Pyg4ometry
load, manipulate, visualize, convert GDML, Fluka, CAD geometry

Stewart Boogert (stewart.boogert@rhul.ac.uk), Laurie Nevay, William Shields

G4 technical forum, September 2022

ROYAL HOLLOWAY UNIVERSITY OF LONDON

https://bitbucket.org/jairhul/pyg4ometry/src/develop/
http://www.pp.rhul.ac.uk/bdsim/pyg4ometry/

Introduction

Geometry tools in python for Geant4/Fluka/MCNP etc

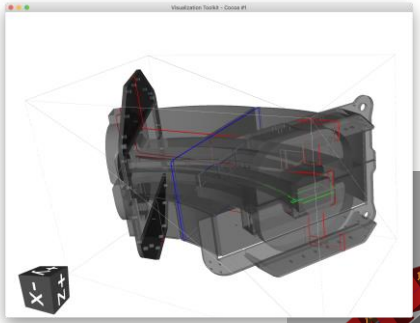
- Develop a (rapid) workflow inspired from parametric CAD to modify geometry in a reproducible way
- Key technical development : python API for loading/manipulating geometry
- **Have users**
- Of course, my group at Royal Holloway ;-)
- Accelerator community (PSI, ULB, CERN, DESY, Johannes-Universität Mainz, Warwick)
- Laser/plasma (QUB)
- HEP (LHCb, Legend, MESA, LUXE, nuStorm)
- **Paper published**
- **New features discussed here** (since https://indico.cern.ch/event/872309)

Paper : https://doi.org/10.1016/j.cpc.2021.108228

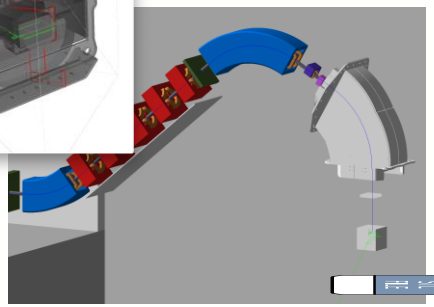
Manual : http://www.pp.rhul.ac.uk/bdsim/pyg4ometry/

Features (see 2019 Technical forum talk)

