FLUKA advanced geometry

Selected topics to build a modular geometry

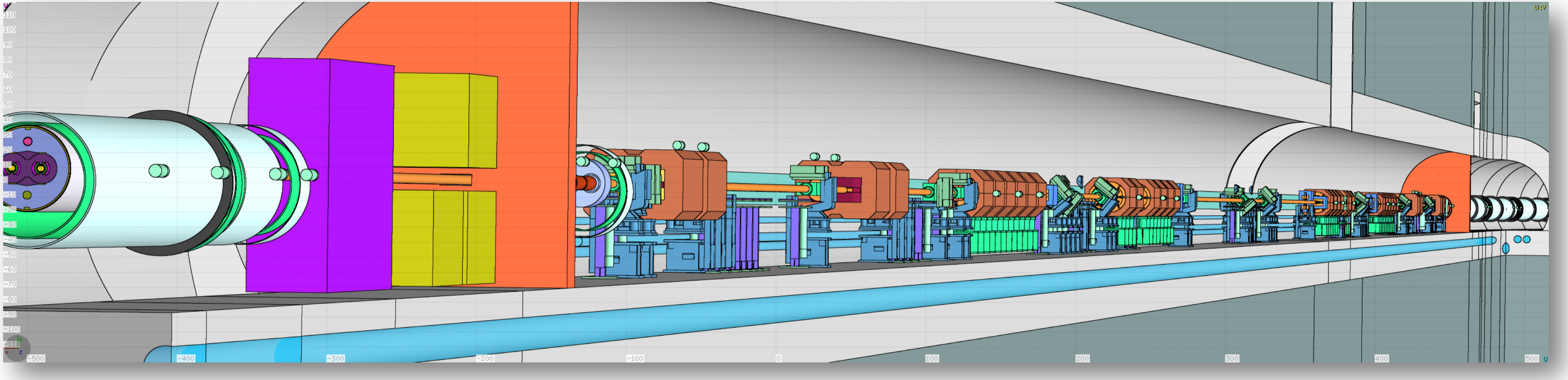
Basic Geometry Concepts

Three concepts are fundamental in the FLUKA Combinatorial Geometry, which have been described earlier in the course:

- **Bodies**: basic convex objects + infinite planes & cylinders + generic quadric
- **Zones**: portion of space defined by intersections **(+)** and subtractions **(-)** of bodies (used internally)
- **Regions**: union of multiple zones **(|)**
  (it can be also be a single zone)

Complex and modular geometry

3D rendering of the LHC IR7 with Collimators and BLM



Complex and modular geometry models like the one shown here are built with the LineBuilder [A. Mereghetti et al., IPAC2012, WEPPD071, 2687]

- Note that such geometry model heavily depends on **lattices** (i.e. duplication of existing regions), which is not covered here
- More examples of complex and modular geometries are available at
  - http://fluka.cern
  - http://cern.ch/flair

Contents

- Roto-translation transformations
  - `ROT-DEFIni` card
- Geometry directives
  - `translat`
  - `transform`
  - `expansion`
- Additional card related to a transformation
  - `ROTPRBIN` card
- Build a modular geometry with bounding boxes

