# MDISim, update

**MDISim : H.B. + Manuela Boscolo**

Tools for Flexible Optimisation of IR Designs with Application to FCC, IPAC 2015 **tupty031**

**Flexible ROOT based machine detector interface toolbox between**

**accelerator described by MAD-X sequence**

**particle interactions in the IR/detector regions using programs like GEANT4** ref1, ref2

**Guideline : use as much as possible existing, openly available and well supported codes**

**improve them if needed**

examples : automatic MAD-X sequence editing MAKETHIN

GEANT4 G4SynchrotronRadiation.cc, TestEm16

## Technical presentation, concepts

- **geometry, coordinate transformations : Euclidian EU ⟷ CS Courant-Snyder**
- **bending magnets, curved/straight ( BOX, TUBE, TORUS ), orbit correctors**
- **quadrupoles**
- **Generation of geometries, as suitable for input particles shower simulations - GEANT4**

# Basics : Coordinate systems, naming, ref. links

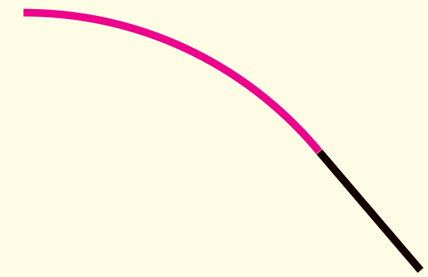
Two different coordinate systems --- related to general geometry as described in Wikipedia :

- Three-dimensional Euclidian space
  Cartesian coordinate system in 3-dim Euclidean_geometry
  also known as global_system in MAD-X
  Natural choice for tracking around detectors, neutrals, GEANT4, (off-momentum)

- local accelerator coordinates, described in the MAD8 physics guide
  also valid as basis for current MAD-X
  Special case of Curvilinear coordinates related to Frenet-Serret formulas
  with 3 dimensions (x, y, s) here referred to as Courant-Snyder coordinates
  along the design path of the accelerator
  Used everywhere on the accelerator side : twiss, multi-turn tracking, long term stability
  exception : the MAD-X survey module, which generate global cartesian coordinates
  at element boundaries as required to position magnets in real space ( survey in tunnel )

Essential basic ingredient MDISim : EU <--> CS

# Transformation CS <--> EU



**Design path : generated by the connection of the elements in the sequence**

**Initial position and rotation + two types of elements**

**arc segment : bending magnets -- angle α and bending radius ϱ**

**straight : all other elements, including quadrupoles, correctors**

**Local Courant Snyder v_cs = (*x*, *y*, *s*) *s* along design path, *x*, *y* perpendicular to path**

**Global Euclidian v_eu = (x, y, z) from single origin, for MDI typically the IR**

**Transformation 3 dim shift vector V and rotation matrix W for every volume**

**Orthogonal 3x3 matrix (SO3), defined by 3 angles θ, ϕ, ψ, using the MAD convention**

**related to https://en.wikipedia.org/wiki/Rotation_formalisms_in_three_dimensions**

$$\mathrm{ThetaCS}[\theta] = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{pmatrix} = R_y(-\theta) \qquad \mathrm{PsiCS}[\psi] = \begin{pmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix} = R_z(-\psi)$$

$$\mathrm{PhiCS}[\phi] = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{pmatrix} = R_x(\phi)$$

**general 3-dim rotation**

$$\mathrm{Wcs}[\theta, \phi, \psi] = R_y[-\theta]\, R_x[\phi]\, R_z[\psi] = \mathrm{ThetaCS}[\theta]\ \mathrm{PhiCS}[\phi]\ \mathrm{PsiCS}[\psi]$$

# MDISim transformation CS <--> EU

**Set up transformation for all volumes, just needs lengths L and angles (0 if straight)**

**of all consecutive elements in the sequence**

**read from tfs table generated with twiss** **bending radius $\varrho$ = L / Angle**

```
CourantSnyderSystem(const valarray<double>& L,const valarray<double>& Angle,const Vec3&,
                     V0=Vec3(0,0,0),const Mat3x3& W0=M3x3::Identity,unsigned int verbose=0);
// set up internally all element eu (end) positions V and rotation W
if(fabs(Angle[i])<eps) // straight element --  all except bends
{
  R=Vec3(0,0,L[i]); // shift by length
  W[i]=W[i-1];      // no change in rotation, same matrix next step for next element
}
else // bend,  arc segment
{
  double rho=L[i]/Angle[i];
  R=Vec3( rho*cos(Angle[i])-rho, 0, rho*sin(Angle[i]) ); // displacement in element
  S=Rot_y(Angle[i]);
  W[i]=W[i-1]*S; // rotation, used next step for next element;    same as  WCS_mat3(theta,phi,psi);
 }
 V[i] = W[i-1] R[i] + V[i-1]; // eu (end) position of volume
```

**CS to EU**

```
SurVey ToEuclidian(const Vec3& V_cs); // Survey is eu vector and angles (x,y,z , θ,φ,ψ)
```

**EU to CS** - as used for example to determine the GEANT4 tracking precisions

```
unsigned int FindVolume(const Vec3& V_eu);     // find Volume   (not trivial)
Vec3 ToCS(unsigned int ivol,const Vec3& V_eu); // EU -> VS
```

# Main transformation code

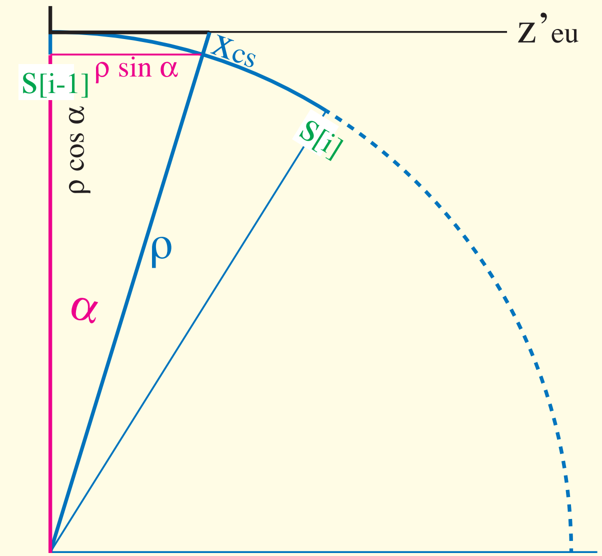
## CS to EU

```
Vec3 R;
if(fabs(sl)<eps || fabs(Angle[i])<eps) // 0-length or straight element
{
  R=V_cs-Vec3(0,0,spos[i-1]); // straight dist from start
}
else  // in bend
{
  double rho=L[i]/Angle[i];
  alfa=(V_cs.r[2]-spos[i-1])/rho; // part of bend angle to current position
  R=Vec3( (V_cs.r[0]+rho)*cos(alfa)-rho, V_cs.r[1], (V_cs.r[0]+rho)*sin(alfa)); // point on circular path
}
Vec3 V_eu=W[i-1]*R+V[i-1];
```

## Eu to CS

Finding the volume ivol may require several iterations : find what is close, transform to volume and check if really inside; then

```
Mat3x3 Winv=Transpose(W[ivol-1]);
Vec3   R=Winv*(V_eu-V[ivol-1]);
Vec3 V_cs;
if(L[ivol]<eps || fabs(Angle[ivol])<eps) // straight element
{
  V_cs=R+Vec3(0,0,spos[ivol-1]);
}
else // curved,  in bend
{
  double rho=L[ivol]/Angle[ivol];
  Vec3 R_rho=R+Vec3(rho,0,0);
  double alfa=atan(R_rho.r[2]/R_rho.r[0]);
  double rhoprime=R_rho.r[0]/cos(alfa); // rho+x_cs
  double x_cs=rhoprime-rho;
  double s=spos[ivol-1]+rho*alfa;
  V_cs=Vec3(x_cs, R.r[1], s);
 }
```

