THE FLUKA LINEBUILDER AND ELEMENT DATABASE: TOOLS FOR BUILDING COMPLEX MODELS OF ACCELERATOR BEAM LINES

A. Mereghetti*, University of Manchester, M13 9PL, UK, and CERN, Geneva, Switzerland
V. Boccone, F. Cerutti, R. Versaci, V. Vlachoudis, CERN, Geneva, Switzerland

Abstract

Extended FLUKA models of accelerator beam lines can be extremely complex: heavy to manipulate, poorly versatile and prone to mismatched positioning. We developed a framework capable of creating the FLUKA model of an arbitrary portion of a given accelerator, starting from the optics configuration and a few other information provided by the user. The framework includes a builder (LineBuilder), an element database and a series of configuration and analysis scripts. The LineBuilder is a Python program aimed at dynamically assembling complex FLUKA models of accelerator beam lines: positions, magnetic fields and scorings are automatically set up, and geometry details such as apertures of collimators, tilting and misalignment of elements, beam pipes and tunnel geometries can be entered at user's will. The element database (FEDB) is a collection of detailed FLUKA geometry models of machine elements. This framework has been widely used for recent LHC and SPS beam-machine interaction studies at CERN, and led to a drastic reduction in the time otherwise required to rework old machine models, and to a coherent and traceable description of the inputs used for all the simulations.

INTRODUCTION

A relevant asset of the FLUKA [1, 2] Monte Carlo code is the use of a simple text file as source of input information, where the user states all the simulation settings and fully describes the geometry, with very limited need of coding. Simulations of extended accelerator complexes imply the use of large and "static" files: their difficult modification can be an important liability, especially in case of design of new elements or systems, for which flexibility is a desired feature. Moreover, simulating many points of interest of the same accelerator complex, like the eight Insertion Regions (IRs) of the Large Hadron Collider (LHC), leads to at least an equal number of FLUKA input files: obvious problems of synchronisation arise, concerning in particular geometry definitions. Import/export of elements from one geometry to another one, coherent material definition, propagation of follow-ups and implementation of new details become issues as well.

A new strategy has been recently adopted by the FLUKA Team at CERN: the definition of each accelerator element (including magnets, collimators, monitors, absorbers, tunnel areas...) is collected in a database, featured by a library of shared materials; the final geometry is automatically created by a builder, on the basis of the optics of the machine and directives customised by the user. Clear assets are:

- full synchronisation of the final FLUKA input file with the optics of the machine; moreover, the user is less concerned with precise positioning and magnetic settings of elements, automatically assured, and comparisons between different optics can be easily set up;
- many small specific FLUKA files can be handled and debugged more easily than a single huge file; moreover, follow-ups in the geometry description of elements can be instantaneously propagated, with clear benefit to all the concerned users;
- great flexibility in importing and exporting elements.

Everything is kept under the Subversion (SVN)[1] revision control system.

A similar approach was already set up for collimator induced shower simulations concerning the LHC IR3 and IR7 cleaning insertions, although based on only one big input file and a builder fairly focussed on these regions. The present approach can be applied to *any* portion of *any* accelerator of interest, once the suitable database of elements is set up.

Latest developments in the FLUKA code concerning the assembly of separate input files and new directives for geometry transformations have significantly simplified the overall process and increased the robustness of the built geometry.

Framework

SVN repositories for many IRs of the LHC, Long Straight Sections (LSSs) of the Super Proton Synchrotron (SPS), and the two injection lines TI2 ("Tunnel d'Injection") and TI8 have been set up and given a common internal structure, so that useful FLUKA user routines and other simulation settings can be easily exported and used for different portions of the same machine, or even for a different machine. Configuration scripts and analysis tools have been developed accordingly.

A set of special FLUKA source routines is available, for simulating the interaction of the beam with the residual gas in the pipe, for the readout of loss maps from tracking codes like SixTrack [3], and for the readout of beam distributions at a certain location along the accelerator as computed by optics codes like MADX [4].

* alessio.mereghetti@cern.ch

[1] `http://subversion.apache.org`

ISBN 978-3-95450-115-1

THE FLUKA ELEMENT DATABASE

Every element inserted in the FLUKA Element DataBase (FEDB) is described isolated from the others, at the origin of the reference system: its full description is split in specific text files, declaring the needed analytical bodies, their logical combinations for obtaining regions, and the assignment of materials. Additional FLUKA files (e.g. defining useful scoring detectors or sizing the steps of particle transport in magnetic fields) and information files (e.g. technical documents, drawings, pictures. . .) can be stored as well.

Special elements called “assemblies” are obtained combining more than one basic element. Figure 1 shows a 3D rendering through the FLAIR Geometry Editor [5] of the Roman Pot assembly for the FLUKA geometry of LHC IR5: in order to create the assembly, the basic geometry model of the pots (featured by the beam pipe and the support structure) is combined with the geometry model of the detectors. The important asset of assemblies is the possibility of automatically modifying the relative positions of the used basic elements, according to user-defined or optics specifications (e.g. the position with respect to the beam axis at which the detector should be placed).