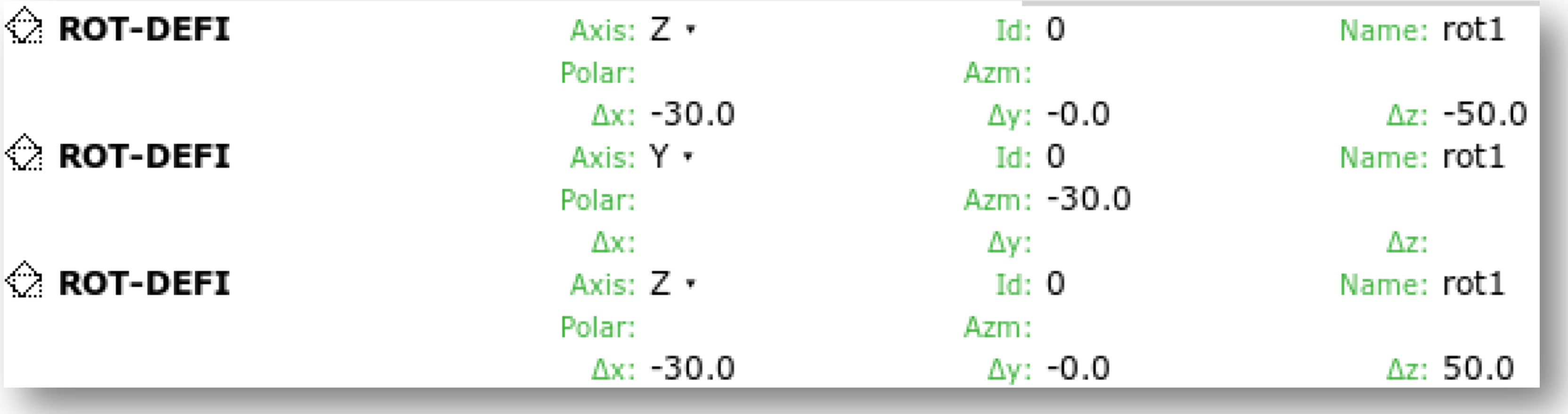
ROT-DEFIni card – Introduction

- The ROT-DEFIni card defines roto-translations that can be applied to

i. Bodies

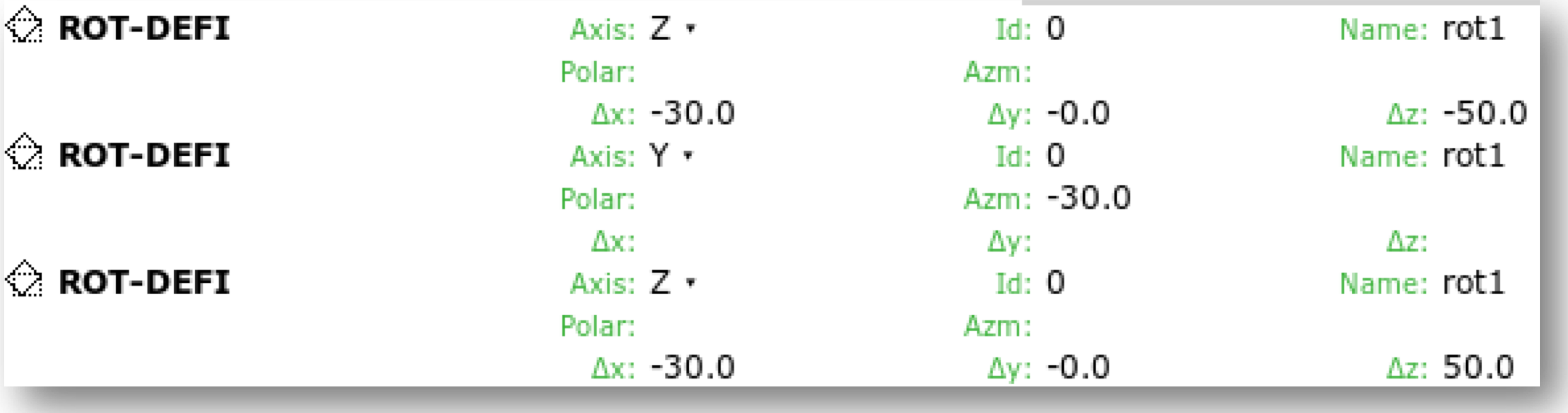
ii. USRBIN and EVENTBIN cards (see ROTPRBIN card later)

iii. LATTICE (not covered here)



```
ROT-DEFI        Axis: Z ▾      Id: 0          Name: rot1
               Polar:         Azm:
                  Δx: -30.0    Δy: -0.0         Δz: -50.0
ROT-DEFI        Axis: Y ▾      Id: 0          Name: rot1
               Polar:         Azm: -30.0
                  Δx:          Δy:              Δz:
ROT-DEFI        Axis: Z ▾      Id: 0          Name: rot1
               Polar:         Azm:
                  Δx: -30.0    Δy: -0.0         Δz: 50.0
```

