About config file

September 22, 2011

1 Underscore notation (_y etc)

A detector is described by "onfig" file in which we write a description of objects (box, cylinder etc; volume-shape). Any volume-shape has a unique canonical form for which the origin of the coordinate and the attributes of the volume are defined. We use three coordinate systems: canonical, local and world. So far (v9.08 or earlier) the canonical and local coordinates were the same.

Any volume may be rotated and/or shifted to construct a detector system. The rotation frequently used is 90° rotation. For such a rotation, special notation was available for "prism" "cyl(inder)" and "pipe": for example, "pipe" means the canonical form, while "pipe_y" is a variant of 90° rotation. (See Fig.) In the older versions, the

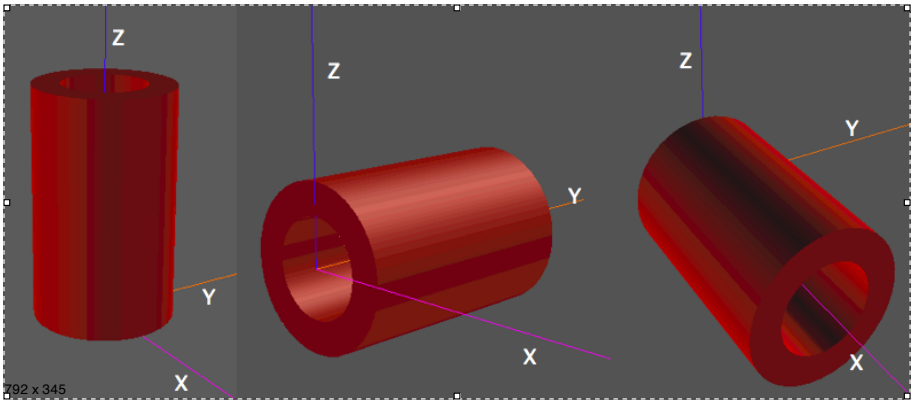

Figure 1: From left. pipe, pipe_y and pipe_x. "pipe" is the canonical form(=pipe_z).

local coordinate of "pipe_y" is the same as the coordinate of the canonical form.

From version 9.081, however, the internal treatment was changed so that "pipe_y" itself defines the local coordinate[1].

If the user wants to coincide the local and canonical coordinate for rotated form, explicit rotation may be specified like "pipe 0 0 0 / 0 0 0 a b h 0 1 0" which will show up the same figure as "pipe_y". Similarly "pipe 0 0 0 / 0 0 0 a b h 1 0 0" will look like "pipe_x".

The particle position and direction informed to the user (via userde or userbd) are always those in the local coordinate.

Therefore, one and the same position and direction could be different in the local coordinate informed to the user depending on whether one uses "cyl_y" or "cyl" with

[1]When particle tracking is performed internally, the intrinsic canonical coordinate is still used. Therefore, with "_y" etc notation, the local and canonical coordinate are different.

rotation specification. However, if they are converted to the world coordinate, the local coordinate difference disappears.

1.1 Another change and caution

In the older versions (9.08 or earlier), the name of a volume-shape must be 8 characters or less. Now it is extended to 12 characters. If one uses "call epqstruc(n, struc)" to know the volume-shape name of the $n-$th component, "struc" must be 12 or more character length. If it is less than 12, say, 8, "honeycomby" will be informed as "honeycom".

2 octagon

The canonical octagon is shown in Fig. 2. An octagon could be made simply by using a box which contains 4 prisms made of "sp" at the 4 corners. However, this will increase the number of components and such a structure has difficulty in light transport inside the octagon.

An example of a canonical octagon in a config is

```
#news   new-1   octagon
1 octagon PWO 3 2 / 0 0 0 a b c d
```

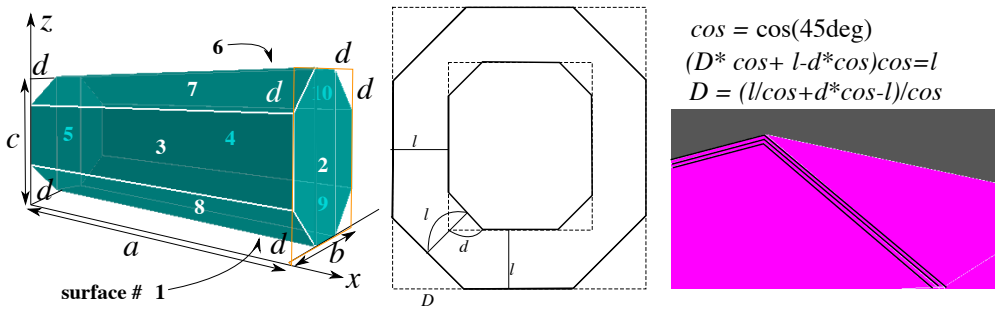

Figure 2: Canonical octagon (left). Volume atribute a, b, c, d and surface number definition are shown. d is common to all 4 corners. Formula for making clad structure (right)

We can use octagon_y and octagon_z[2].