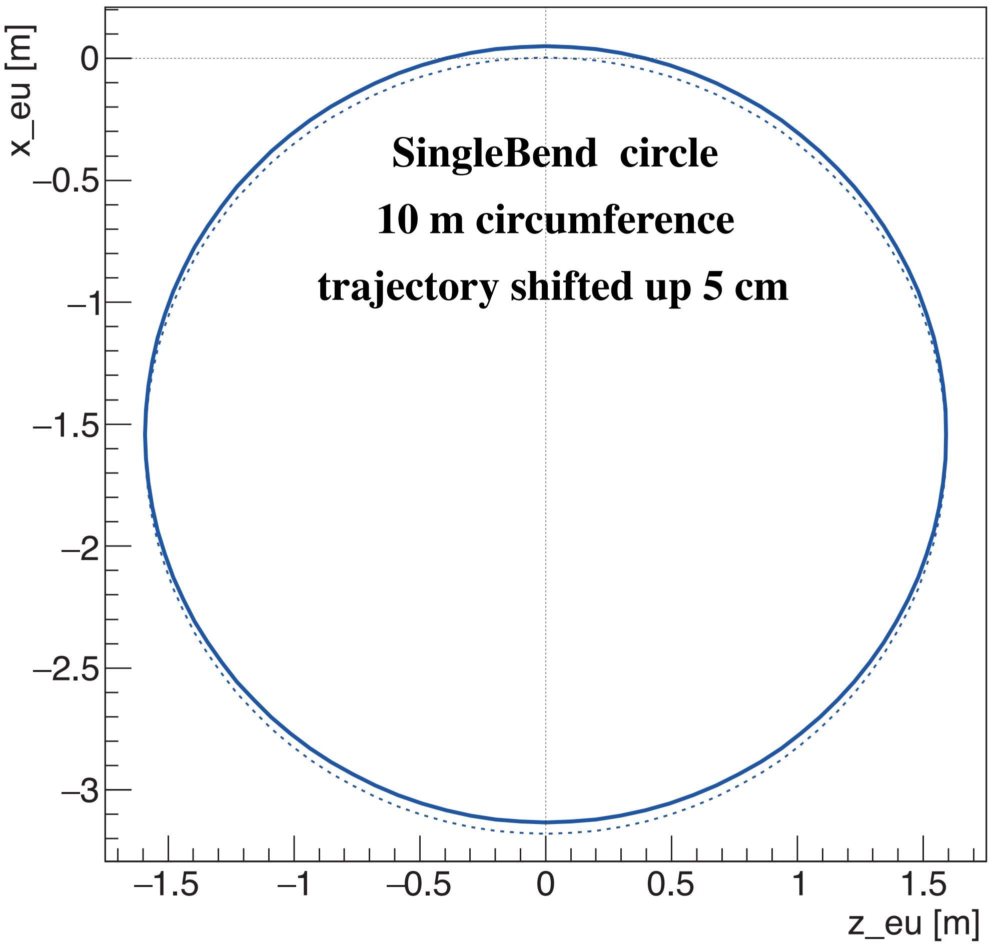


# Testing

**Using dedicated MAD-X model sequences, w/o thick slicing SingleBend, SingleQuad..**

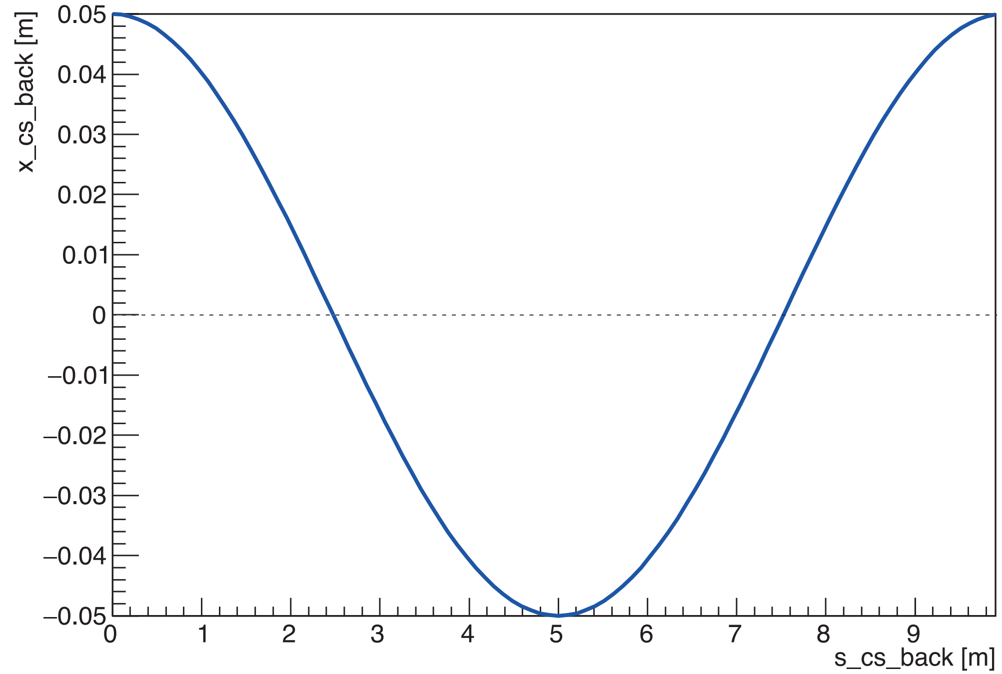
going back and forward EU, CS; varying initial position, rotation ...

here on simple example :




SingeBend.madx, geometry using L, angles from twiss table
trajectory start shifted up by 5 cm
translated to v_eu using MDISim

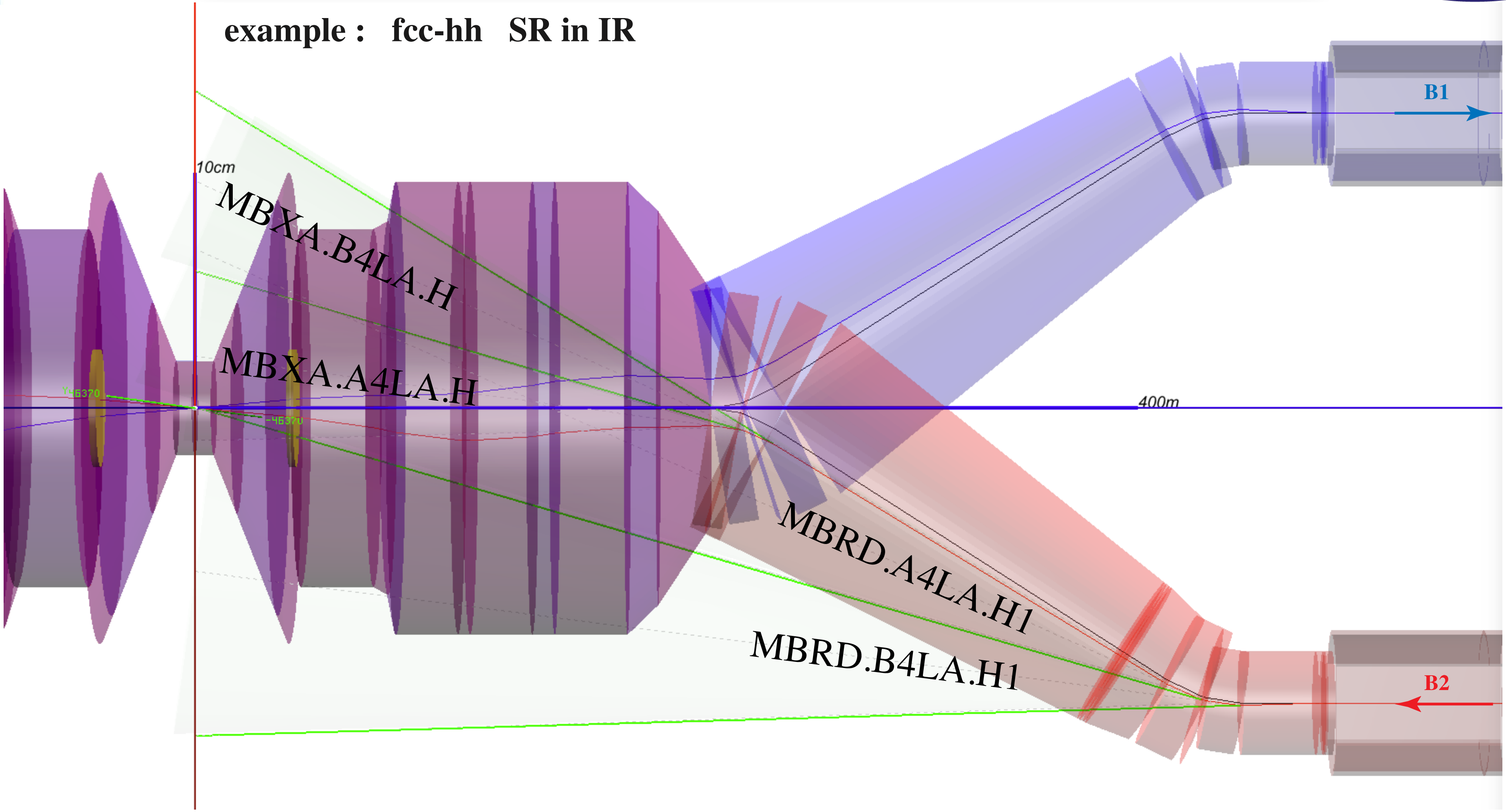
**CS coordinates show as expected the weak (geometrical) "focusing"**




