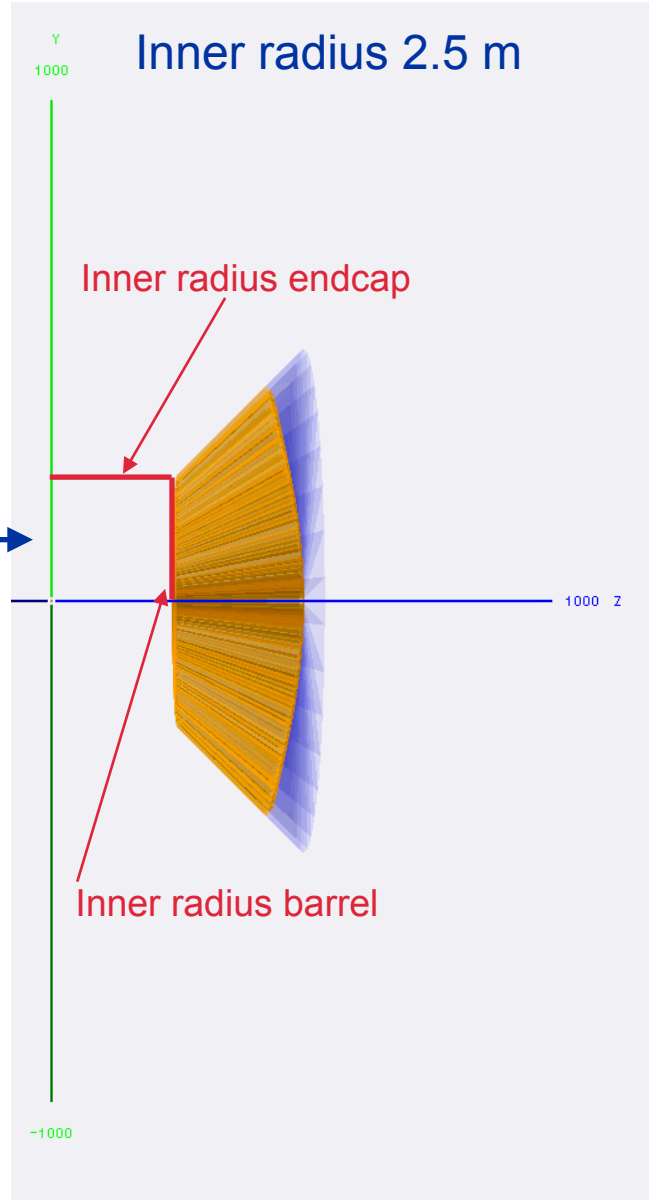


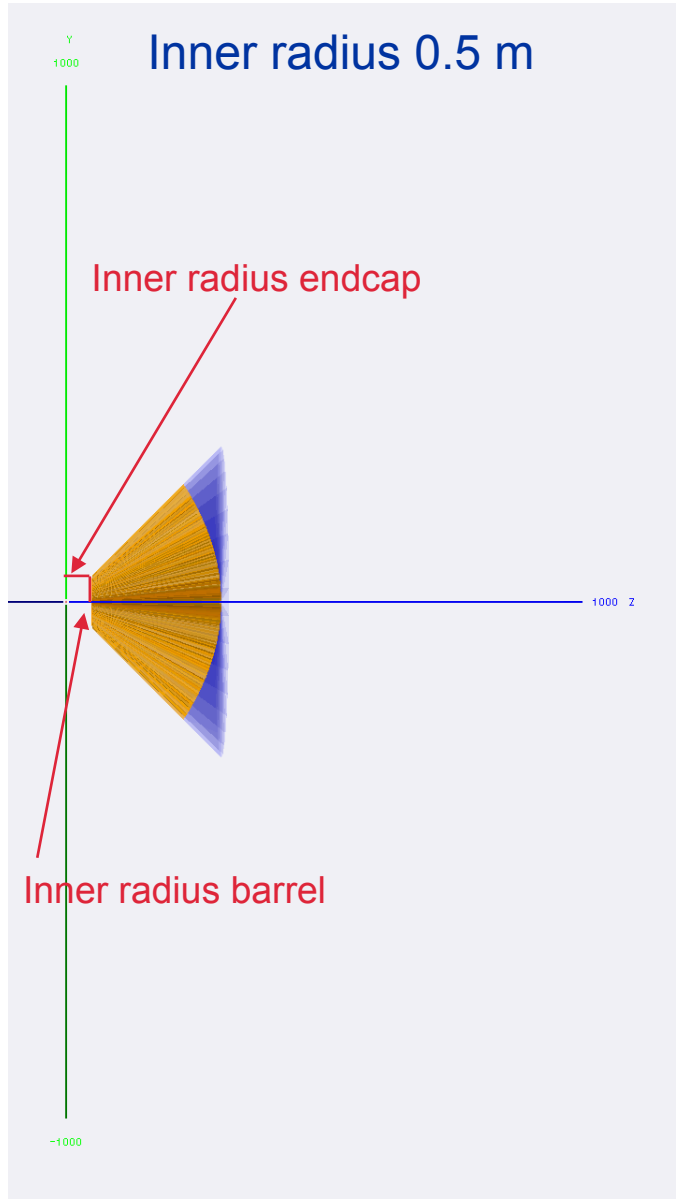
Modularity

- All the geometry custom parameters are encapsulated in the XML description file

```
<define>
  <constant name="world_side" value="6*m"/>
  <constant name="world_x"    value="world_side/2"/>
  <constant name="world_y"    value="world_side/2"/>
  <constant name="world_z"    value="world_side/2"/>
  <constant name="innerRadius"value="2.5*m"/>
  <constant name="towerHeight"value="2.5*m"/>
  <constant name="NbOfZRot"   value="36"/>
  <constant name="TubeRadius" value="1.0*mm"/>
  <constant name="CladRadius" value="0.5*mm"/>
  <constant name="CoreRadius" value="0.45*mm"/>
</define>
```

- The only geometry constrain: the barrel inner radius and the endcap inner radius are identical or, equivalently, the endcap starts at $\theta = \pi/4$





Conclusions

- A DD4hep geometry for the IDEA endcap calorimeter using the INFN capillary tubes technology was implemented
  - Software-wise it will be isolated code from the barrel description but used with it (we will not have a single IDEA tubes calo code)
- Next steps:
  - If you agree, I might complement the barrel description of this detector in the Feasibility Study Report with the endcap part
  - Then we should validate the code in terms of compatibility with the barrel and navigation (overlaps, stacked tracks, …),
  - add the sensitive detector from the test-beams simulation performed with capillary tubes calorimeter prototypes,
  - and finally port this geometry to the FCC sw (some help from the experts is needed here…)