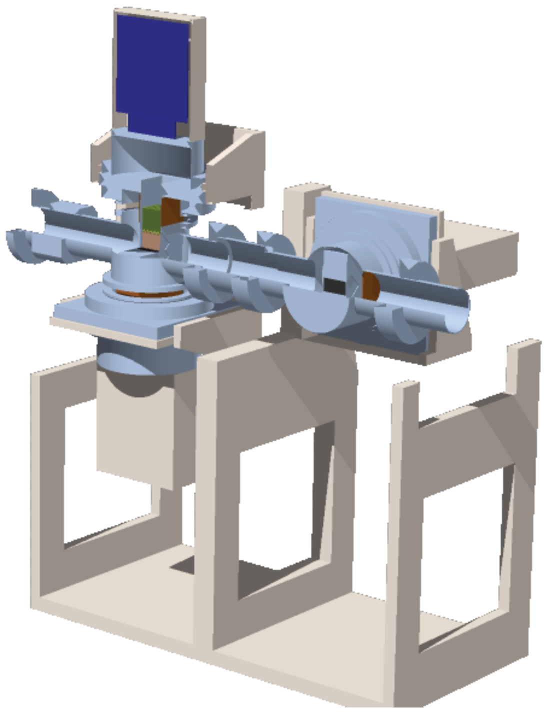
Figure 1: 3D rendering of the FLUKA geometry of the Roman Pot assembly. The geometry is cut in order to show the inside.

FLUKA geometries of tunnels and underground installations are stored as well, with a full 3D description whenever details of non-standard tunnel regions are needed (e.g. the LHC experimental caverns and the service areas UJ and RR), or with a simple 2D model of the cross section of a standard tunnel (e.g. the tunnel of the LHC arc).

Most of the elements of the LHC and the TI2/TI8 injection lines have been already modelled in FLUKA and inserted into the database, while SPS elements are steadily being inserted. The same applies to tunnels and underground installations.

THE LINEBUILDER

The LineBuilder is a Python program with a complete set of libraries, aimed at the generation of complex FLUKA geometries of accelerator beam lines, based on TWISS[2] files and directives from the user. It is particularly effective in assuring precise positioning of replicated elements, which can be rather painful and error-prone if done by hand, especially in case of long accelerator lines and tiny bendings. The user selects the needed prototypes from the FEDB, and the program automatically creates a replica for each occurrence in the TWISS files matching the corresponding name, with the correct position and orientation. The LineBuilder is provided with FLUKA routines for magnetic fields: different field types are available and, on the basis of the element strength, intensities are automatically scaled according to the magnetic rigidity of the beam. The user can assign any set of scoring detectors to any element family, to be repeated for every replica inserted in the geometry.

The beam pipe is automatically created, following user specifications about shape, dimensions, and materials. A limited amount of smooth transitions is allowed.

The user can put any element (but dipoles) at the origin of the resulting FLUKA geometry.

Auxiliary files allow the user to modify the sequence of elements, with no need of manually editing the TWISS file(s): it is thus possible for instance to insert elements off the beam line, e.g. Beam Loss Monitors.

Collimators can be singularly given the proper aperture, roll angle about their longitudinal axis, and jaw tilt by means of an external file: the user can thus easily implement different collimation settings for the same optics.

It is possible to embed the built portion of accelerator in the mentioned 3D model of a tunnel (provided the use of special flags in the tunnel file, instructing the program about the affected geometry regions), or to wrap a tunnel with a constant cross section around the machine.

Figure 2 shows 3D renderings of the last 270 m of the TI2 injection line: the presence of vertical dipoles further tilted by $\sim$15 mrad about their longitudinal axis leads to a very complicated chain of geometry transformations for placing the downstream magnets (that would be quite time-consuming, error-prone and particularly painful in case of editing by hand). In this case, the design orbit computed by MADX can be reproduced by FLUKA with an accuracy better than 1 μm over the entire geometry[3].

[2] A TWISS file is the output file from an optics code (e.g. MADX), stating the longitudinal coordinate, the associated optics functions (TWISS parameters) and strengths of each element along the beam line.

[3] It should be kept in mind that FLUKA tracks particles in a magnetic field dividing the curved path into linear substeps.