Check backwards, v_eu --> v_cs

# Improving Geometry and Tracking details

example : fcc-hh SR in IR




strong scaling

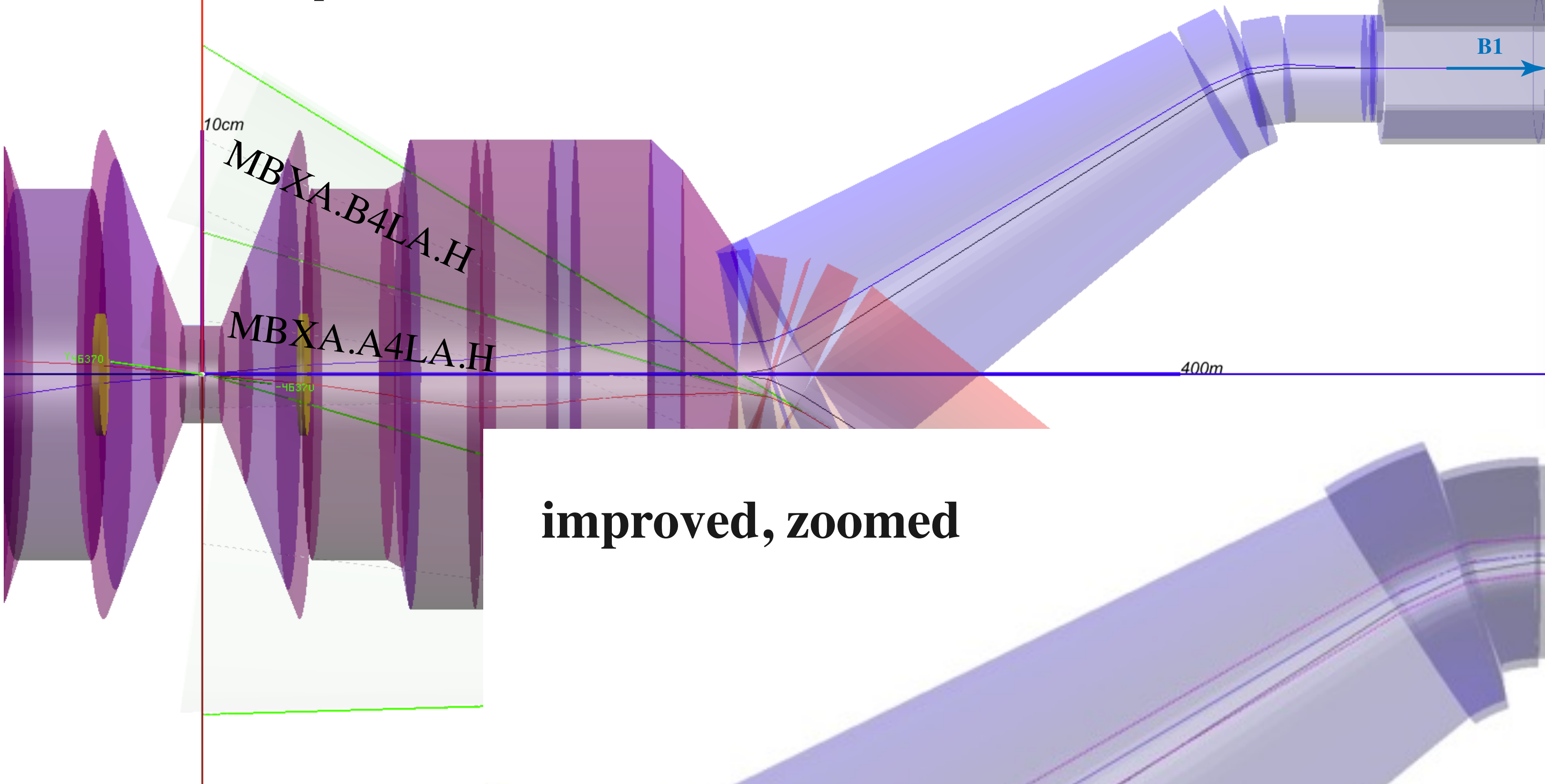
here x, y × 1000

essential for FCC display

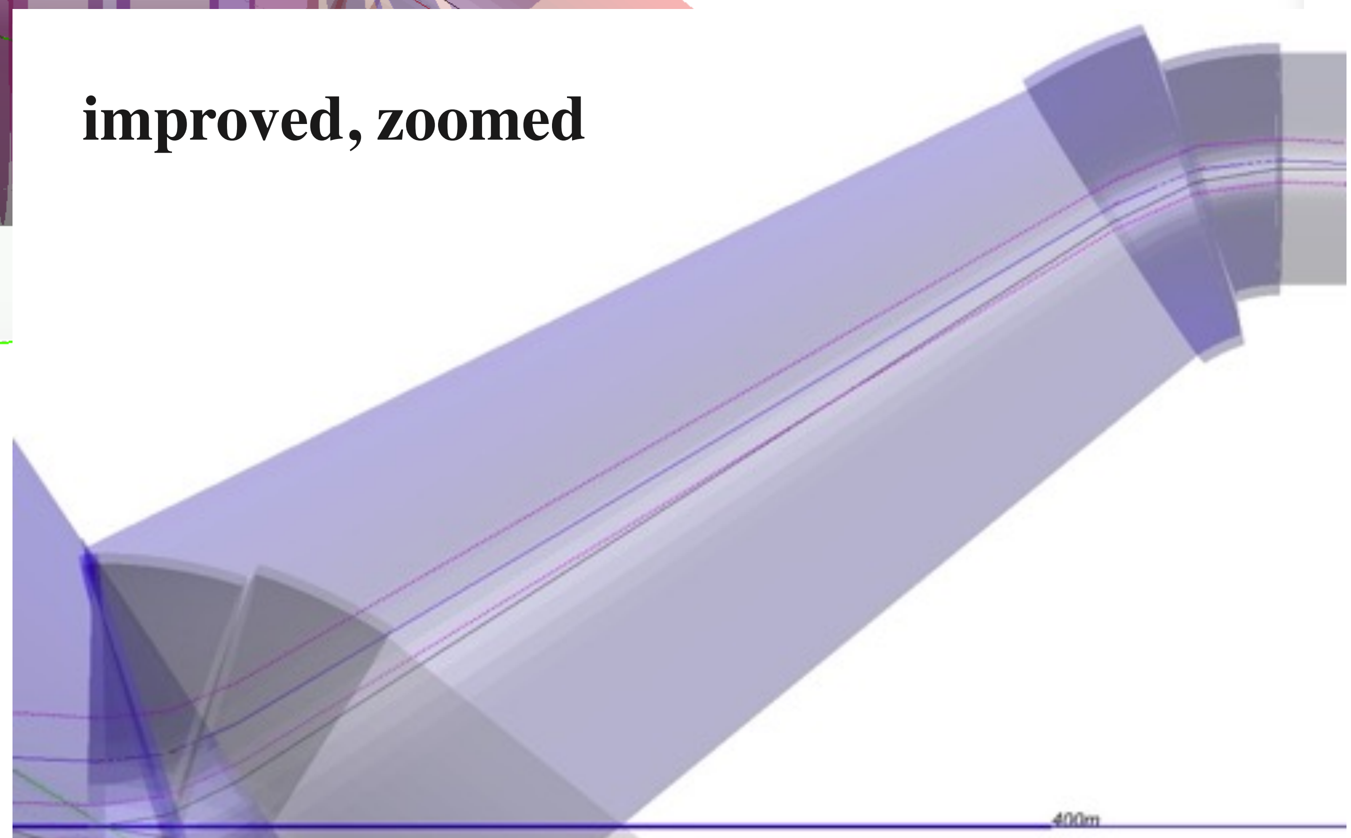
# Improving Geometry and Tracking details



**example : fcc-hh SR in IR**




**improved, zoomed**



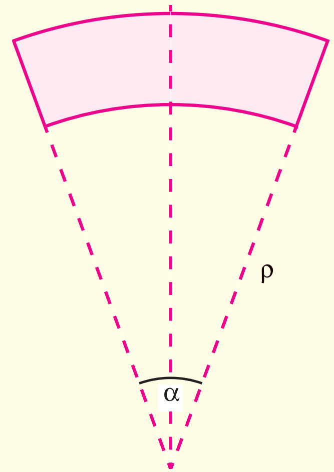

**strong scaling**

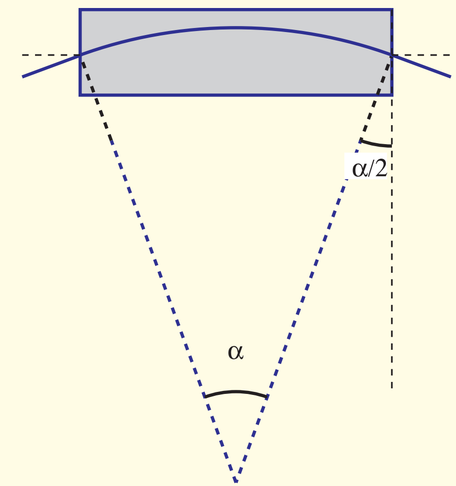
**here x, y × 1000**

**essential for FCC display**

# Bending magnets

**Different types on MAD-X level**

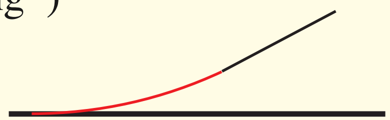



1. **SBEND**

2. **RBEND**

3. **Orbit corrector** ( MAD-X type called HKICKER, VKICKER misleading )



**Same for Synchrotron Radiation : particle vector and field vector B** ( $E_c \propto \gamma^2 B\perp$ )

**Differences on geometry level :**

**BENDs define design path = centre of beam pipe**

turning off a bend on MAD-X level not losing the beam -- changing the accelerator geometry

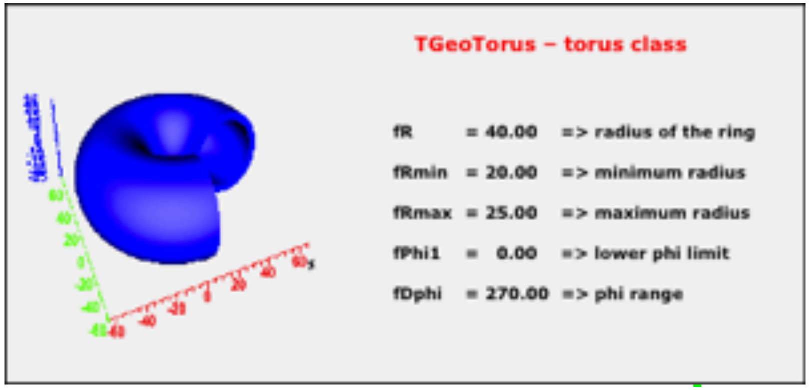
**RBEND - Tube, Box geometry, for optics = dipdege + sbend + dipedge** (translated by makethin)