ROT-DEFIni card – Definition



```
ROT-DEFI       Axis: Z ▾         Id: 0            Name: rot1
               Polar:            Azm:
                 Δx: -30.0         Δy: -0.0         Δz: -50.0
ROT-DEFI       Axis: Y ▾         Id: 0            Name: rot1
               Polar:            Azm: -30.0
                 Δx:               Δy:              Δz:
ROT-DEFI       Axis: Z ▾         Id: 0            Name: rot1
               Polar:            Azm:
                 Δx: -30.0         Δy: -0.0         Δz: 50.0
```

Axis	rotation with respect to axis
Id	transformation index
Name	transformation name (optional but recommended)
Polar	polar angle of the rotation $\mathbf{R_2}$ ($0 \le \vartheta \le 180$ degrees)
Azm	azimuthal angle of the rotation $\mathbf{R_1}$ ($-180 \le \varphi \le 180$ degrees)
Δx, Δy, Δz	offset for the translation **T**

In a ROT-DEFI, the transformation is defined as $X_{new} = \mathbf{R_2}(\vartheta) \times \mathbf{R_1}(\varphi) \times (X_{old} + \mathbf{T})$

- Refer to the manual [**ROT-DEFIni** section, Note 4] for the rotation convention adopted in FLUKA
- It is preferable to define a rotation through the azimuthal angle $\mathbf{R_1}(\varphi)$
- Clearly the order of the rotations/translation is relevant (they do not commute)
- As in the example, transformations can be conveniently defined with multiple ROT-DEFI cards, which are identified by the same id/name (recommended!!!)

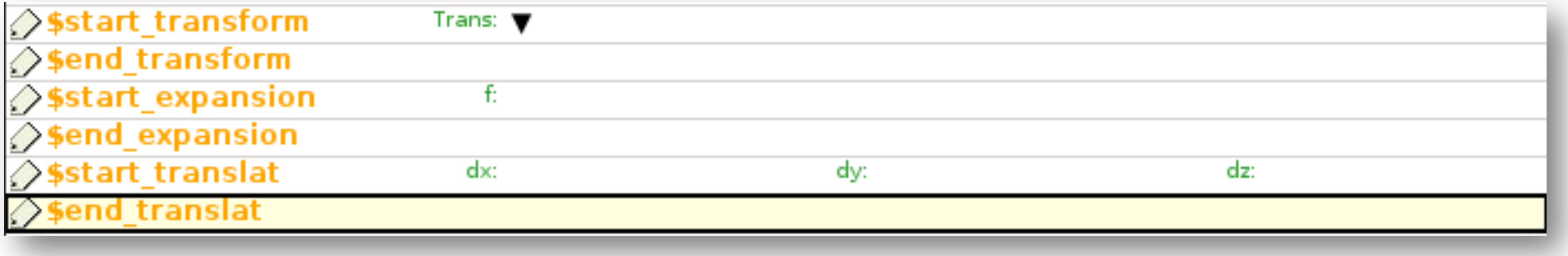
Geometry directives

- Special commands enclosing a body (or a list of bodies) definition:

```
$start_xxx
...
$end_xxx
```

where "`xxx`" stands for “`expansion`“, "`translat`" or "`transform`”

- The directive is applied to the list of the bodies embedded between the starting and the ending directive lines

