INTT Event Display

2023/1/24 INTT Collaboration MT

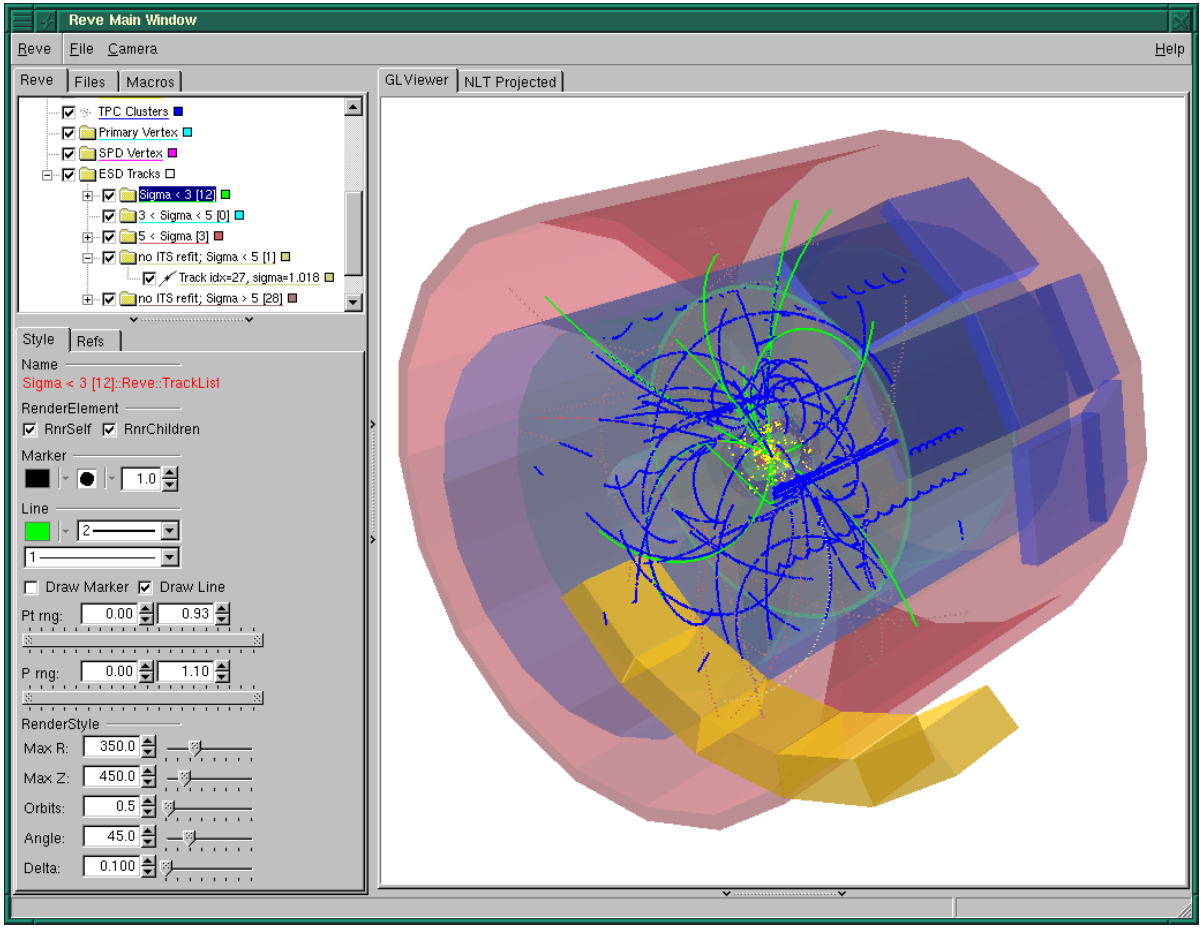
奈良女子大学 藤原愛実

概要

- Event Display とは
- Event Display 作成の流れ
- 現在の状況
- 今後の予定

Event Display

- 粒子のヒット位置を、再構成した飛跡で結び、検出器のGeometry とともに描画したもの



INTT Event Display