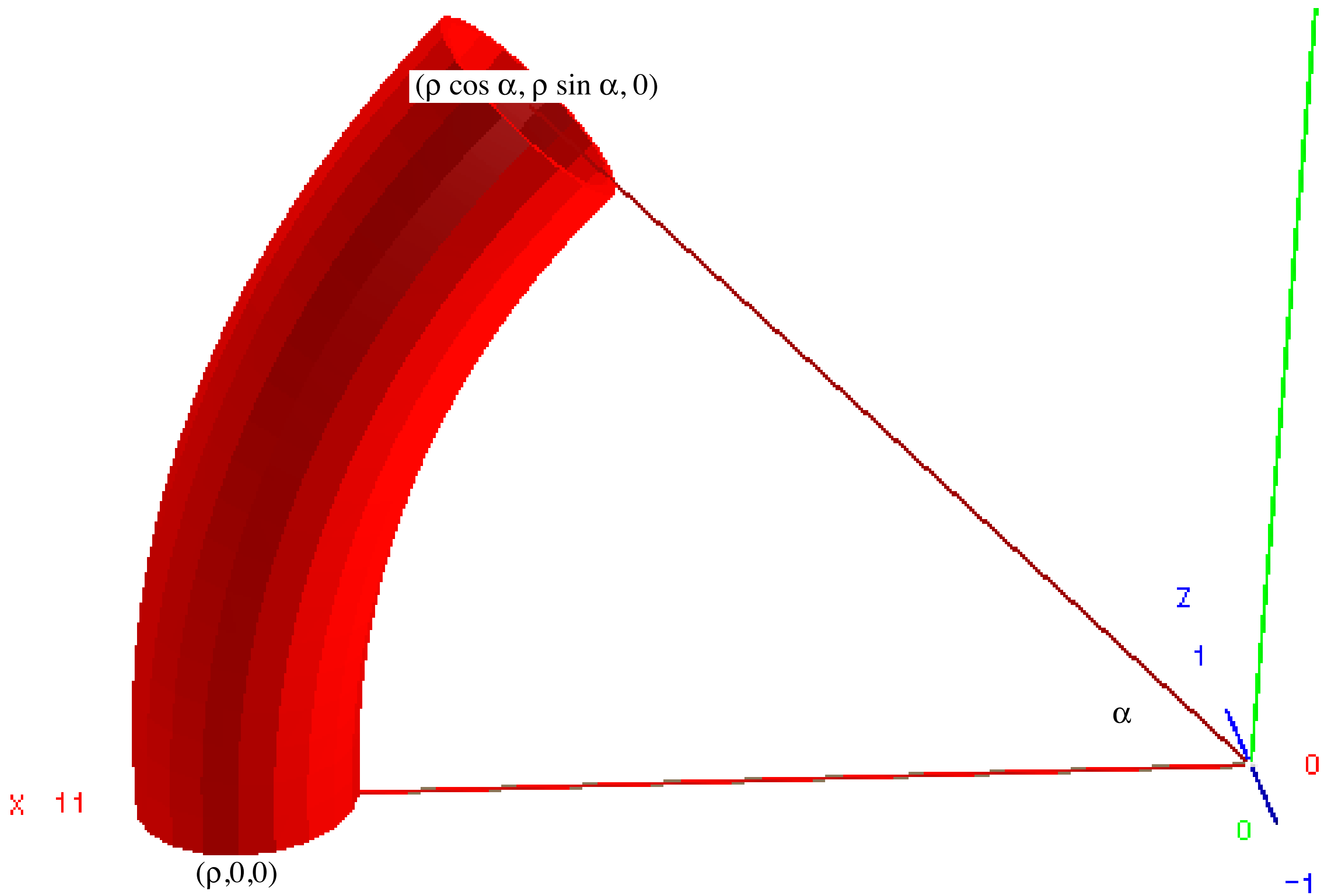
**SBEND - Torus segment**

**Corrector changes the particle trajectory --- no change in geometry**

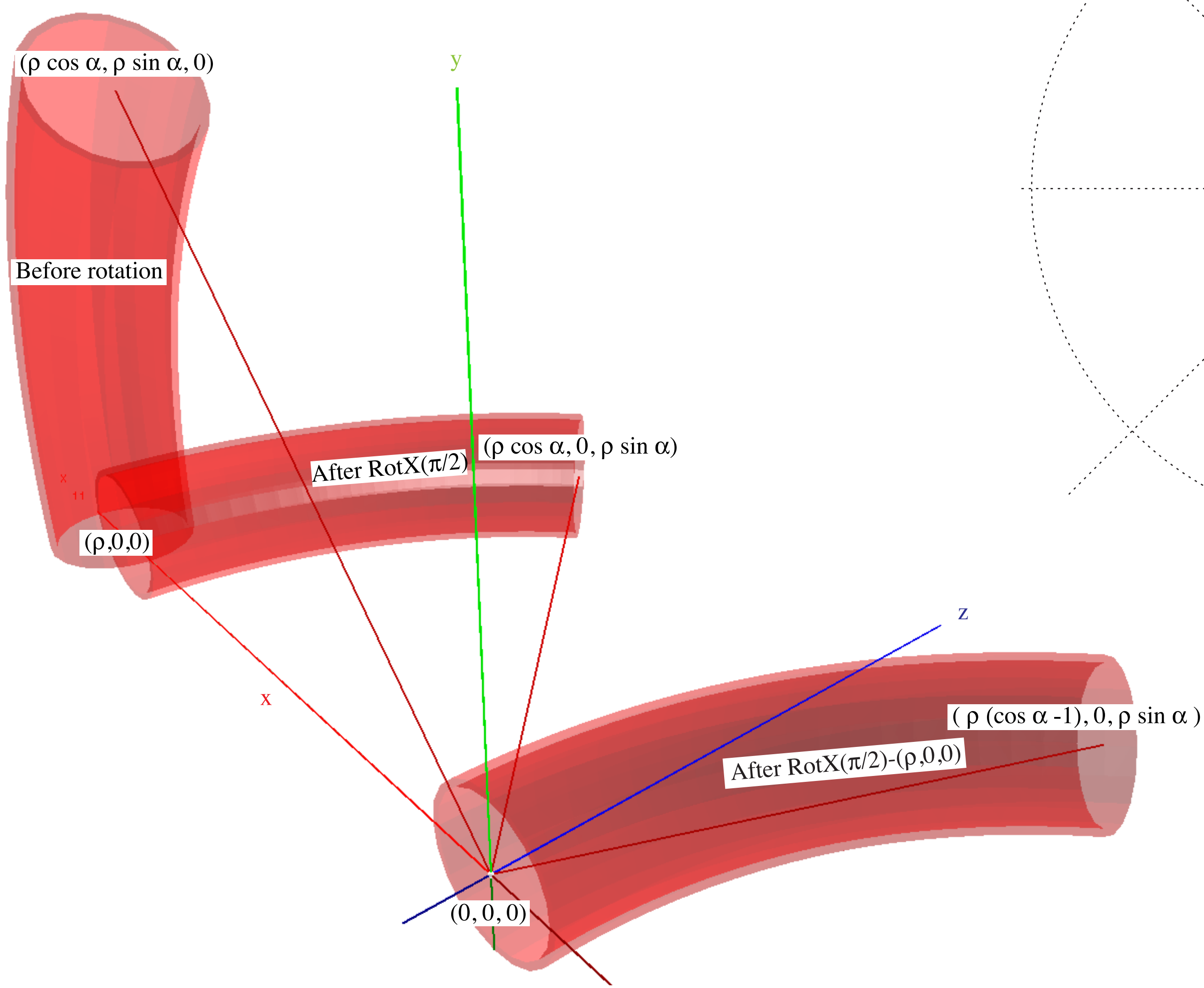
# Bending magnets, geometry generation

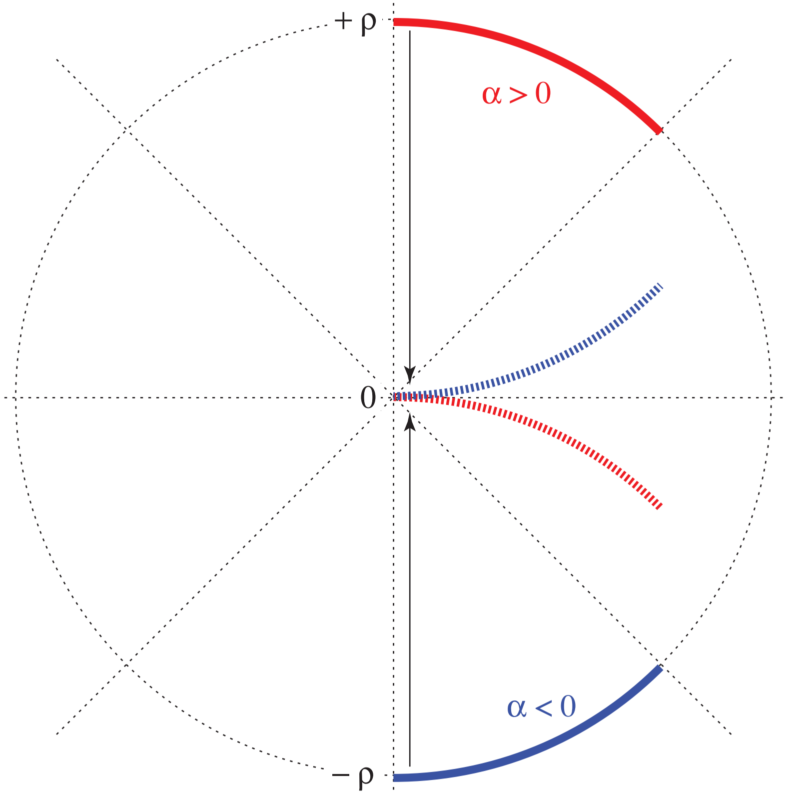
**Torus, defined in ROOT and GEANT4**



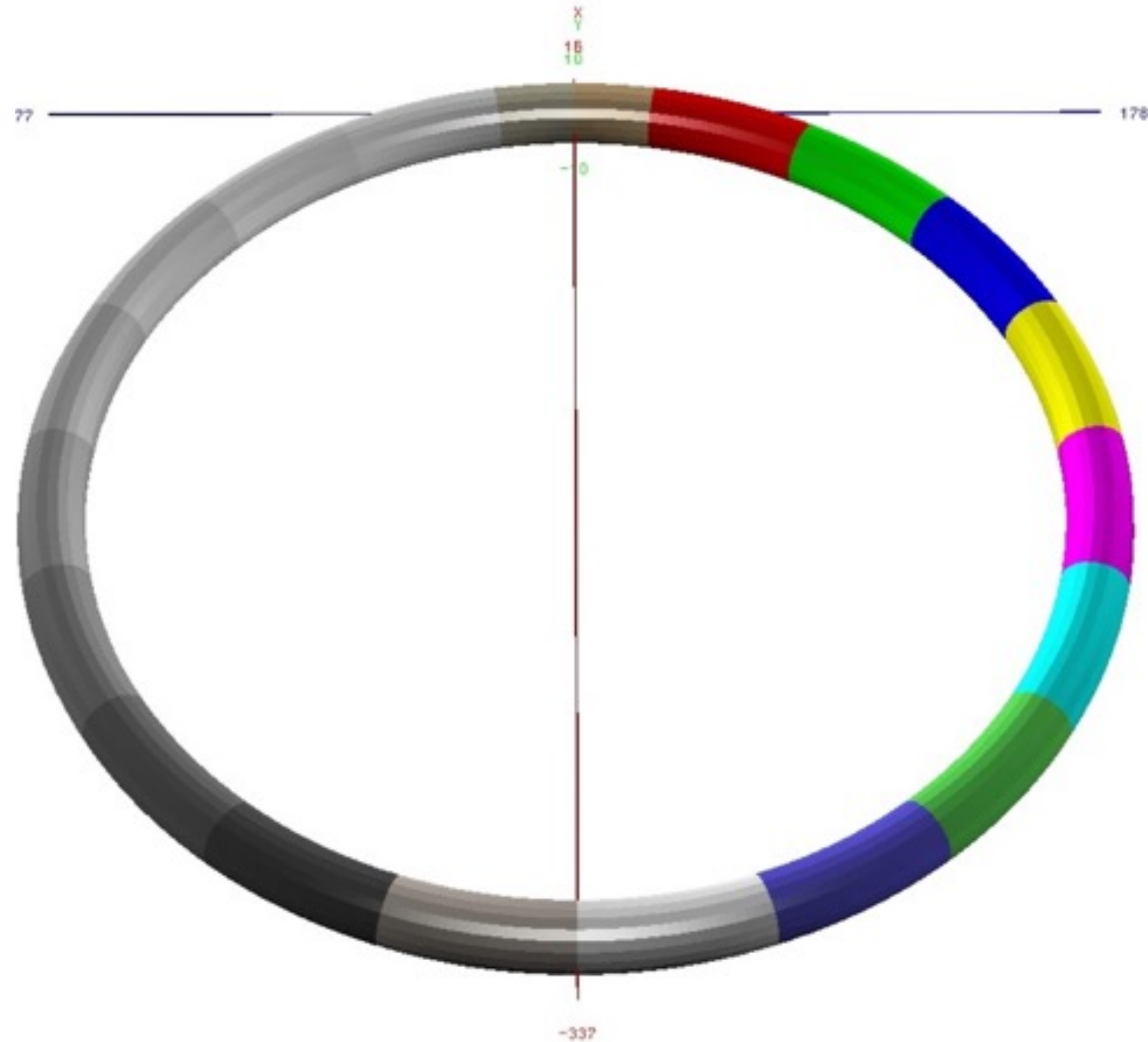