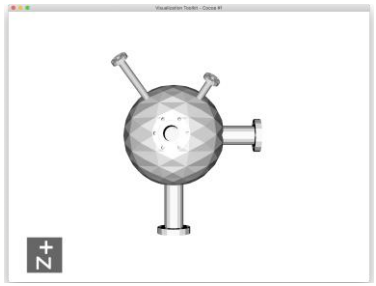
ROYAL HOLLOWAY UNIVERSITY OF LONDON

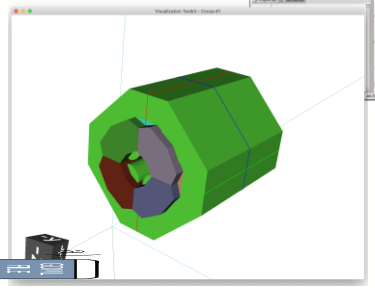


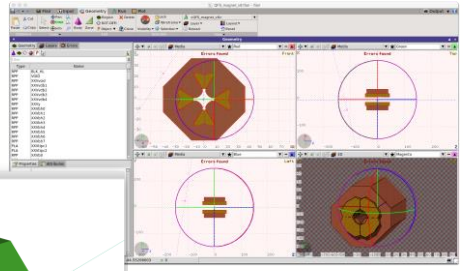
Load CAD



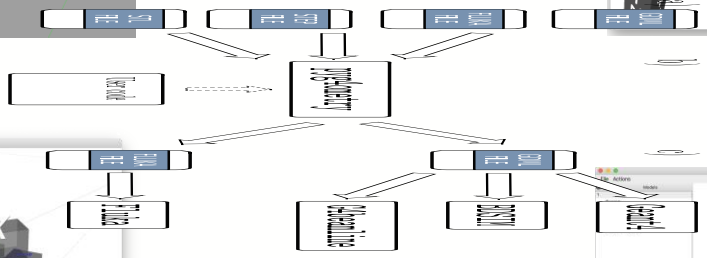
Parametric creation



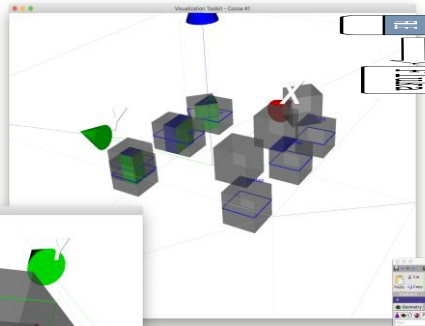


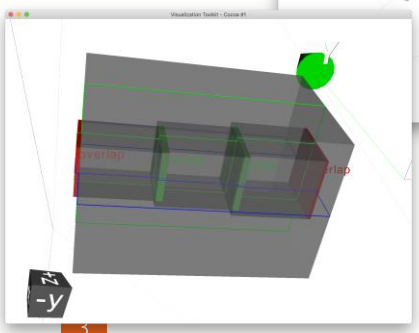


Load FLUKA

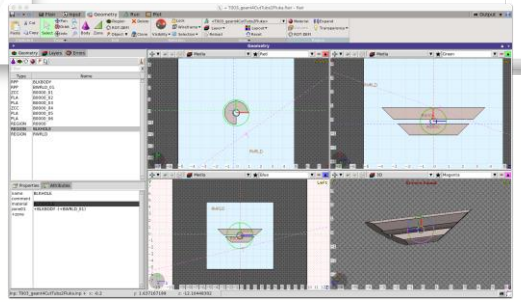


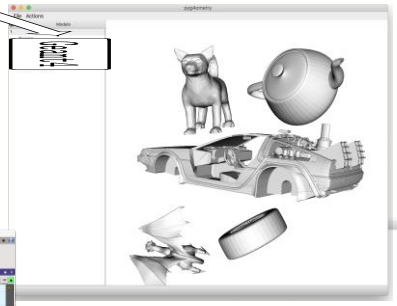
Coverlap detection



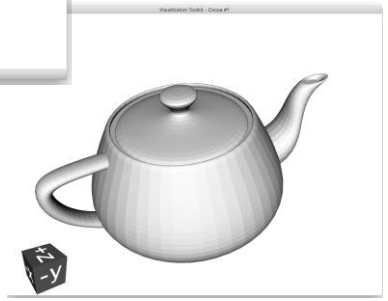


Save FLUKA



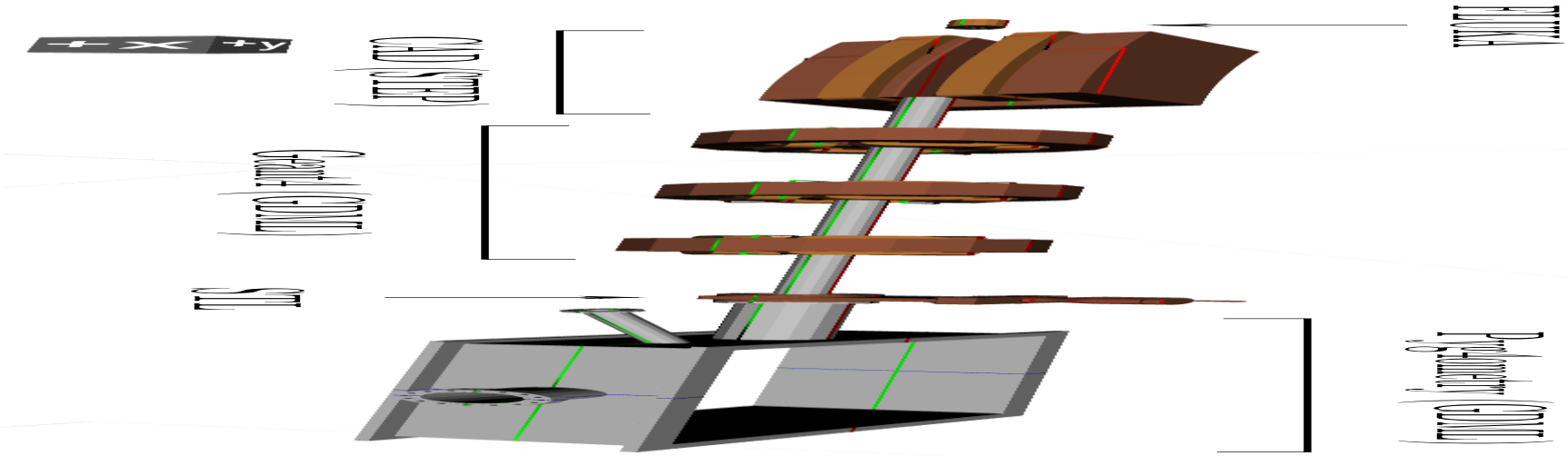


Load STL



Compositor system

Create a complete system with different geometry sources