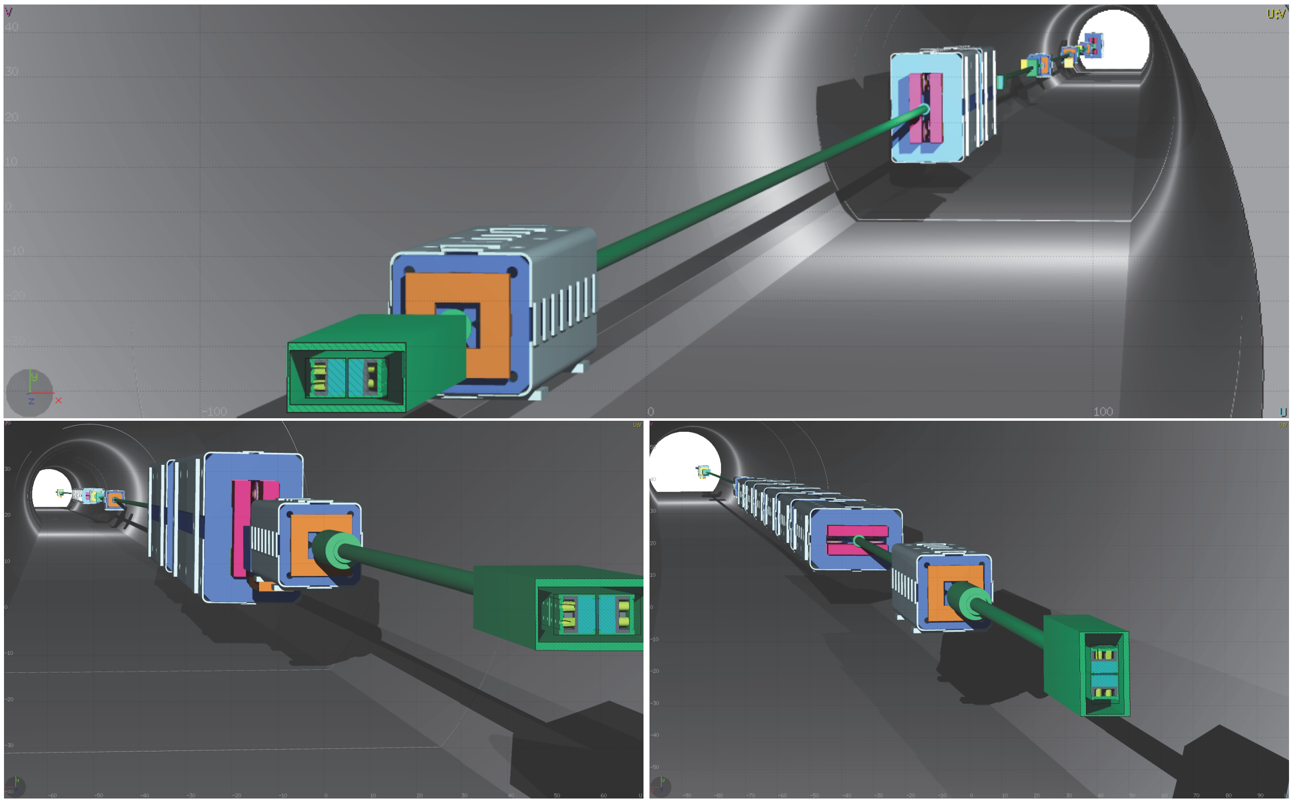

Figure 2: 3D renderings of the last 270 m of the TI2 injection line. Upper frame: view upstream of the TCDIH.29205 collimator. Lower frame, left side: view downstream of the TCDIH.29050 collimator. Lower frame, right side: view downstream of the TCDIV.29234 collimator. The quoted collimators are in foreground, transversely cut.

Future Developments

Although the LineBuilder can build the geometry of a two-beams accelerator, at present it cannot embed the geometry of an injection (or extraction) line into the one of the circulating beam: this feature can be interesting for supporting the design of a collimation system in the vicinity of such a point.

When the LineBuilder is used for building the entire geometry of a circular accelerator, precision issues on the closure of the ring (presently below a cm in the case of the SPS accelerator, i.e. over almost 7 km) and on the management of the region definitions arise. Follow-ups of the code in this direction can be of interest for relatively small machines, like the Proton Synchrotron at CERN or the DAΦNE accelerator at Laboratori Nazionali di Frascati, Italy.

3D scoring meshes (“USRBIN cards”) can heavily affect the computation time, especially if they are present in high number and if they are translated/tilted in order to match the position/orientation of the replica they are associated to. If a detector is assigned a purely translational transformation, its definition could be automatically modified, in order to drop the associated transformation, and consequently gain computation time.

CONCLUSIONS

An innovative approach to the preparation of extended FLUKA geometries of any accelerator beam line has been developed, based on the use of the FEDB and the LineBuilder, ensuring accurate positioning of elements, synchronisation with optics, easier handling of FLUKA input files, prompt propagation of geometry modifications, and uniform material definition.

REFERENCES

[1] A. Fassò et al., “FLUKA: a Multi-Particle Transport Code”, CERN-2005-10, INFN/TC-05/11, SLAC-R-773.

[2] G. Battistoni et al., “The FLUKA code: Description and Benchmarking”, Proc. of the Hadronic Shower Simulation Workshop 2006, Fermilab 6^{th}–8^{th} September 2006, M. Albrow, R. Raja eds., AIP Conf. Proc. 896, 31-49, 2007.

[3] G. Robert-Demolaize et al., Proc. of PAC05, Knoxville, TN, USA, 2005.

[4] F. Schmidt, G. Iselin, H. Grote, “MADX User Guide”, `http://madx.web.cern.ch/madx/`

[5] V. Vlachoudis, “FLAIR: A Powerful But User Friendly Graphical Interface For FLUKA”, Proc. of the International Conference on Mathematics, Computational Methods and Reactor Physics (M&C 2009), Saratoga Springs, New York, 2009.