- INTT 上のヒット位置など、INTT に関連する情報をイベントごとに一目で確認できる
- ラダーのアライメントの確認ができる
- ROOTで実行できる

Event Display 作成の流れ

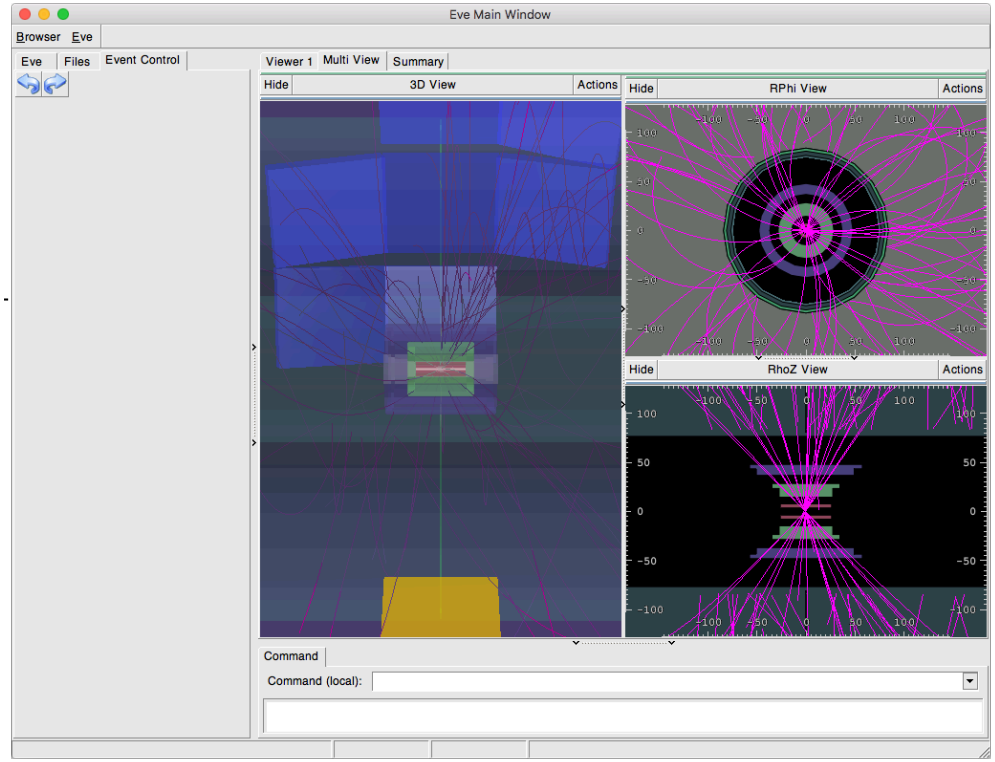
1. シミュレーション ✓

2. ヒット位置に点を打つ ✓

3. INTTのGeometry を用意する ✓

---

4. ヒット位置を線で結ぶ

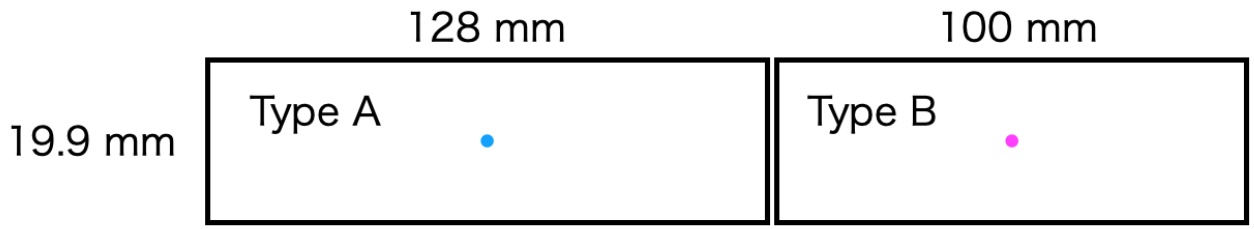