Technology tools and dependencies

All dependencies are all open source and well maintained

Python

ANTLR

Visualisation Tool kit

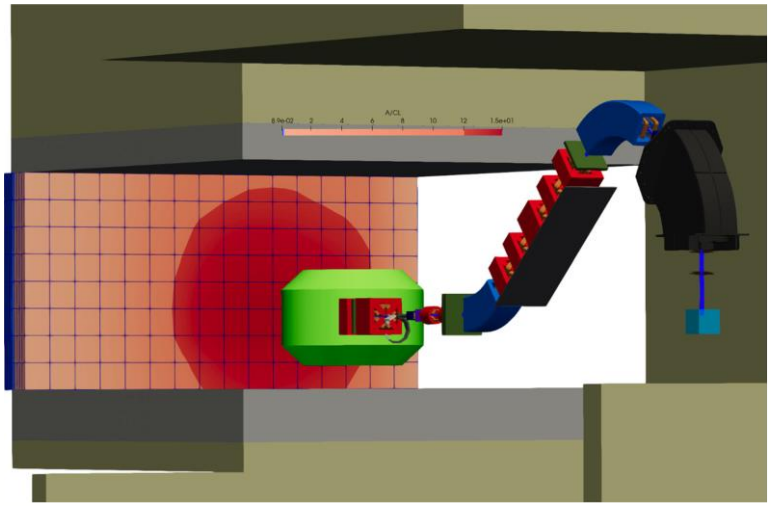
Open Cascade

Computational Geometry Algorithms Library

Symbolic Python

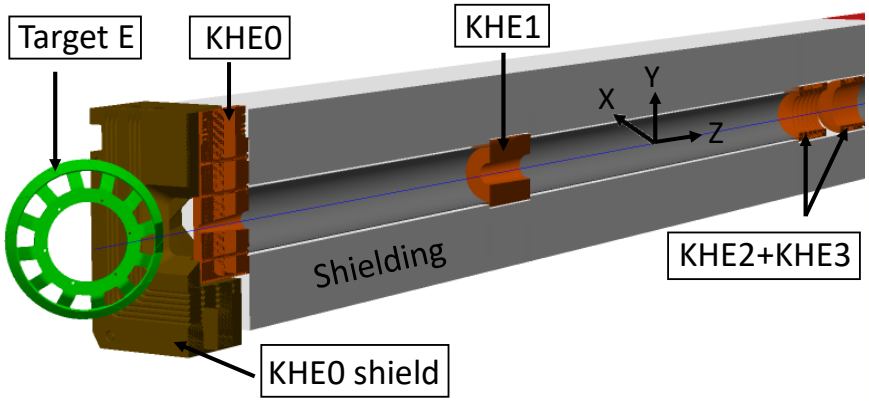
Users' applications

- IBA Protus One proton therapy system (ULB/IBA/RHUL)
- Viewed in Paraview



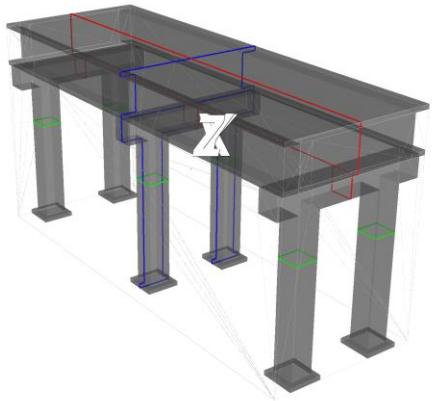
https://doi.org/10.1140/epjp/s13360-022-02960-9

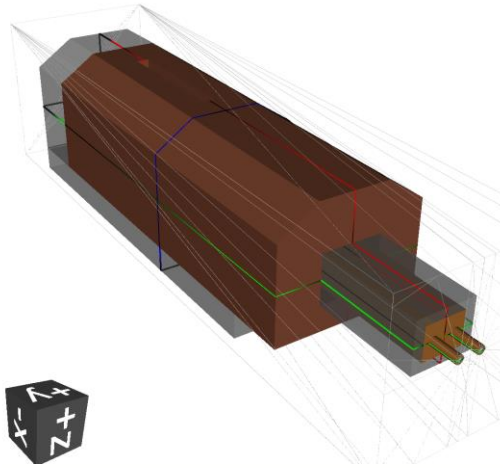
- HIPA PSI transfer line
- Transfer line from high power cyclotron