# Torus generation / positioning




**Principle of constructing a Torus for negative angle by starting a π and shifting by +ρ rather than −ρ**

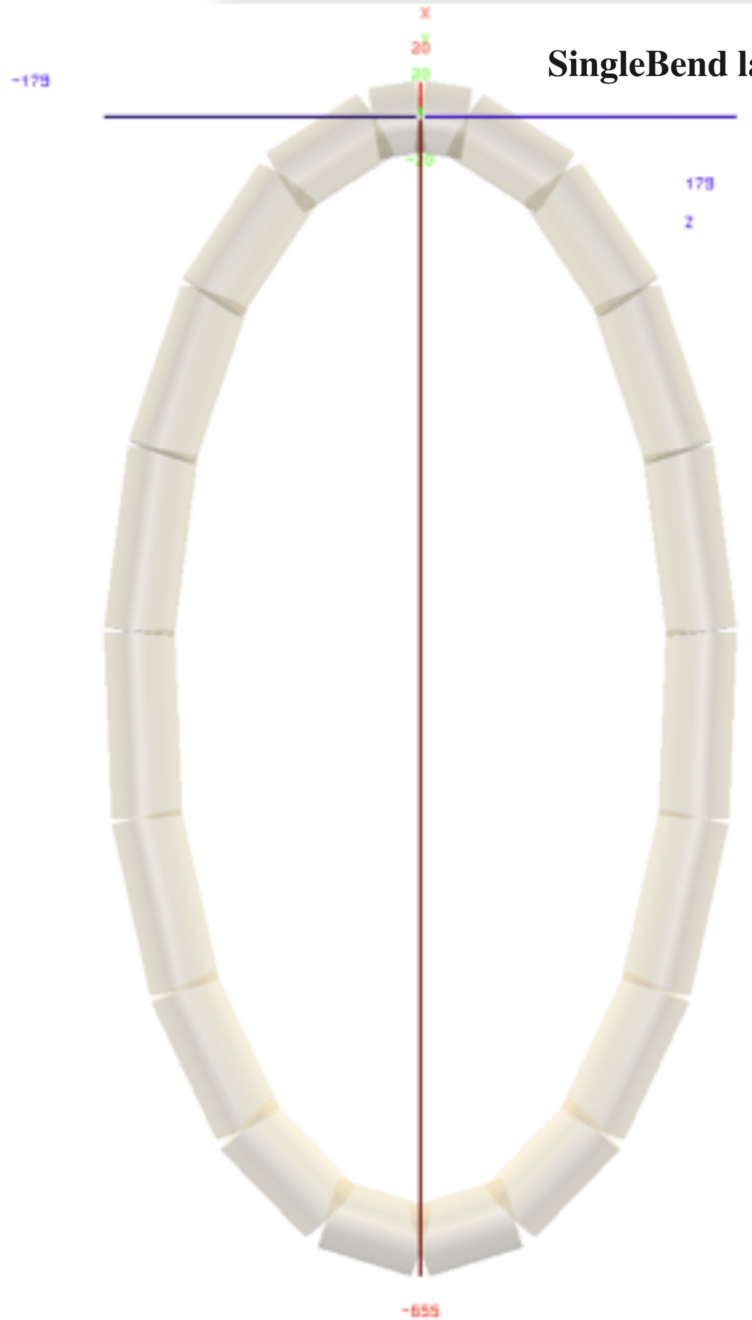
# Tested / illustrated for simple ring

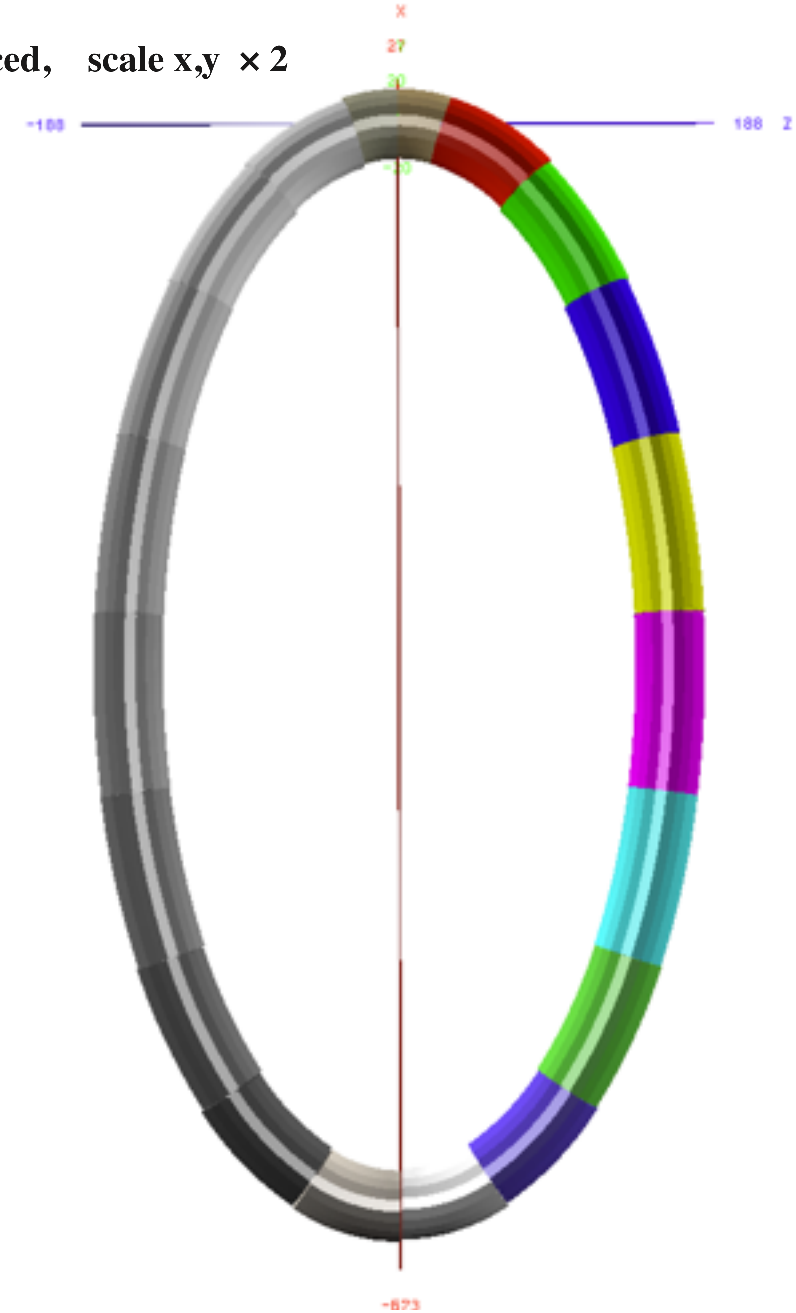


**Geometry generated from MAD-X SingleBend ring lattice, using makethin with 20 thick slices**
**ROOT/GEANT4 