How to define a phantom

Any geometry definition is set in square brackets:
[*type* : *commands*]

type defines the geometry type, the details are specified with the *commands*.

The possible types of geometries and the meaning of the commands are explained in the next sections. **All geometry values are zero by default.**

Volume geometries

are needed to define the phantom objects.

Volume geometry types

Volume geometries are used to compose a phantom from objects. The variables `x`, `y` and `z` define (except for cones and tetrahedrons) the object's center of gravity coordinates. Alternatively the object center may be defined by specifying the cartesian vector `center(`*expression*`,`*expression*`,`*expression*`)`. All paramaters are initialized to zero. Note that all objects can be combined with clip planes (see section Clip planes).

`Sphere`
: sphere with radius `r`.

`Box`
: box with edges parallel to the coordinate axis and edge lengths `dx`, `dy`, `dz`.

`Cylinder_x`
: cylinder with length `l` and radius `r` parallel to the x axis.

`Cylinder_y`
: cylinder with length `l` and radius `r` parallel to the y axis.

`Cylinder_z`
: cylinder with length `l` and radius `r` parallel to the z axis.

`Cylinder`
: cylinder with length `l` and radius `r`. Vector `axis(`*expression*`,`*expression*`,`*expression*`)` defines the (x,y,z)-direction of the axis.

`Ellipsoid`
: triaxial ellipsoid with *half* axis `dx`, `dy`, `dz`; all axes parallel to the coordinate axes.

`Ellipsoid_free`
: triaxial ellipsoid with arbitrary axis directions given by any two of the following vectors forming a mutually orthogonal set: `a_x(`*expression*`,`*expression*`,`*expression*`)`, `a_y(`*expression*`,`*expression*`,`*expression*`)` and `a_z(`*expression*`,`*expression*`,`*expression*`)`. The two vectors have to be orthogonal but not necessarily normalized. The sizes along these axes are given by the *half* axis lengths `dx`, `dy` and `dz`.

`Ellipt_Cyl`
: cylinder with elliptical cross section, arbitrary orientation in space. Length `l`, half axis `dx`, `dy`. The orientation is defined by any two of the following vectors forming a mutually orthogonal set: `axis(`*expression*`,`*expression*`,`*expression*`)` points in axis direction, `a_x(`*expression*`,`*expression*`,`*expression*`)` defines the direction of the

ellipsis axis given by `dx`, `a_y(`*`expression`*`,`*`expression`*`,`*`expression`*`)` defines the direction of the ellipsis axis given by `dy`. The two vectors have to be orthogonal but not necessarily normalized.

`Ellipt_Cyl_x`
: cylinder with elliptical cross section, axis parallel to the x axis, length `l`, half axis lengths `dy`, `dz`.

`Ellipt_Cyl_y`
: cylinder with elliptical cross section, axis parallel to the y axis, length `l`, half axis lengths `dx`, `dz`.

`Ellipt_Cyl_z`
: cylinder with elliptical cross section, axis parallel to the z axis, length `l`, half axis lengths `dx`, `dy`.

`Cone`
: truncated cone, arbitrary orientation in space as given by the axis `axis(`*`expression`*`,`*`expression`*`,`*`expression`*`)`. Length `l`, radii `r1` and `r2` at the two ends of the truncated cone. Moving along the cone axis in the direction given by `axis`, first the end corresponding to `r1`, then the end corresponding to `r2` is met. `x`, `y` and `z` define the center of the truncated cone axis.

`Cone_x`
: truncated cone of length `l`, radius `r1` at the end with smaller x, radius `r2` at the end with larger x, axis parallel to the x axis. `x`, `y` and `z` define the center of the truncated cone axis.

`Cone_y`
: truncated cone of length `l`, radius `r1` at the end with smaller y, radius `r2` at the end with larger y, axis parallel to the y axis. `x`, `y` and `z` define the center of the truncated cone axis.

`Cone_z`
: truncated cone of length `l`, radius `r1` at the end with smaller z, radius `r2` at the end with larger z, axis parallel to the z axis. `x`, `y` and `z` define the center of the truncated cone axis.