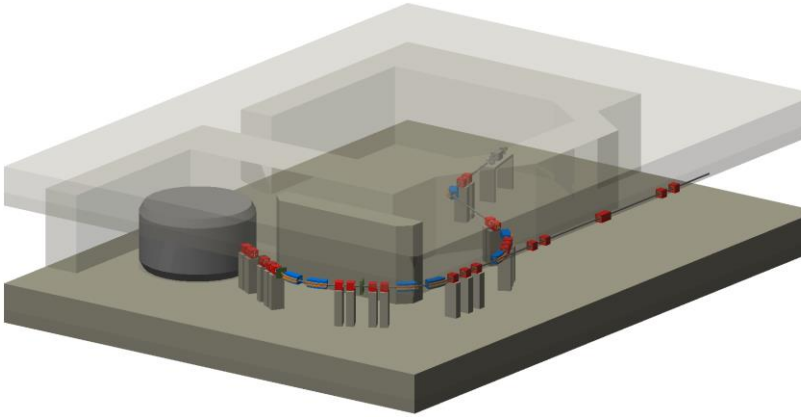


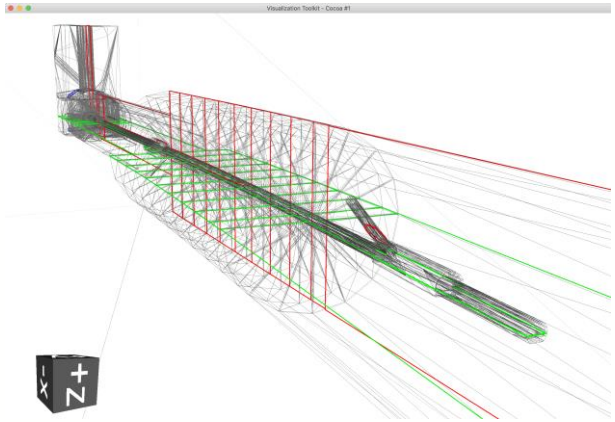
arXiv:2205.12536

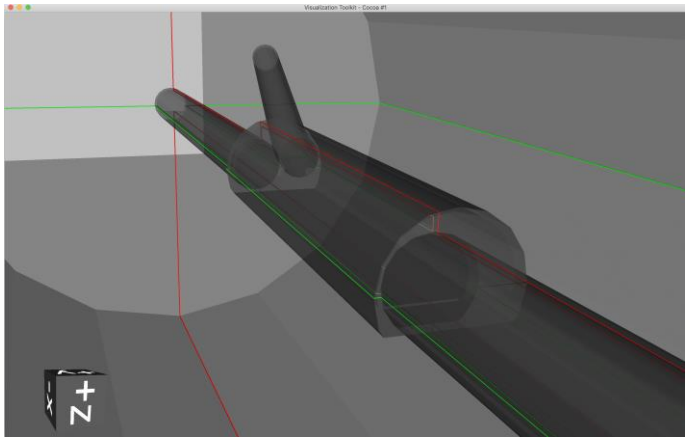
In house examples











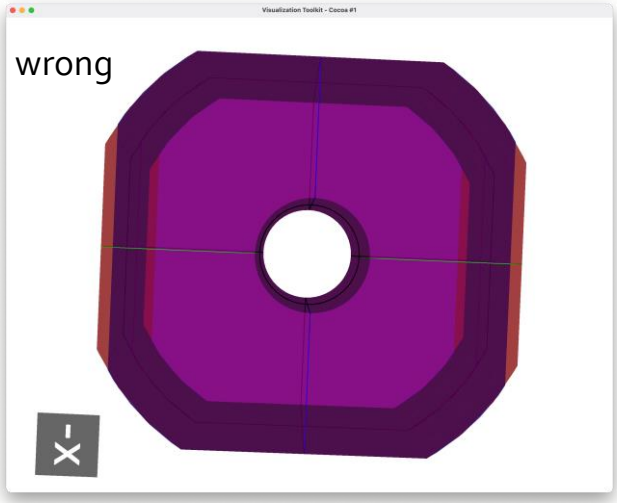
Geometry comparison

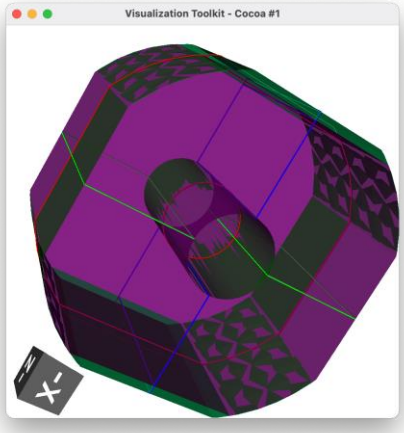
- When converting between geometry formats it is useful to compare geometries
  - irrespective of if the conversion was done with pyg4ometry or not
- "Comparison" could mean strict equality, but also equivalence
  - could construct geometry from different primitives but want to check the resultant volume
  - we may know the geometry is different but want to compare within a tolerance
- General strategy:
  - **Tests** class - Boolean flags of which things to test, e.g. solids, materials, volume, surface area
  - **TestResult** class - pass / fail, details on what was compared, nice message if failed
  - **functions** for each comparison, e.g. logicalVolumes, solids, materials, assemblies
  - **higher level functions**, e.g. geometr(treeA, treeB), gdmlFiles(fileA, fileB)
- We choose the tests we want to run and apply them to a geometry tree producing a set of results that can be printed out or summarised
- Used by LHCb to compare their new MC geometry (GDML, ROOT, custom)

Test Name
names
namesIgnorePointers
nDaughters
solidExact
shapeExtent
shapeVolume
shapeArea
placement
scale
copyNumber
materials
materialClassType
materialCompositionType
testDaughtersByName

Geometry comparison

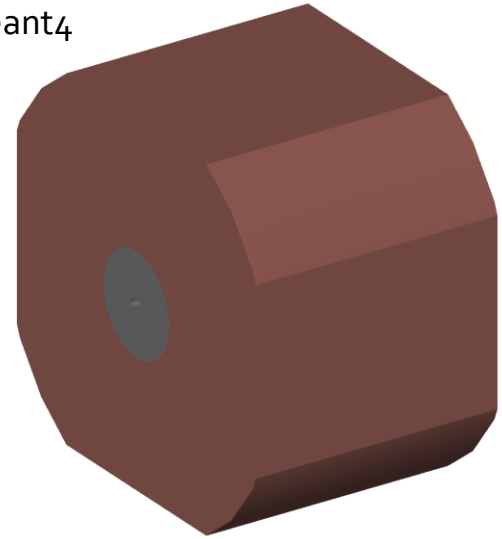
- Tolerance also for pointers in object names
- A small example is a collimator for a CERN North Area beamline
- CAD-converted tessellated (original) solid vs. (new) geometry from primitives (for improved simulation speed)
- Compare by volume and surface area - exploit visualisation meshes in pyg4ometry!
- Visualise two geometries on top of each other as well as the difference to see the problem