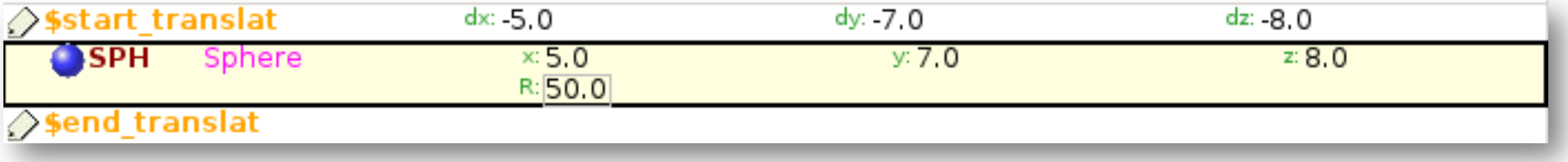


```
$start_transform    Trans: ▼
$end_transform
$start_expansion    f:
$end_expansion
$start_translat     dx:          dy:          dz:
$end_translat
```

Directives in geometry: translation

```
$start_translat
...
$end_translat
```

provides a coordinate translation for all bodies embedded within the directive



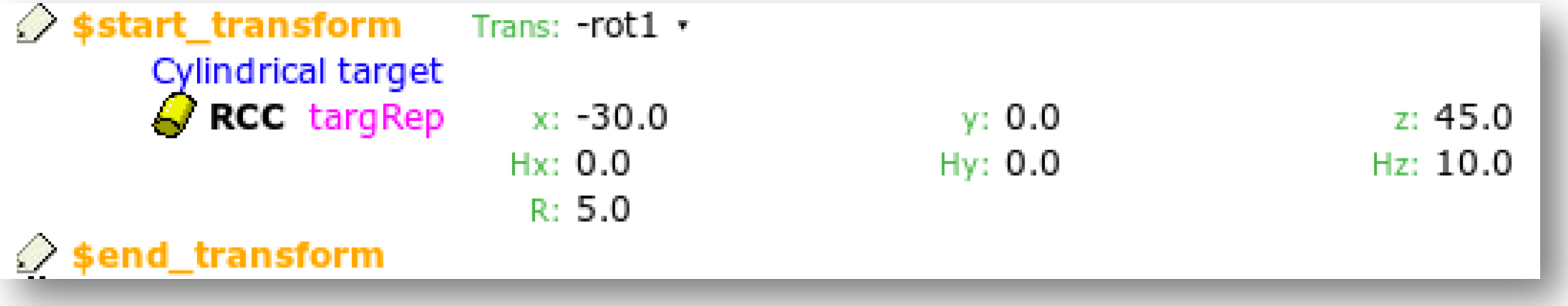
```
$start_translat          dx: -5.0          dy: -7.0          dz: -8.0
   SPH   Sphere            x: 5.0            y: 7.0            z: 8.0
                           R: 50.0
$end_translat
```

In the example, the `translat` directive transforms a sphere of radius 50 centered in (+5,+7,+8) into a sphere of radius 50 centered in (0,0,0)

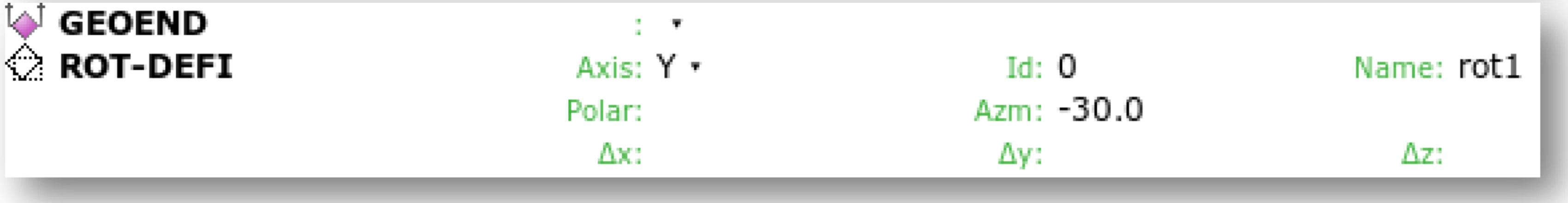
Directives in geometry: transform

```
$start_transform
...
$end_transform
```

applies a pre-defined (via **ROT-DEFI**) roto-translation to all bodies embedded within the directive



```
$start_transform      Trans: -rot1 ▾
      Cylindrical target
      RCC  targRep        x: -30.0          y: 0.0          z: 45.0
                         Hx: 0.0           Hy: 0.0         Hz: 10.0
                          R: 5.0
$end_transform
```



```
GEOEND                           : ▾
ROT-DEFI                   Axis: Y ▾           Id: 0          Name: rot1
                          Polar:              Azm: -30.0
                             Δx:               Δy:                 Δz:
```