geometry generated with MDISim module Tfs2Geom**

# Scaling x, y

**SingleBend lattice sliced, scale x,y × 2**



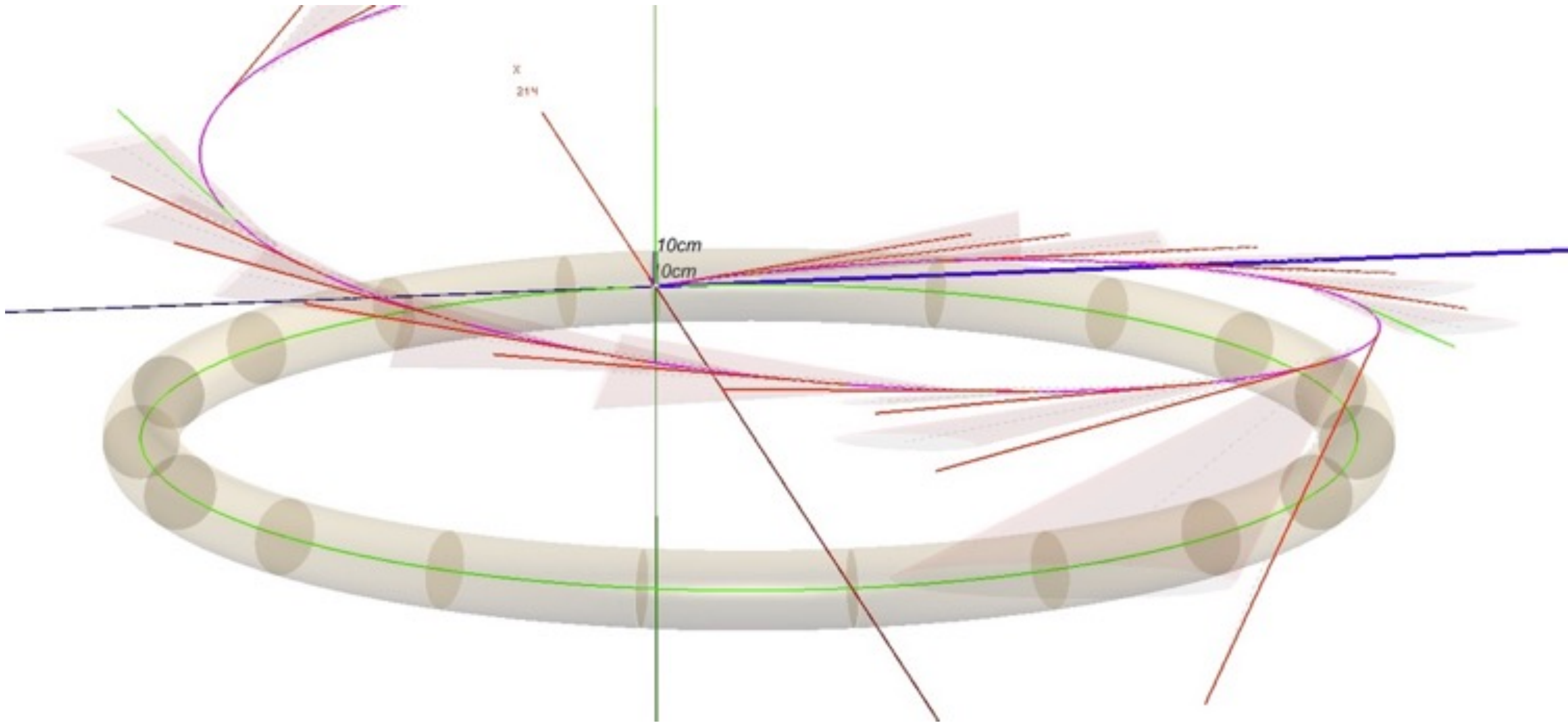


**For illustration**

**no need to be perfect**

```
CourantSnyderSystem::Get_SurVey_Scaled
x_eu=scale_x * V[i].r[0];
y_eu=scale_y * V[i].r[1];
theta = atan2(scale_x* W[i].m[0][2], W[i].m[2][2]);
```