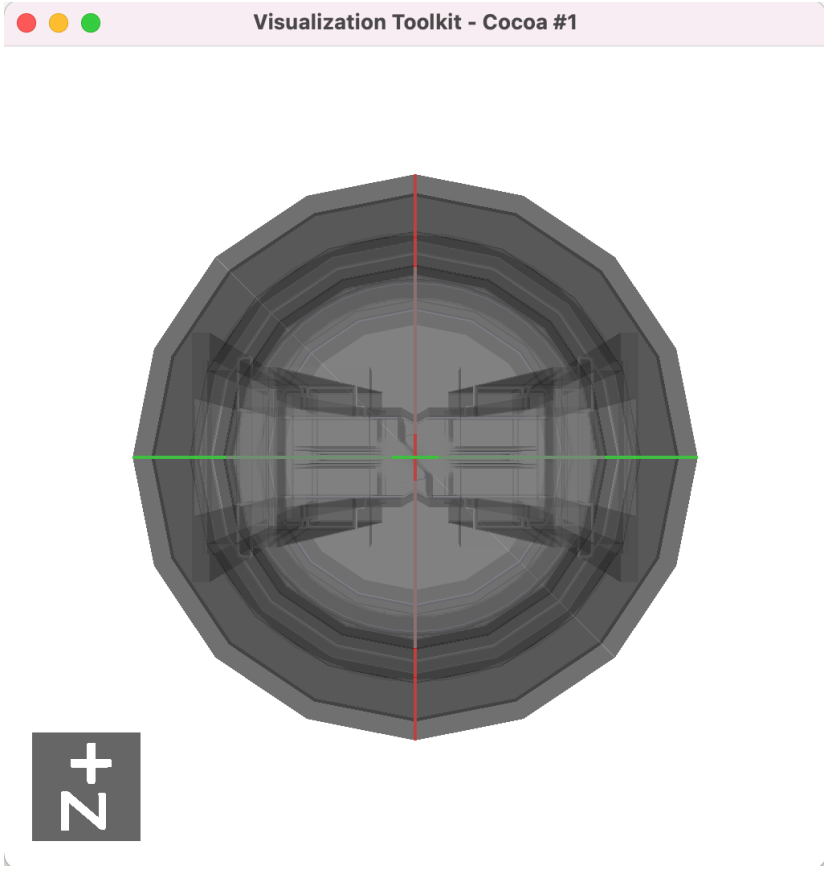
corrected

in Geant4



ROOT file loading

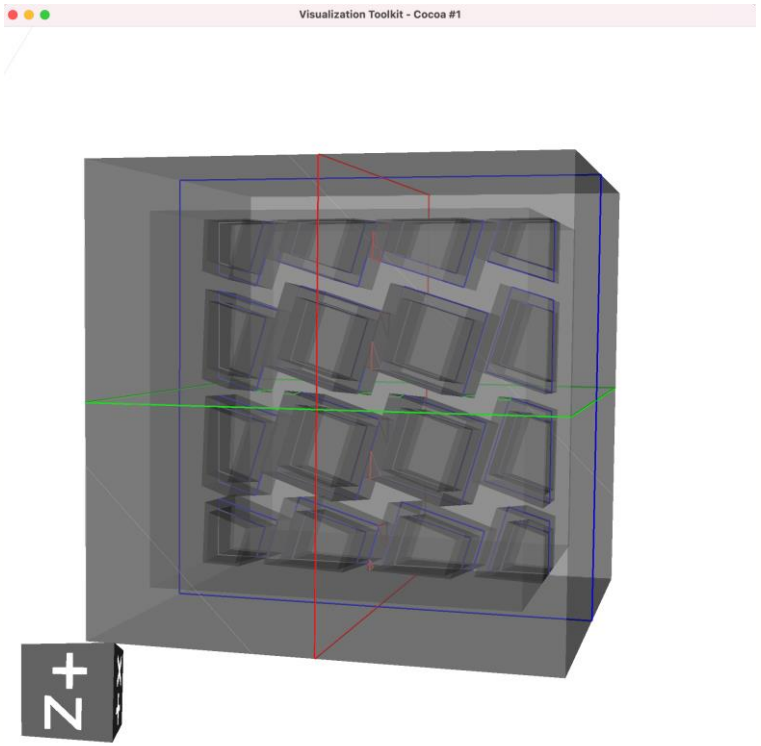
- ROOT has excellent python bindings.
- ROOT's TGeo is also excellent
- Format is being used (DD4Hep etc.)
- Load root file via standard bindings and transfer to pyg4ometry internal memory format
- Detector developers can check their geometry etc.



LHCb Velo : T. Latham

Clipping geometry for reuse

- What if people provide geometry and it is just too much? (Happens a lot in accelerator community e.g., FCC-ee IR region)
- Need a good mechanism to safely trim the geometry safely
- Replace mother volume and recursively cut down the daughter volume solids using Boolean intersections
- Still needs developing for replica, parametric solids etc.
- Would avoid this as would cause and explosion of Boolean solids if not careful



Geant4 to FLUKA conversion

ROYAL HOLLOWAY UNIVERSITY OF LONDON

- Follow CERN FLUKA
- Added materials
- Tested on relatively large experiment (LUXE)
- Planning on adding features beyond geometry (other non-geo CARDS)