Note that also the **inverse** transformation can be used (as in the example)

Directives in geometry: warnings

- `$Start_expansion` and `$Start_translat` are applied when reading the geometry
  → no CPU penalty (the concerned bodies are transformed once for ever at initialization)

  `$Start_transform` is applied runtime
  → some CPU penalty

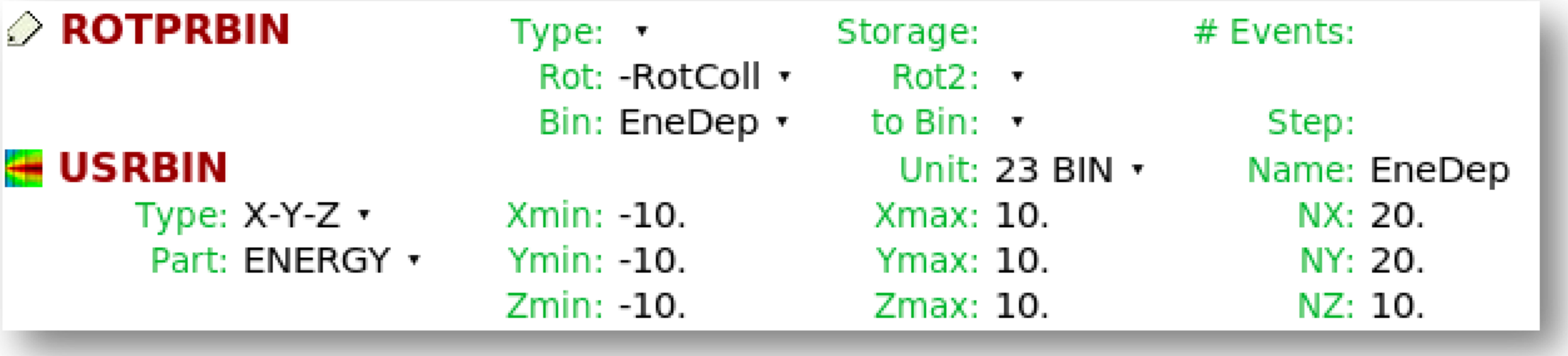
- One can nest the different directives (at most one per type!) but, no matter the input order, the adopted sequence is always the following:

```
$Start_transform
$Start_translat
$Start_expansion
...
$End_expansion
$End_translat
$End_transform
```

- Directives are not case sensitive (whereas transformation names are)

ROTPRBIN card

- It can be used to set the correspondence between roto-translation transformation and binnings (**USRBIN** and/or **EVENTBIN**)