Viewer機能

- r-ϕ、ρ-z プロジェクションの実装

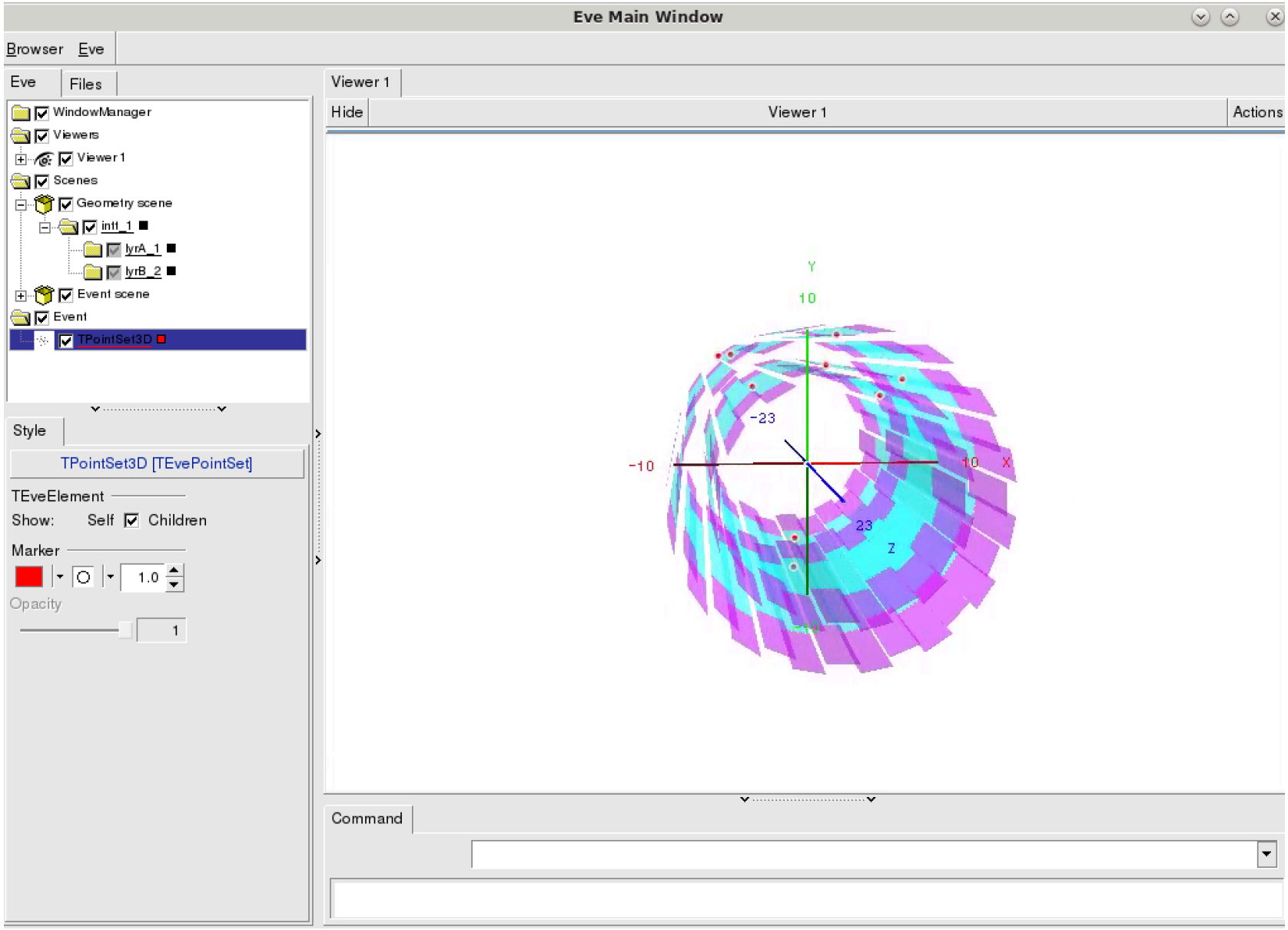
Geometry作成

1. Type Aセンサーの中心とType B センサーの中心座標を取得
2. Type A センサーとType B センサーの大きさに合わせた箱を作成
3. 座標と角度を調節して、箱を複製
4. root ファイルに書き出し、Event Display を描くマクロで読み込みヒット位置と共に出力