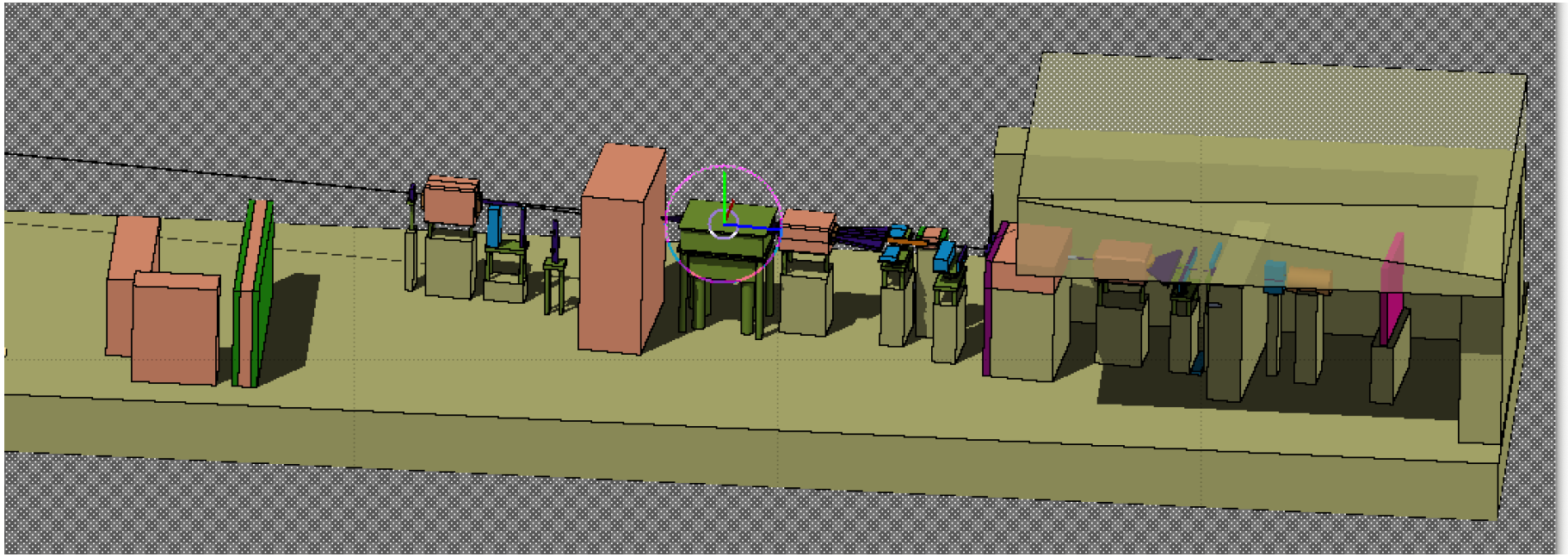
Still problems with

- General tessellated solids
- Twisted solids
- Solution implemented using convex decomposition is very unstable and very time consuming, rarely works
- Might base new solution on tessellation or voxelization

LUXE experiment

Non-linear QED experiment, planned for the EU.XFEL at DESY

- Measure Compton (BW) between laser and XFEL beam (Bremhstralung)
- Need FLUKA simulations for dump studies and radioprotection

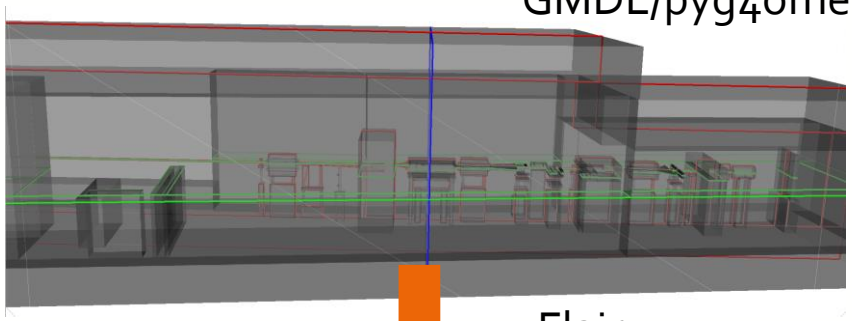


FLUKA workflow for LUXE

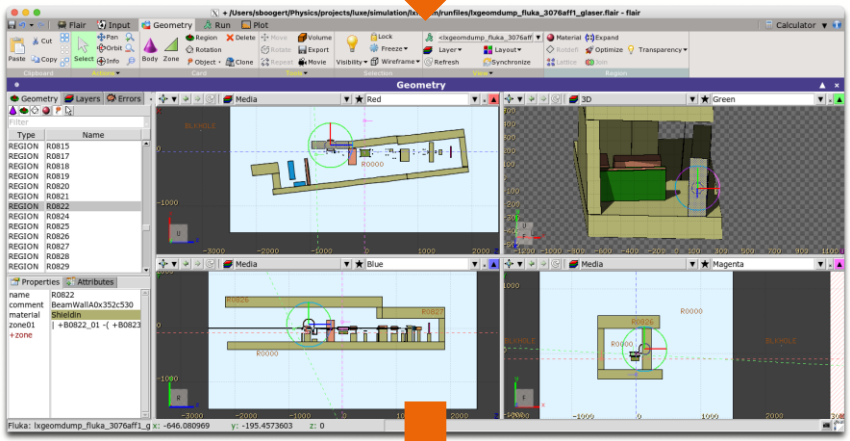
1. Run lxsim and generate GDML geometry
2. Convert lxsim GDML geometry to FLUKA
3. Augment with control cards (BEAM, BEAMPOS…etc)
4. Run FLUKA jobs
5. Merge output from all jobs (utilities provided by Fluka)
6. Plotting in matplotlib and/or VTK (pyvista)