3 honeycomb

A honeycomb could be also made by using a number of small unit cells. However, we make a new volume-shap for the honeycomb. The canonical form is illustrated in Fig.3.

The effective honeycomb area is the shaded one. Other parts are non existent in the simulation although drawing may show some of them. In config file, the user must always define a subdetector with a tight world (i.e., box_w sp 0 0 0 / Xs Ys 0 a b h); sp may be air, H2O etc in some case. If Xs=Ys=0, you may simply write box_w sp 0 0 0 /0 0 0.

[2]Note, in this case, the canonical form is octagon_x.

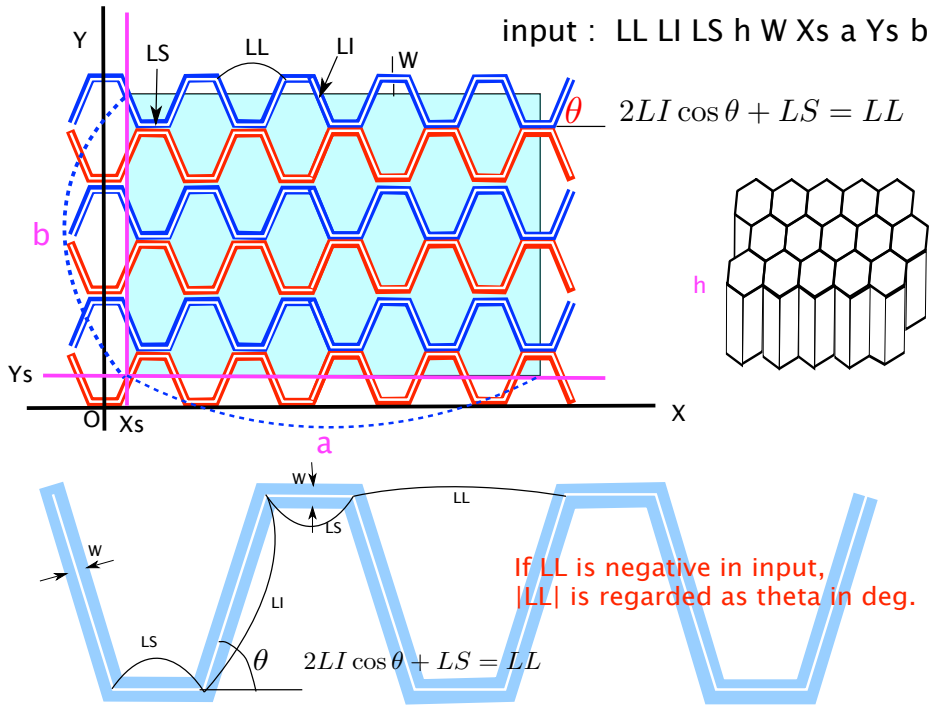

Figure 3: Honecycomb

if LL is negative, -LL is regarded as θ in deg. Hence, -60 0.5 0.5 ... means a honeycomb with a regular hexagon cell (to be exact, if W=0) For honeycomb too, we can use honeycomb_y and honeycoomb_z (canonical one is regarded as honeycomb_x).

One example of a honeycomb is

```
#news  new-1  honeycomb
#eq    sq  35
#eq    sq2  40
#subd  honey
1 honeycomb Al     3  0  0 /  0   0  0  -60 1 0.6  1  0.05 0 sq  0 sq
2 box_w      sp    0  0  0 /  0 0 0
#end honey
1 box W       0 0 0  /  0 0  0     sq2 sq2 0.5
2 box Carbon  0 0 0  /  0  0  +    =  = 0.1
3 honey  sp   0   0   0 / 2.5  2.5   +
4 box Carbon  0 0 0  /  0  0  +    sq2  sq2 0.1
5 box W       0 0 0  /  0 0  +     =  = 0.5
6 box_w sp      0 0 0  / 0 0 0
```

It is to be noted that even if you use honeycomb_y or honeycomb_z, `#news` line may contain only honeycomb as shown above.