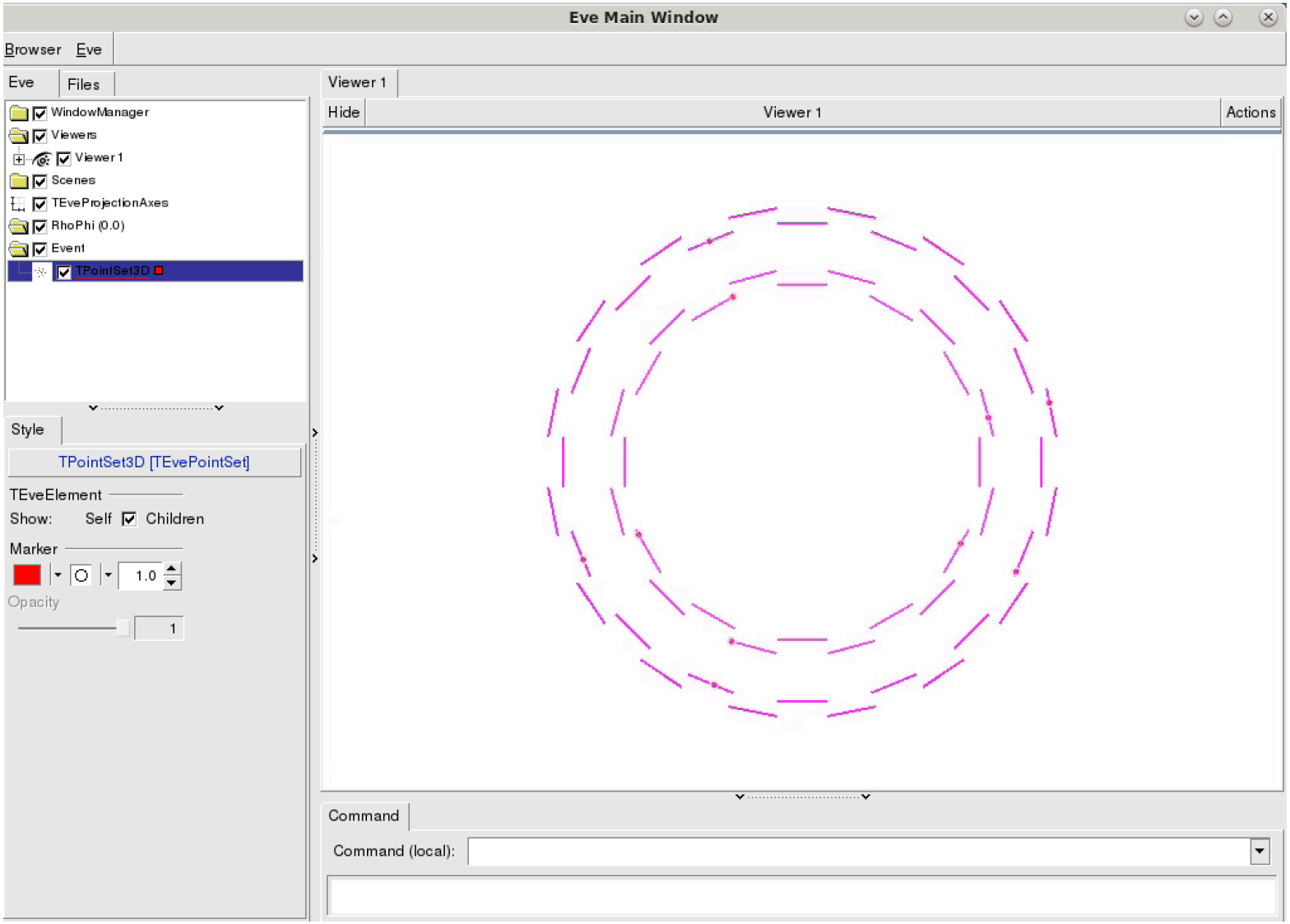
現在の進捗

INTT Geometry



現在の進捗

r-ϕ Projection



今後の予定

- 飛跡を書いてヒット位置が同一線上に乗るかを確認する
- r-ϕ、ρ-z プロジェクションの実装

Back Up

find_segment_center

Code

```
uint8_t type;
uint8_t ladder_index_max;
int time_bucket = 0;

for (uint8_t lyr = 3; lyr < 7; lyr++)
{

  // number of ladders
  switch (lyr)
  {
  case 3:
    ladder_index_max = 12;
    cout << "ladder_index_max =" << ladder_index_max << endl;
    break;
  case 4:
    ladder_index_max = 12;
    cout << "ladder_index_max =" << ladder_index_max << endl;
    break;
  case 5:
    ladder_index_max = 16;
    cout << "ladder_index_max =" << ladder_index_max << endl;
    break;
  case 6:
    ladder_index_max = 16;
    cout << "ladder_index_max =" << ladder_index_max << endl;
    break;
  }
```