Original conversion done by 1st year PhD student in hours

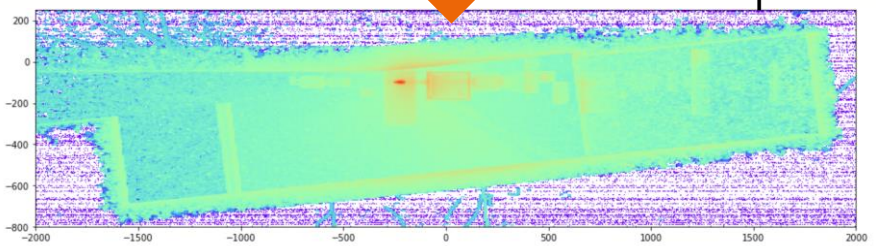
GMDL/pyg4ometry



Flair

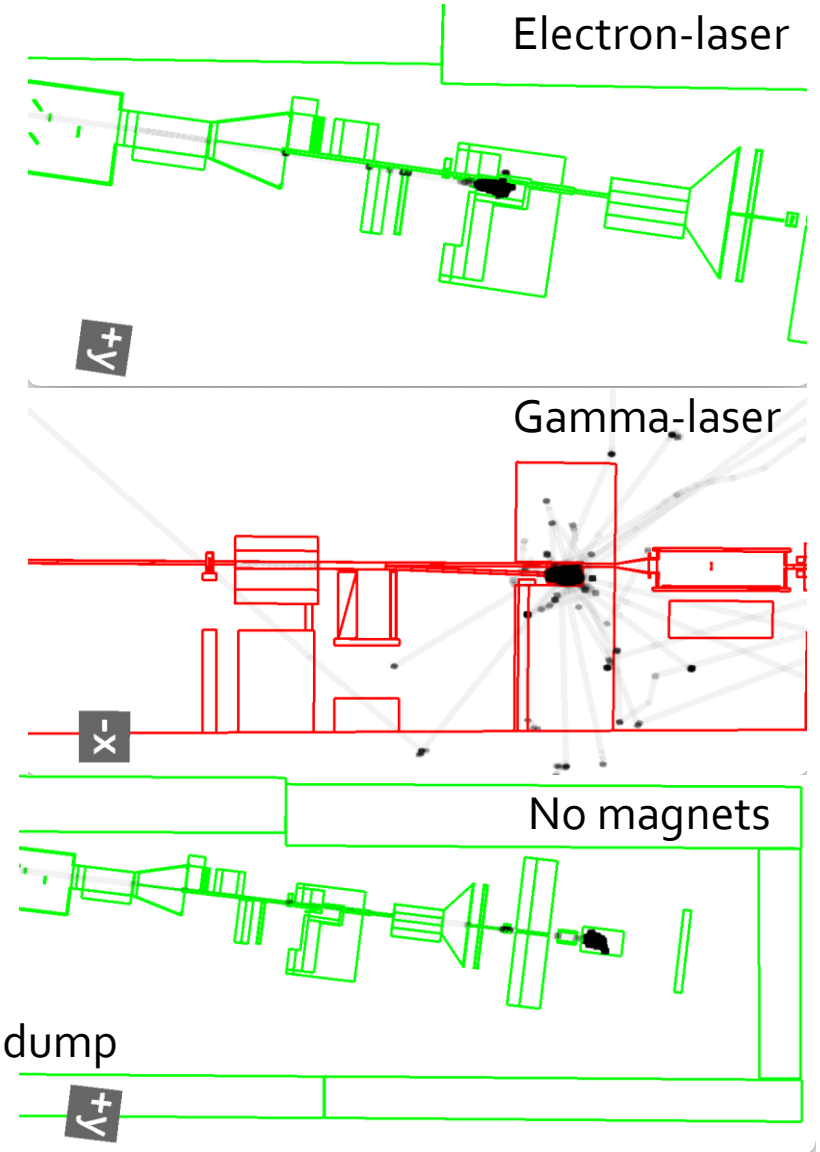


Fluka output



Model verification

- Geometry
- Very advanced conversion (written by me for another project)
- Flair (GUI for FLUKA) would indicate geometry errors (overlaps)
- Materials
- Converted from Geant4 down to isotopic composition
- Hard to independently check (could with simplified models i.e a block of material in G4/Fluka)
- Magnetic fields (3 important)
- Track test particles and dump all trajectories (see right)



2D sections created by pyg4ometry, data : fluka dump

Improved use of CGAL

ROYAL HOLLOWAY UNIVERSITY OF LONDON

- CGAL is one of the most templated libraries I have ever encountered.
- Previous sol^{n} :Push problem into monolithic C++ lib with very small API
- Loose power and utility of CGAL
- Split and bind sensible fraction of the CGAL API to python. Keeping as close to the CGAL API as possible
- Users can write their own algorithms!
- **Implemented tetrahedralization and mesh repair in hours**

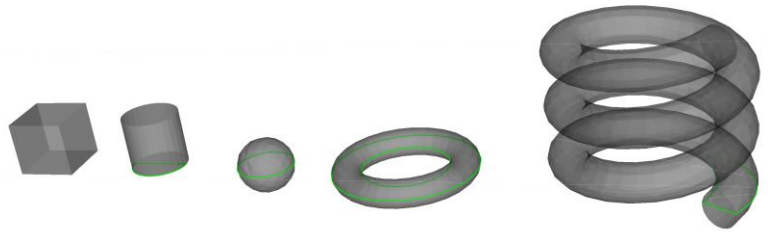
```
@classmethod
def nefPolyhedron_to_convexPolyhedra(cls, np):

    CGAL.convex_decomposition_3(np)
    vi = np.volume_begin()
    ve = np.volume_end()
    pList = []
    while vi != ve :
        si = vi.shells_begin()
        se = vi.shells_end()
        if vi.mark() :
            while si != se :
                p = Polyhedron_3.Polyhedron_3_EPECK()
                np.convert_inner_shell_to_polyhedron(si,p)
                pList.append(p)
                si.next()
        vi.next()

    return pList
```

Improved CAD file handing

ROYAL HOLLOWAY UNIVERSITY OF LONDON

- Previously used FreeCAD
- Exceptionally large dependency (but had python)
- Move towards OpenCascade (C++ library FreeCAD is based on)
- Complete replication of CAD model with
- Assemblies (LV/AV)
- Parts (Solids/PVs)
- Allows the tessellation of CAD models in an efficient manner without creating too many tessellations etc.
- Allows for the later replacement of tessellated objects with minimal effort

```
def commonCode(fileName, mats ={}, skip = [], mesh = {}) :
    fileName = _os.path.join(_os.path.dirname(__file__), fileName)
    r = _pyg4.pyoce.Reader(fileName)
    #r.shapeTool.Dump()
    ls = r.freeShapes()
    worldName = _pyg4.pyoce.pythonHelpers.get_TDataStd_Name_From_Label(ls.Value(1))
    reg = _pyg4.convert.oce2Geant4(r.shapeTool, worldName,mats,skip,mesh)
    wa = reg.logicalVolumeDict[worldName]
    wl = wa.makeWorldVolume()
```