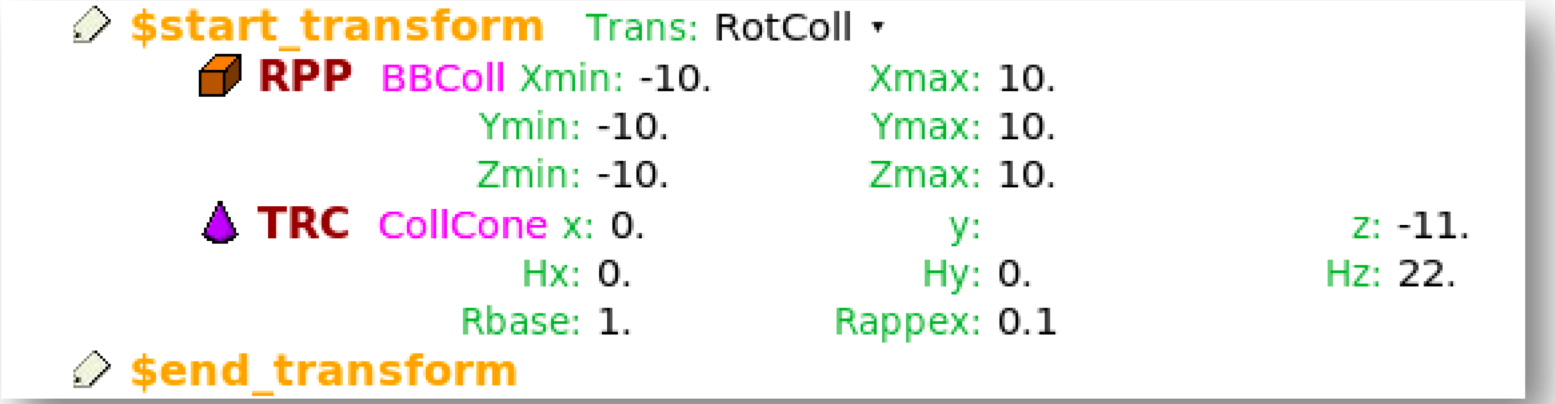


ROTPRBIN	Type: ▾	Storage:	# Events:
	Rot: -RotColl ▾	Rot2: ▾	
	Bin: EneDep ▾	to Bin: ▾	Step:
USRBIN		Unit: 23 BIN ▾	Name: EneDep
Type: X-Y-Z ▾	Xmin: -10.	Xmax: 10.	NX: 20.
Part: ENERGY ▾	Ymin: -10.	Ymax: 10.	NY: 20.
	Zmin: -10.	Zmax: 10.	NZ: 10.

Note that the roto-translation transformation shall bring the scoring point in the geometry onto the mesh of the associated binnings

Build a modular geometry with bounding boxes

- The use of bounding boxes is a recommended practice to build modular geometry models
- It consists of using a finite body (*i.e*. RCC or RPP) to encapsulate a portion of space and describe all the details of an arbitrary complex element
- `$Start_expansion` and/or `$Start_translat` are, eventually, applied to place the element in the desired position of a beam line, without changing any body parameters