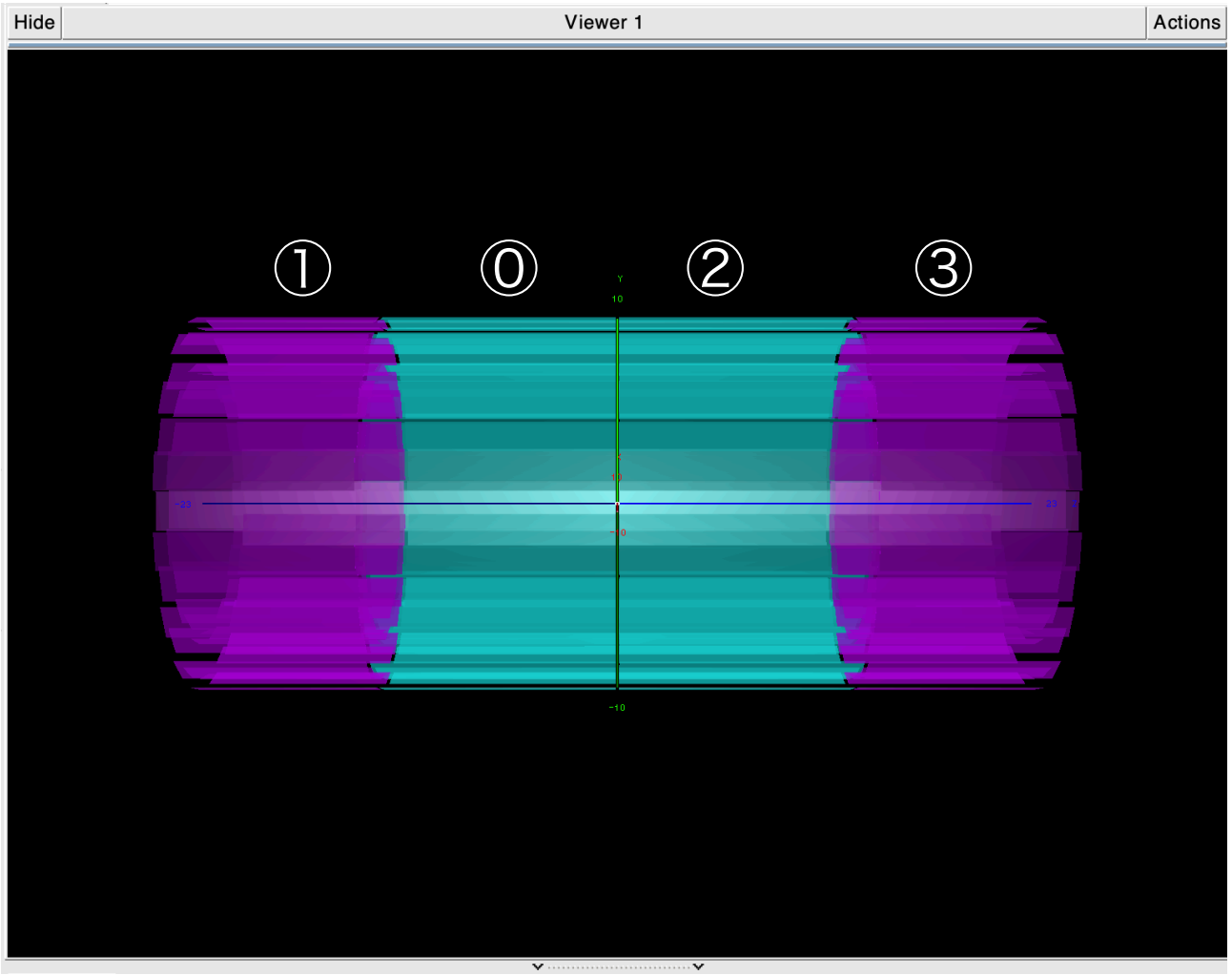
```
// typeA position
type = 0;
for (uint8_t ladder_index = 0; ladder_index < ladder_index_max; ladder_index
++)
{
  auto genhitkeyintt = InttDefs::genHitSetKey(lyr, type, ladder_index,
  time_bucket);
  auto surfintt = m_tGeometry->maps().getSiliconSurface(genhitkeyintt);
  double ladderLocation[3] = {0., 0., 0.};
  m_geom->find_segment_center(surfintt, m_tGeometry, ladderLocation);
  // cout << "ladderLocationA=(" << ladderLocation[0] << "," <<
  ladderLocation[1] << "," << ladderLocation[2] << ")" << endl;

  ladderALocationX.push_back(ladderLocation[0]);
  ladderALocationY.push_back(ladderLocation[1]);
  ladderALocationZ.push_back(ladderLocation[2]);
  // cout<<"(lyr,ladder_index)=("<<unsigned(lyr)<<","<<unsigned
  (ladder_index)<<")"<<endl;
}
```