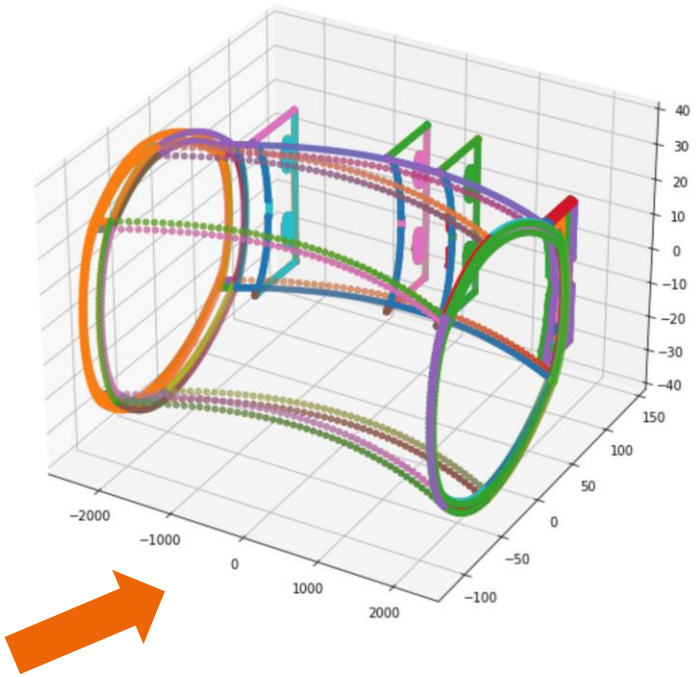


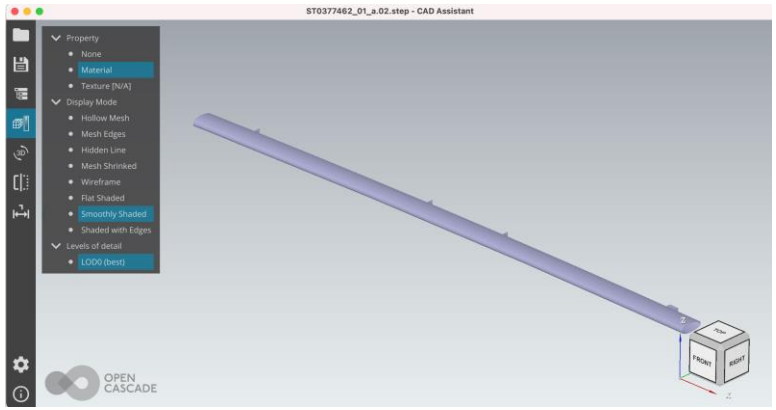
Note bene : excellent user control over mesh quality

CAD feature extraction

- CERN PS beam pipes in Geant4
- Slightly curved and need to place them accurately as define the accelerator aperture
- CAD directly to STL is ok for simulation but how to place the object accurately?
- **Extract data directly from CAD file and user can determine features that conversion can be based on**

```
[3]: r = _pyg4.pyoce.Reader("./ST0377462_01_a.02.step")
     l = _pyg4.pyoce.pythonHelpers.findOCCShapeByName(r.shapeTool,"ST0377462_01")
     s = r.shapeTool.GetShape(l)
     pnts = beamPipeCADFeature1(s)
     fig = _plt.figure()
     ax = _plt.axes(projection='3d')
     for i in range(0,len(pnts)) :
         ax.scatter3D(_np.array(pnts[i])[:,0],_np.array(pnts[i])[:,1],_np.array(pnts[i])[:,2])
```





Packaging still a pain in Python with extensions

ROYAL HOLLOWAY UNIVERSITY OF LONDON

Really hindered adoption

- Skbuild (CMake)
- Part of scikit-hep
- Meson (+pep517)
- Setuptools (+sprinkle of CMake)

Systemic python problem.

- **Plan to make package part of scikit-hep**
- **Contribution from L. Pertoldi**
- Build fix based on skbuild and Cmake
- Not quite ready yet

New features

ROYAL HOLLOWAY UNIVERSITY OF LONDON

Soon

- Tetrahedralization of meshes (for tracking and Multiphysics)
- USD export for high quality rendering and VR/AR
- Improved VTK visualization (significant speed improvements)
- Mesh hashing for geometry comparison (check final solid mesh geometry and not details of CSG tree)
- Command line interface for simple operations

Longer term

- Conversion of CAD to G4 primitives
- Lots of progress in this area in the graphics community
- Direct STEP/IGES output
- More use of CGAL
- Alpha wrap for meshes
- New coplanar algorithms (detect geometry close of geometry safety limits)
- Paraview module to directly load geometry (some discussions with Kitware)

Summary and conclusions

ROYAL HOLLOWAY UNIVERSITY OF LONDON

- **Pyg4ometry** has reached a high level of maturity
- Agnostic system : not for any community (medical, space, HEP etc)
- Pyg4ometry is routinely used in multiple experiments
- Found a home in accelerator applications of Geant4
- Developments are now incremental, but packaging is still a major problem
- Improved material properties (M. Hubert)
- Being used in the wild by various experiments
- Packaging will change this significantly
- Better packaging and building on the way (L. Pertoldi)
- Pypi packages for most platforms coming soon
- Also docker images are available (but need updating)
- Health warning : Not for massive geometry. Need lots of optimization

Acknowledgements

Pull requests merged from

Thomas Latham

Luigi Pertoldi

Manuel Hubert

Former group members

Stuart Walker (DESY)

Andrey Abramov (CERN)

Users with feedback, changes and PRs

Cedric Hernalsteens

Elliot Ramoisiaux

Robin Tesse

Eustache Gnacadja