3.1 Honeycomb drawing

Since the unit cell is normally small, and the area covered by the honeycomb is large, the number of vertexes becomes order of few 10 thousands. For such case, default drawing omits the central part of the honeycomb (though some abbreviated form may appear as

in Fig.3.1. Also, currently, the wall thickness is not shown. To show all the honeycomb elements, you have to give a value > 0.5 to "portion " in Util/Data/honeycomb.dat

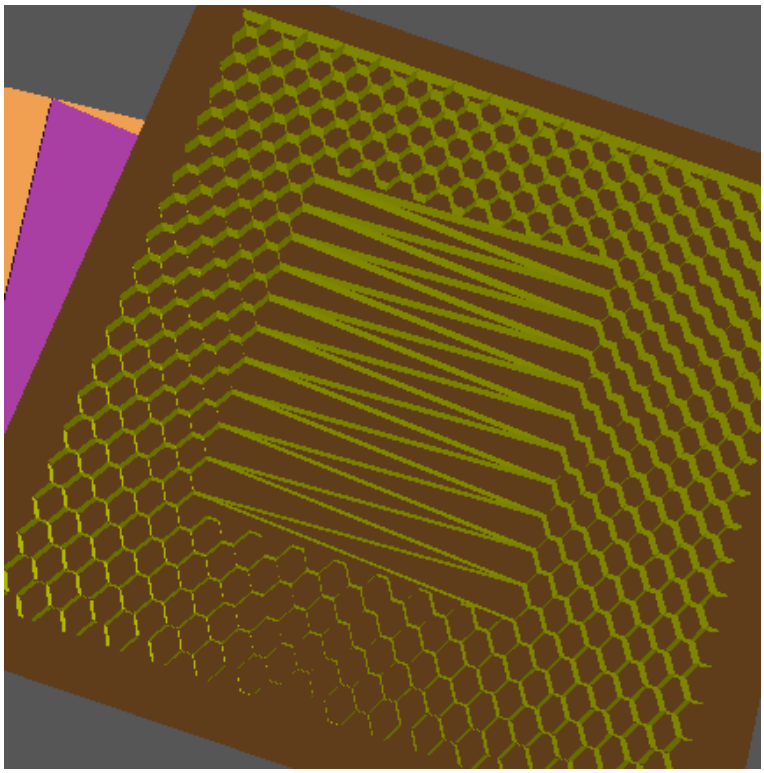

Figure 4: In default drawing, the central region of a honeycomb will be simplifed

4 Note on prism

As described in the old manual, prism has 5 variants with prsim_yz type notation. However, prism permits negative values for some of the volume attributes. And many of the form will be accomplished without rotation. Example config is given below to make a octagon from combination of prism and box.

```
-------------------------------------------------
#eq   cut 0.5d0
#eq  ncut -0.5d0
#eq   a   2
#eq   b   30
#eq   c   1.9d0

1   box  SCIN   0  0  0   / 0  0  0    a   b c  / 2 3 4 5
2   prism Cu  0  0  0   / 0  0  0    cut  b    0  cut
3   prism Cu  0  0  0   / 0  0  c    cut  b    0  ncut
4   prism Cu  0  0  0   / a  0  0    ncut  b   0   cut
5   prism Cu  0  0  0   / a  0  c    ncut  b  0   ncut
6   box_w sp  0  0  0   / -1 -1 -1   4    32  3.9
-------------------------------------------------
```

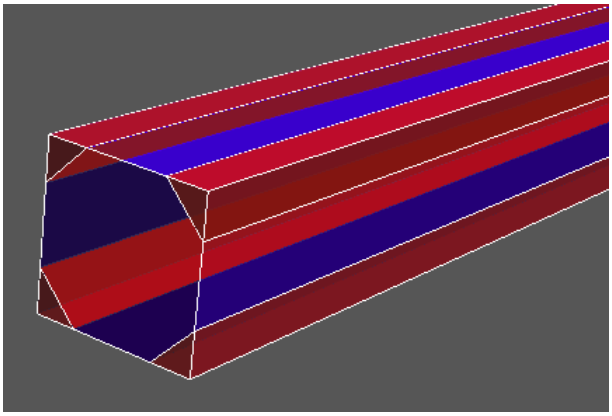

Figure 5: Prism permits negative attributes