Hitsetkey

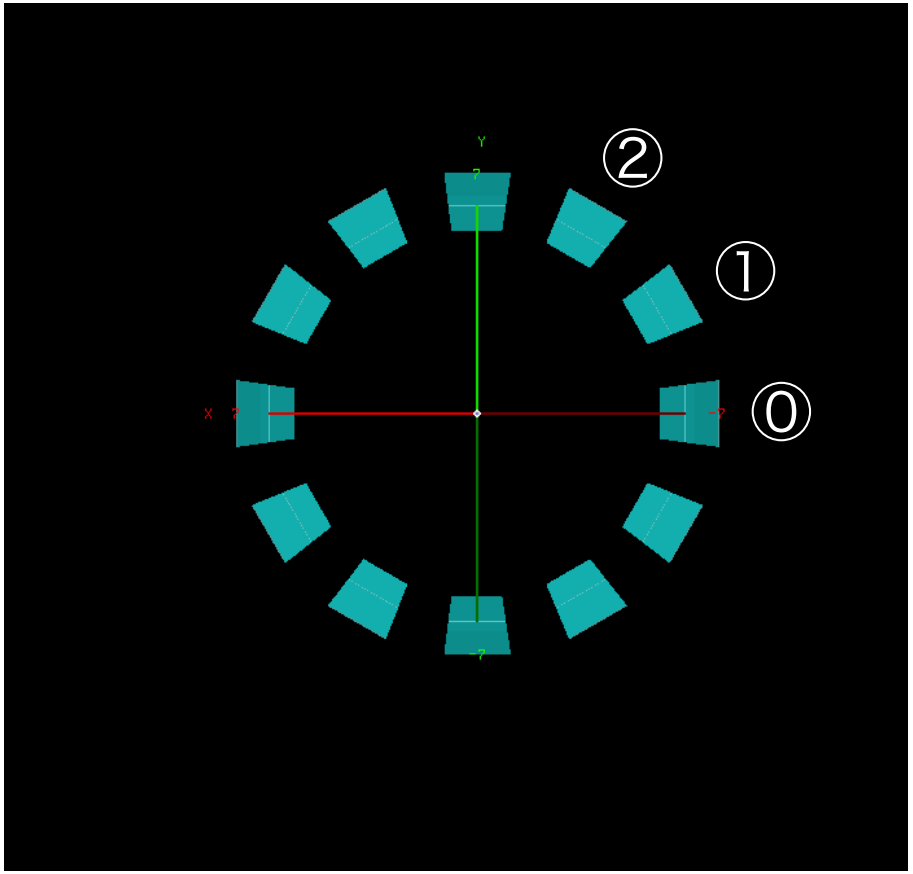
- Hitsetkey ：32bitの符号なし整数
- 最初の16bit は固定
- 残りの16bit を検出器の種類、レイヤー、ラダーId、time bucket の表現に割り当てている

https://github.com/sPHENIX-Collaboration/coresoftware/blob/812908c46064f7ec360e31f23708b3cbe43d5d19/offline/packages/trackbase/InttDefs.h

ladder_z_index



ladder_phi_index



TGeo

- ROOT で実装されているGeometry 作成用クラス
- 箱や円柱などの単純な形状のGeometry を作成したり、作成したGeometry を移動、回転、複製する関数などが実装されている
- INTTのGeometry を作成するために使用

TEve

- ROOT のEvent Display 作成用クラス
- 飛跡を再構成したり、3Dグラフを作成したり、TGeo クラスで作ったGeometry のファイルを読み込んで表示する関数が実装されている

Geometry作成

Code

```
//makeinttgeom
    TGeoManager * geom = new TGeoManager("geom", "geom");
    TGeoVolume * inttgeom = geom->MakeBox("intt", NULL, 20, 20, 20);
    geom->SetTopVolume(inttgeom);
    inttgeom->SetVisibility(kFALSE);

    // segment
    double segment_thickness = 0.32; // mm
    double segment_widthY = 19.9;    // mm
    double segmentA_widthZ = 128.0;  // mm
    double segmentB_widthZ = 100.0;  //mm
    TGeoVolume *segmentA = geom->MakeBox("BOX", NULL, segment_thickness /
    10/2, segment_widthY / 10/2, segmentA_widthZ / 10/2);
    TGeoVolume *segmentB = geom->MakeBox("BOX", NULL, segment_thickness /
    10/2, segment_widthY / 10/2, segmentB_widthZ / 10/2);
    segmentA->SetLineColor(kCyan);
    segmentB->SetLineColor(kViolet);

    TGeoVolume * lyrA = geom->MakeBox("lyrA",NULL,10,10,10);
    lyrA->SetVisibility(kFALSE);
    inttgeom->AddNode(lyrA,1,0);

    TGeoVolume * lyrB = geom->MakeBox("lyrB",NULL,10,10,10);
    lyrB->SetVisibility(kFALSE);
    inttgeom->AddNode(lyrB,2,0);
```