`Tetrahedron`
: tetrahedron with the four corners given by the (x,y,z) vectors `p1(`*`expression`*`,`*`expression`*`,`*`expression`*`)`, `p2(`*`expression`*`,`*`expression`*`,`*`expression`*`)`, `p3(`*`expression`*`,`*`expression`*`,`*`expression`*`)` and `p4(`*`expression`*`,`*`expression`*`,`*`expression`*`)`. `x`, `y`, `z` are meaningless. Note that every polyhedron can be composed of tetrahedrons. Another way to create *convex* polyhedrons is to treat some object (e.g, a sphere) with clip planes (see section Clip planes).

Clip planes

All volume geometries can be combined with one or several **clip planes**. These clip planes remove parts of the object not contained in the half-space defined by the plane. Clip planes can be defined in two ways:

- By specifying an inequality of the form

  `x <` *expression*
  or
  `x >` *expression*

which excludes all points of the object which do not fulfill the inequality (analog for y, z).

- By specifying the normal of an arbitrary plane and its distance to the origin:

  `r(`*expression,expression,expression*`) <` *expression*
  or
  `r(`*expression,expression,expression*`) >` *expression*

  This specification is to be understood as *intersection with the set of all points that obey the equation **r.n** <(>) a*, where **n**=**v**/|**v**|, **v** is the given vector, a is the value of the right hand side expression and '.' stands for the scalar product.

Hence, if **n** is the normal vector of the clip plane pointing *outside* the relevant region (*inside* the region which is to be cut off), the specification is `r(`**n**`) <` **n**.**p**, where **p** is an arbitrary point in the clip plane.

If several clip planes (of any type) are given, the object is intersected with all given half-spaces.

Sample geometries

Here are a few examples for volume geometry definitions:

Without clip planes:

1. `[Sphere: r = 4]` creates a sphere of radius 4 cm around the origin,
2. `[Box: x = 1 y = 1 z = 2 dx = 2 dy = 2 dz = 4]` creates a box of 2*2*4 cm3 with one corner located in the origin,
3. `[Cylinder: l=10 r=2 axis(1,1,1)]` creates a cylinder of length 10 cm and a diameter of 4 cm centered in the origin and pointing into the (1,1,1) direction,
4. `[Tetrahedron: p1(0,0,0) p2(1,0,0) p3(0,1,0) p4(0,0,1)]` creates a tetrahedron with the four corners (0,0,0), (1,0,0), (0,1,0) and (0,0,1).

Using clip planes:

1. `[Sphere:r=5 x<0 y<0]` produces a quarter sphere (negative x and y) with the curvature center in the origin,
2. `[Sphere:x=-4 r=5 x>0]` produces an object in the shape of a planoconvex lens of thickness 1 cm, the convex part pointing into the positive x direction,
3. `[Box:x=0.5 y=0.5 z=0.5 dx=1 dy=1 dz=1 r(1,1,1)<1/sqrt(3)]` creates the same tetrahedron as in example 4 by truncating a box,
4. `[Sphere:r=100 x>0 y>0 z>0 x<2 y<2 z<4]` creates the same box as in example 2 by truncating a sphere.

Surface geometries

are used for detectors, antiscatter grids, etc.

The variables `x`, `y` and `z` define the surface reference point. Alternatively the surface reference point may be defined by specifying the cartesian vector `center(`*expression*,*expression*,*expression*`)`. The extension of the plane is given relative to this point. *Please note:* this point is always in the detector surface.

`Plane_xy`
: rectangular plane parallel to the xy plane.

`Plane_xz`
: rectangular plane parallel to the xz plane.

`Plane_yz`
: rectangular plane parallel to the yz plane.

`Plane`
: arbitrary rectangular plane. It is defined through any *two* of the following vectors:
`norm(`*expression*,*expression*,*expression*`)`, the surface normal
`a_x(`*expression*,*expression*,*expression*`)`, the row direction ("east") and
`a_y(`*expression*,*expression*,*expression*`)`, the column direction ("north"). The two vectors have to be orthogonal but not necessarily normalized.

`Cylindrical_z`
: cylindrical geometry, axis parallel to the z axis. Rectangular region on the cylinder surface. The angular extension of the detector is defined relative to the vector from the axis (defined elsewhere) to the surface reference point.

`Cylindrical`
: cylindrical geometry, arbitrary axis given by `axis(`*expression*,*expression*,*expression*`)`. Rectangular region on the cylinder surface. The angular extension of the detector is defined relative to the vector from the axis (defined elsewhere) to the surface reference point.

`Spherical`
: spherical cone geometry. The surface reference point is irrelevant.