# Test geometry + trajectory + SR cones



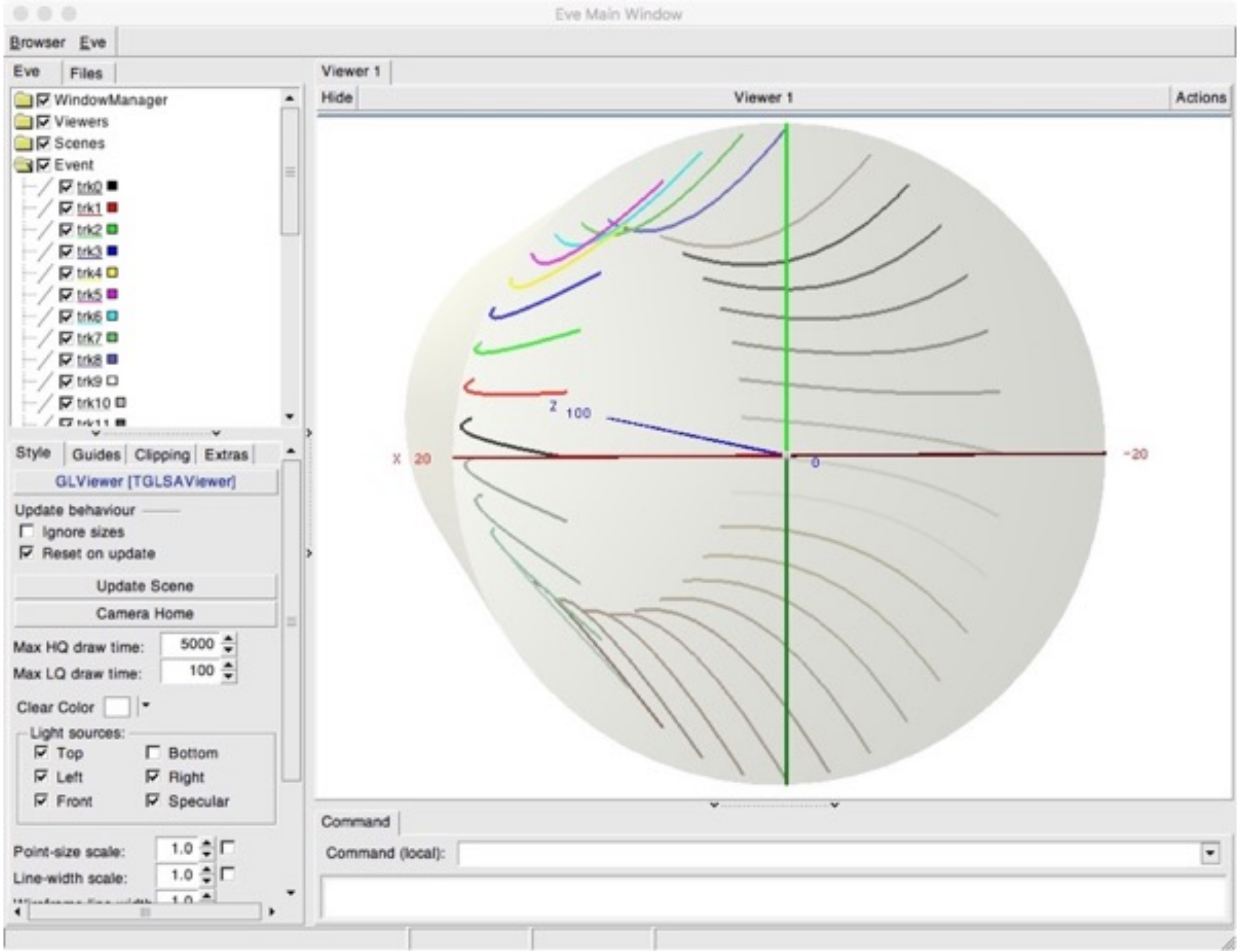

**MAD-X**

**Geometry : SingleBend ring, 20 slices**

**Trajectory : with initial angles px=0.05, py=0.1**

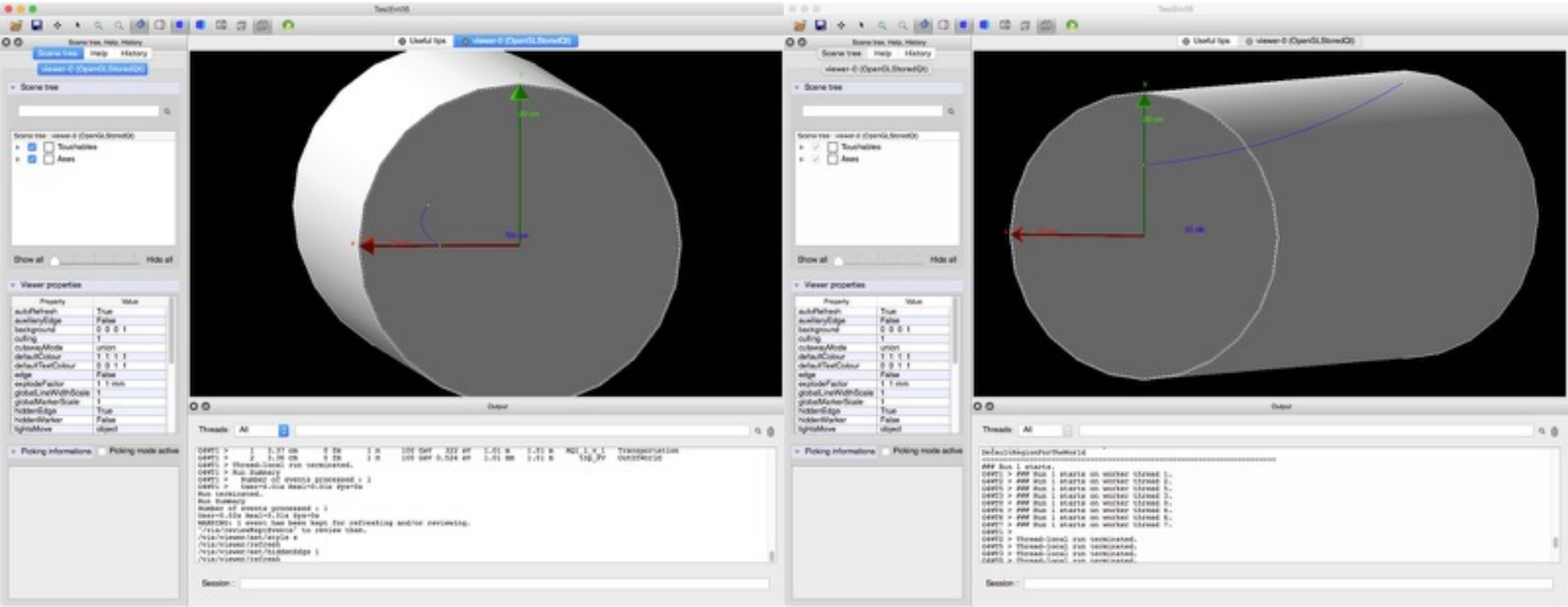
**MDISim : generation of geometry, 3d plotting of geometry and MAD-X trajectory calculation and plotting of synchrotron radiation cones**

# Quadrupole, MAD-X - MDISim level




SingleQuad.madx + tracking + plotting in MDISim
Focusing in x (curved to inside), defocusing in y. 500 T/m.
1m long, 20 cm quad radius, 16.5 cm offset.

# Quadrupole GEANT4

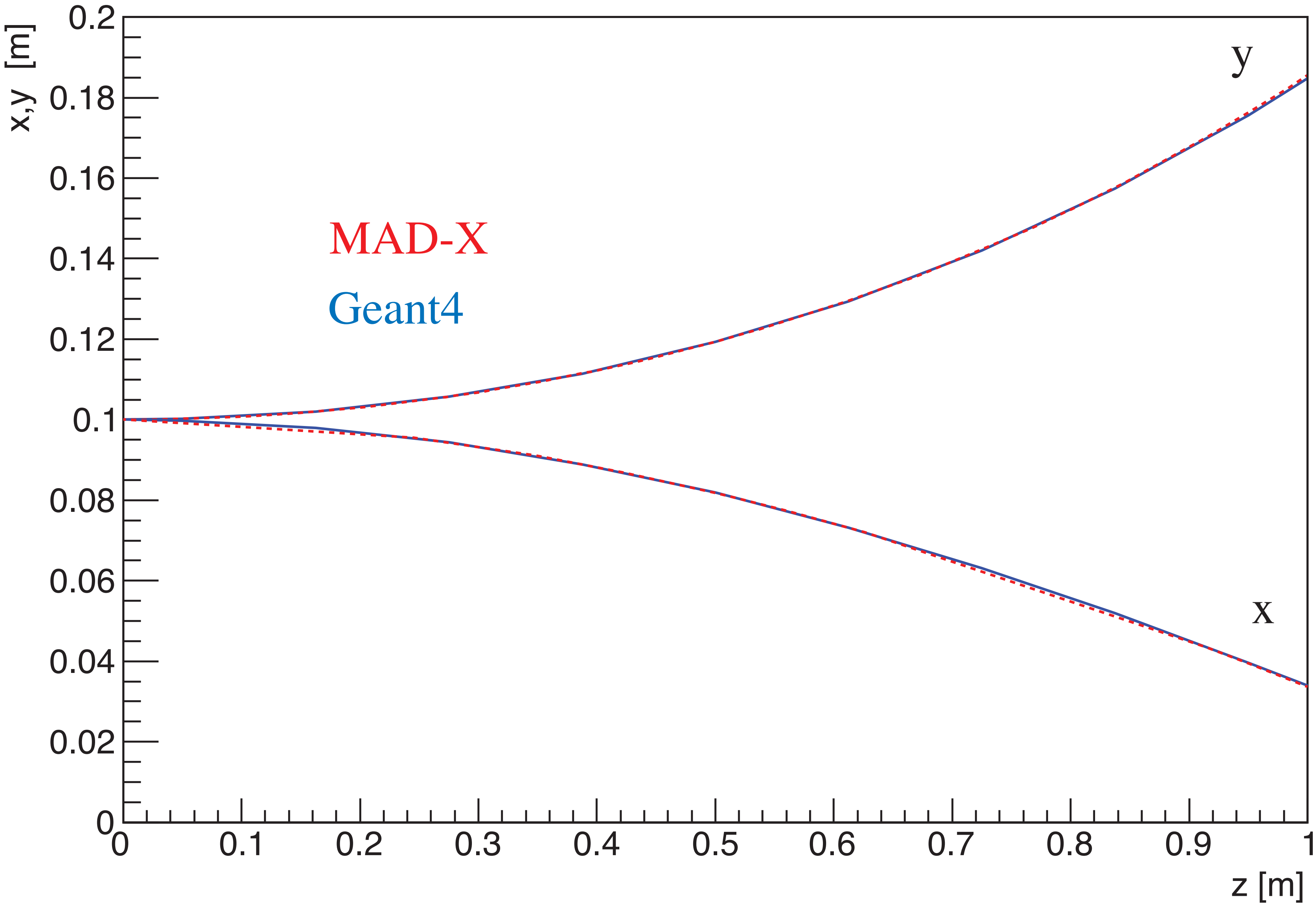


**100 GeV e+, SingleQuad geant4 500 T/m.**
**1m long, 20 cm quad radius, 10 cm offset in x (left) and in y (right)**
**SynRad off**