Type A,TypeB センサー部分と同じサイズの箱を用意する

Geometry作成

Code

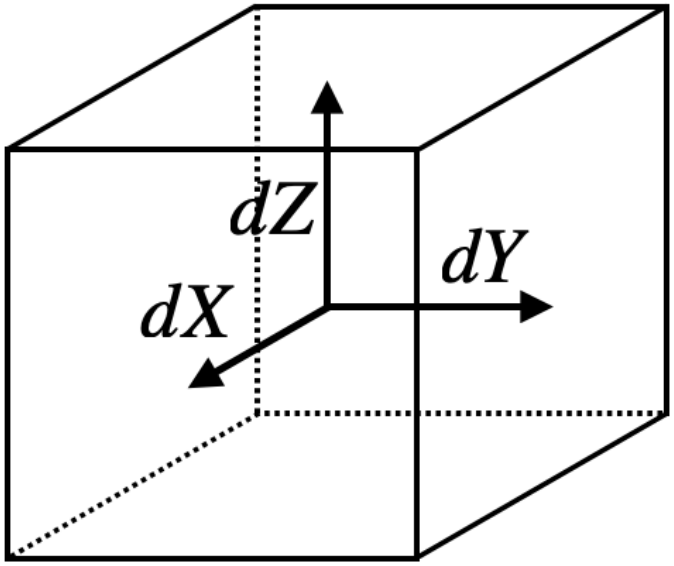
```
//definition
int sum_segmentA = segmentALocationX.size();
int sum_segmentB = segmentBLocationX.size();

TGeoRotation * rotA[sum_segmentA];
TGeoCombiTrans * combitranslyrA[sum_segmentA];

TGeoRotation * rotB[sum_segmentB];
TGeoCombiTrans * combitranslyrB[sum_segmentB];

double phi;

//make segments
for(int i=0;i<sum_segmentA;i++){
  phi = atan2(segmentALocationY[i], segmentALocationX[i]) * 180 / M_PI;
  rotA[i] = new TGeoRotation(Form("rotA[%d]",i), phi, 0., 0.);
  combitranslyrA[i] = new TGeoCombiTrans(segmentALocationX[i],
  segmentALocationY[i], segmentALocationZ[i], rotA[i]);
  lyrA->AddNode(segmentA, i, combitranslyrA[i]);
}

for(int i=0;i<sum_segmentB;i++){
  phi = atan2(segmentBLocationY[i], segmentBLocationX[i]) * 180 / M_PI;
  rotB[i] = new TGeoRotation(Form("rotB[%d]",i), phi, 0., 0.);
  combitranslyrB[i] = new TGeoCombiTrans(segmentBLocationX[i],
  segmentBLocationY[i], segmentBLocationZ[i], rotB[i]);
  lyrB->AddNode(segmentB, i, combitranslyrB[i]);
}
```