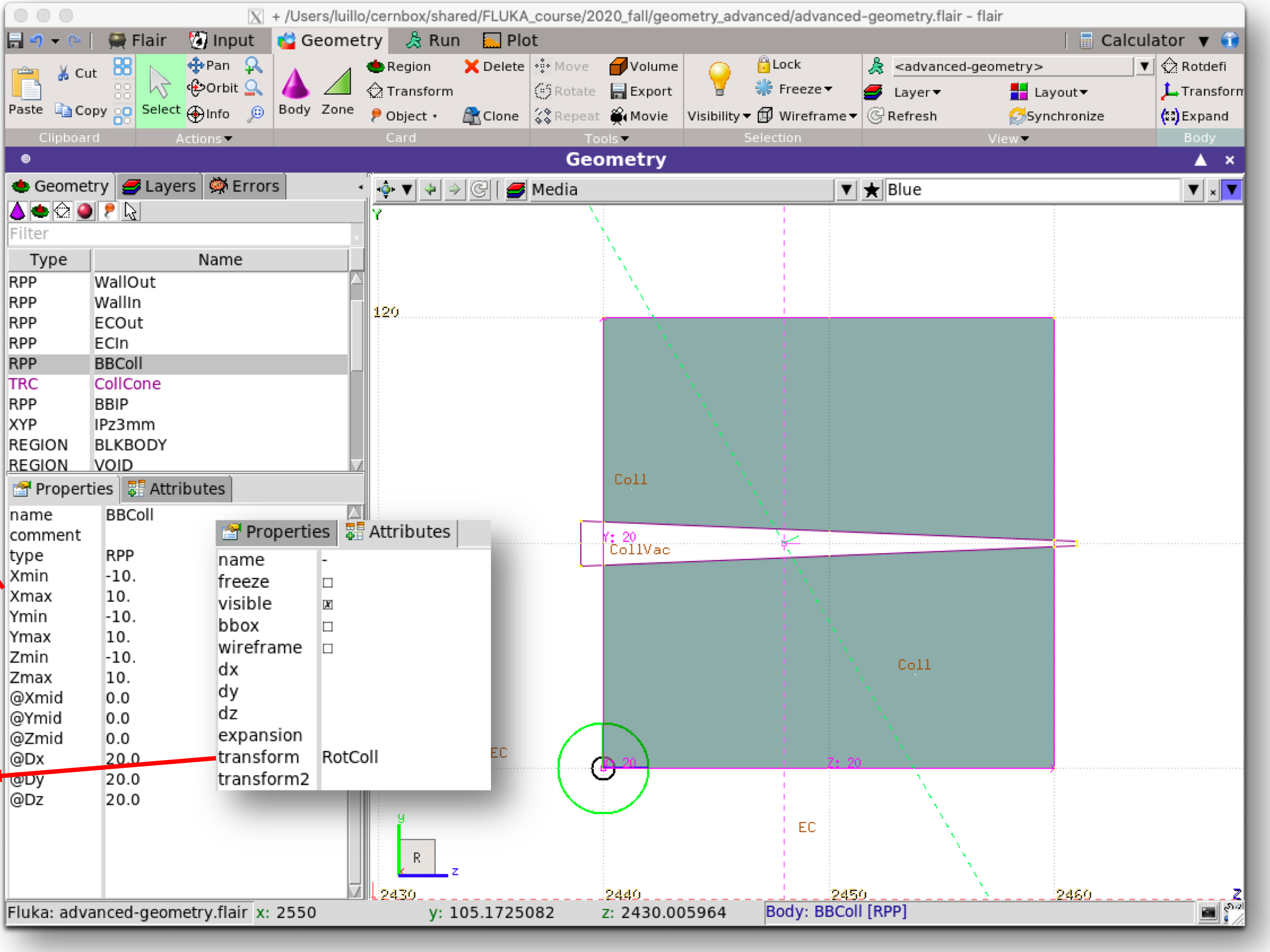


```
$start_transform   Trans: RotColl ▾
    RPP  BBColl Xmin: -10.            Xmax: 10.
                Ymin: -10.            Ymax: 10.
                Zmin: -10.            Zmax: 10.
    TRC  CollCone x: 0.                  y:                    z: -11.
                 Hx: 0.                 Hy: 0.                Hz: 22.
              Rbase: 1.             Rappex: 0.1
$end_transform
```

Example of a bounding box

1. Bounding box is defined where it is more convenient

2. A roto-translation places the element in the final position



+ /Users/luillo/cernbox/shared/FLUKA_course/2020_fall/geometry_advanced/advanced-geometry.flair - flair

Type	Name
RPP	WallOut
RPP	WallIn
RPP	ECOut
RPP	ECIn
RPP	BBColl
TRC	CollCone
RPP	BBIP
XYP	IPz3mm
REGION	BLKBODY
REGION	VOID

name	BBColl
comment	
type	RPP
Xmin	-10.
Xmax	10.
Ymin	-10.
Ymax	10.
Zmin	-10.
Zmax	10.
@Xmid	0.0
@Ymid	0.0
@Zmid	0.0
@Dx	20.0
@Dy	20.0
@Dz	20.0

name	-
freeze	□
visible	☒
bbox	□
wireframe	□
dx	
dy	
dz	
expansion	
transform	RotColl
transform2	

Fluka: advanced-geometry.flair x: 2550 y: 105.1725082 z: 2430.005964 Body: BBColl [RPP]

Summary of the relevant input cards

- **`ROT-DEFI`** to define roto-translations

- Geometry directives (inside the geometry input) to manipulate bodies
  - **`$Start_expansion $End_expansion`**
    `$Start_translat $End_translat`
    `$Start_transform $End_transform`

- **`ROTPRBIN`** to set the correspondence between a roto-translation transformation and selected **`USRBIN`** and **`EVENTBIN`** binnings

FLUKA