**Geometry + fields automatically generated from MAD-X using BDISM ---> input to Geant4**

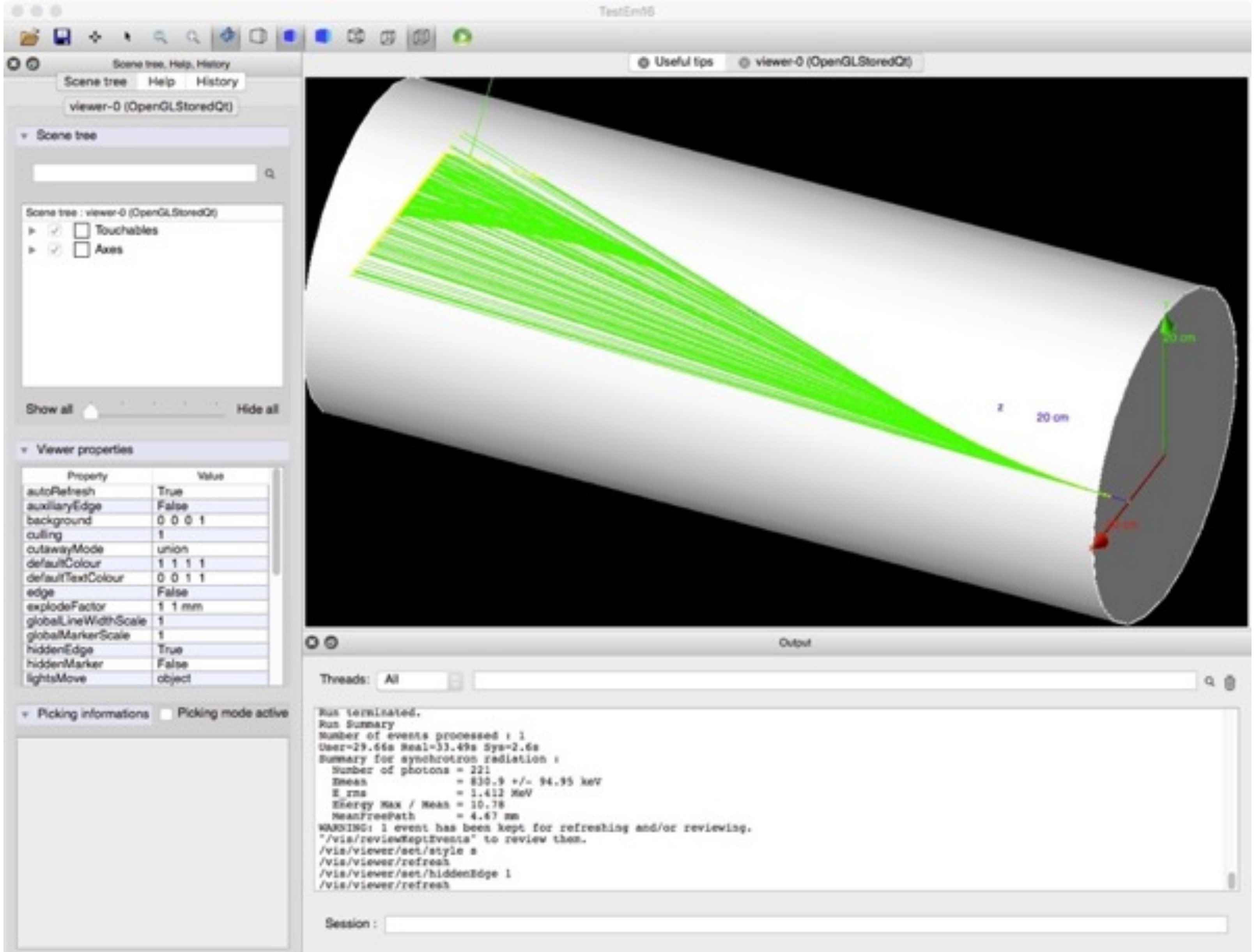
```
quadgradient = –brho * K1l * tesla / MagLen ;
G4QuadrupoleMagField* quadrupoleMagField = new G4QuadrupoleMagField (quadgradient ) ;
fLocalFieldManager = new G4FieldManager ( quadrupoleMagField ) ;
```

# Comparison MAD-X Geant4 Quadrupole




MAD-X : using MAKETHIN, 10 thick quadrupole slices to visualize trajectory in quadrupole

# GEANT4 Synchrotron Radiation in Quadrupole



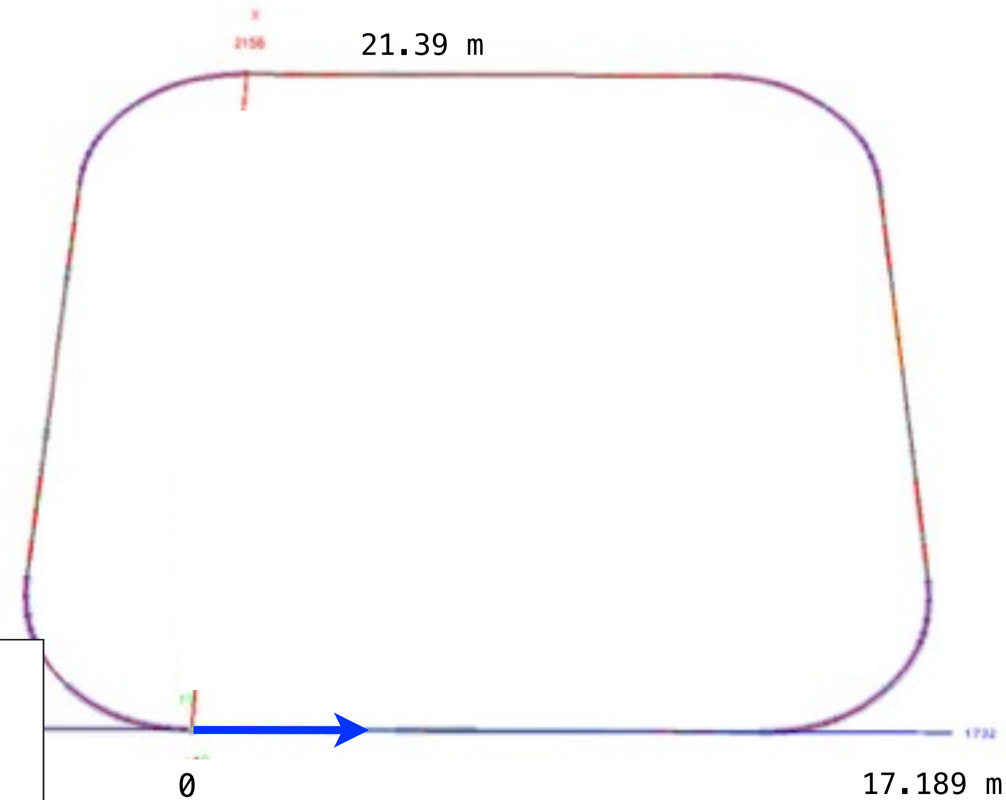

10 GeV e+, SingleQuad 500 T/m

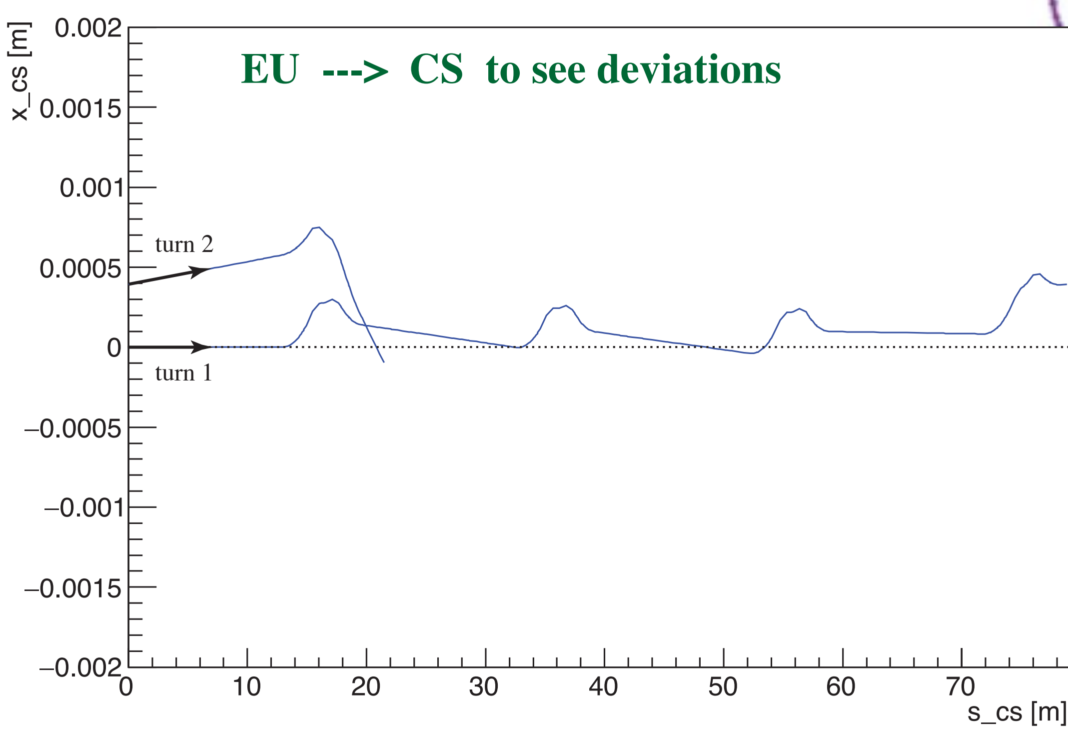
1m long, 20 cm quad radius, 10 cm offset in x, SynRad on

# GEANT4 default tracking precisions

**Real life example [LEIR](LEIR)**
**getting few turns with GEANT4**
**Here stopped after 100 m**
**< 1mm deviations**
**not bad as start**

small enough to track without scaling l = 78.54370 m




**first trial - not yet optimized :**
**GEANT4 has many field tracking options**

# Concluding remarks

Good progress in technical details --- still lots of room for further improvements :

- check / fill / clear volume boundaries
- check options to optimize tracking with fields in GEANT
- which precision is needed / expected for tracking magnetic fields with solenoid around IR

--- at present no solenoid with crossing angle in MAD-X

--- some workaround could be done with MAKETHIN --- but better proper implementation

- which tool best for which purpose --- how much accelerator tracking with G4 ?
- check and if needed define more complex volume types ? - beam screen ?

Start the next major step : Detailed tracking around IR

- Synchrotron radiation with beam pipe, shields, collimation
- Various loss processes, radiative Bhabha, beam-gas ..
- Link to detector side

Our team will be strengthened soon

Marian Lückhof / doctoral student 1. Nov, Belgin Pilicer / fellow 1. Dec