回転角度を計算

角度と座標を設定

MakeBox

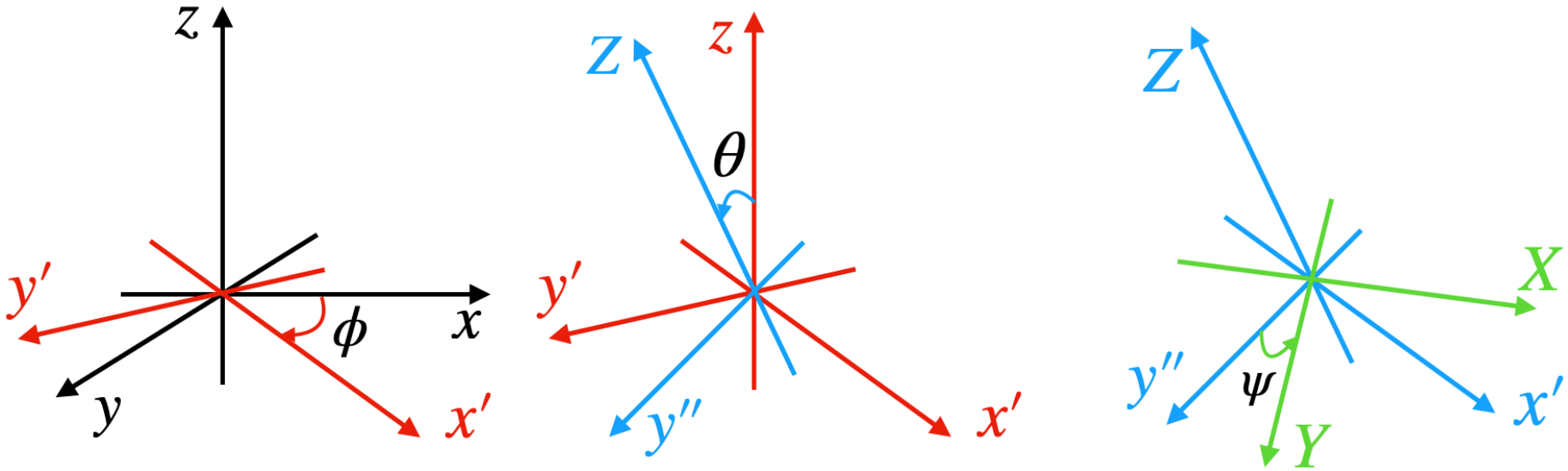
- dX,dY,dZで指定したサイズの箱を作れる
- 単位はcm



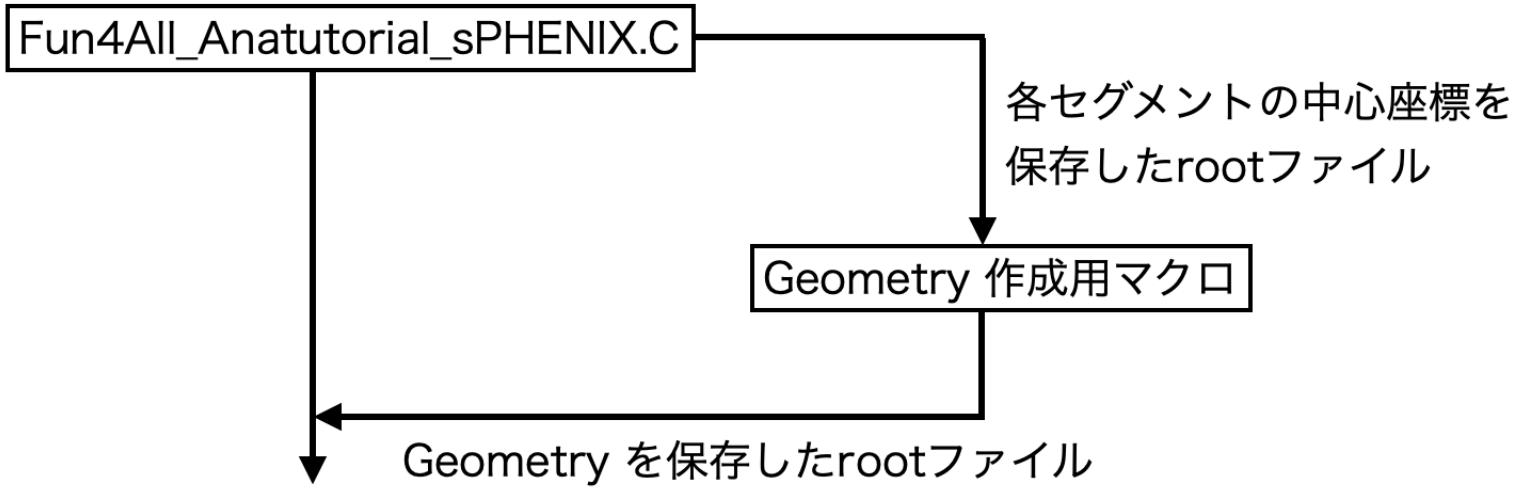
オイラー角

- 最初にz軸を中心に角度ϕ 回転、2番目にx軸を中心に角度θ 回転、最後に回転後のz軸を中心に角度ψ 回転させる。

- TGeoRotation では、単位は度



Geometry 作成の流れ



Geometry ファイルを読み込んで、
Event Display用Viewerを出力