Proceedings of ICFA Mini-Workshop on Tracking for Collimation in Particle Accelerators, CERN, Geneva, Switzerland, 30 October 2015, CERN Yellow Reports: Conference Proceedings, Vol. 2/2018, CERN-2018-011-CP (CERN, Geneva, 2018)

SixTrack Status

R. De Maria, A. Mereghetti, J. Molson, K. Sjobak
CERN, Geneva, Switzerland

Abstract

SixTrack is a 6D symplectic tracking code used to simulate single-particle dynamics in high-energy circular accelerators and storage rings. It is routinely used for the design and optimization of the LHC since its conception, and more recently for SPS, RHIC, HL-LHC, and FCC studies. This paper presents the status of the code, including the recent developments and those foreseen for extending the physics models and adding new features.

Keywords

SixTrack, symplectic tracking, single-particle simulations.

1 Introduction

SixTrack [1] is a 6D single-particle symplectic tracking code able to compute the trajectories of individual relativistic charged particles in circular accelerators for studying dynamic aperture or evaluating the performance of beam-intercepting devices like collimators [2]. It has been developed based on the 4D tracking code RaceTrack [3] by adding the longitudinal degree of freedom, introducing beam-beam forces, and extending the pre- and post-processing capabilities. It can compute linear and non-linear optics functions, time-dependent effects, and extract indicator of chaos from tracking data. SixTrack implements scattering routines and aperture calculations to compute “loss maps” (i.e., leakage from collimators as a function of longitudinal position along the ring) and collimation efficiency [4]. SixTrack also has a well-controlled floating-point arithmetic and fully reproducible results among many compilers and architectures [5].

Massive long-term tracking simulations are performed routinely using SixDesk [6], which is a running environment that is able to prepare, submit, manage, and post-process simulations using the CERN batch system [7] or through the LHC@Home platform for volunteering computing [8]. Often SixTrack lattices are prepared using MAD-X [9] models that are exported using the built-in converter.

The SixTrack project web page [10] contains links to the source code (see Section 5), documentation (see Section 4), and main references.

2 Features on the main branch

SixTrack has tracking maps for the following elements: drift (expanded [11] and full Hamiltonian [12, 13]), thin multipoles, orbit correctors, RF cavities, thick bends and quadrupoles (rigorously symplectic in 6D [14]), RF multipoles [15] (up to 4th order [16, 17]), crab cavities [18], 4D and 6D beam-beam models [19], e-lenses [20–22], and beam-beam compensation wires [23–25].

SixTrack is also able to compute one-turn maps to arbitrary order thanks to a differential algebra package. Optics functions are calculated in 5D by using a matrix formalism, and in 6D by using a differential algebra package and the Mais-Ripken formalism [26]. Optics functions are used to generate initial conditions for the tracked particles and parameters for the beam-beam lenses. In addition, tracking data can be post-processed in linear action-angle space for getting distance in phase space for Lyapounov exponent analysis, average actions, smear and harmonic analysis [27] using a modified version of the Plato library [28] and more recently NAFF [29].

Generic dynamic effects are introduced by the DYNK module, which allows applying arbitrary functions to a large fraction of element attributes [30, 31]. This feature has been successfully used to

https://doi.org/10.23732/CYRCP-2018-002.11.

model failure scenarios and energy error in crab cavities in the context of HL-LHC studies by applying phase modulation [32]. Generic particle exchange protocols using TCP/IP port (Fluka Coupling) and Linux pipe (BDEX module) can be used to interface tracking with external codes (e.g., FLUKA [33,34]).

Scattering models and aperture checks are implemented in a separate branch [4], which is planned to be merged in one of the future SixTrack version, similarly to an extension of the 6D formalism to track different ion species with arbitrary charge state relative to the same synchronous reference particle [35]. These code branches are generally used for collimation studies in the LHC, HL-LHC, and SPS [2].

The initial sample of particles to be tracked can be either generated in SixTrack via scans in linear action-angle space and energy offset or read from a file. The output of the simulation consists of turn-by-turn data as well as the one-turn map (in the form of closed orbit, tunes, and Mais-Ripken normalization matrix), in addition to the output of the post-processing analysis on Lyapounov pairs. Moreover, it is possible to dump particle coordinates or distribution moments (DUMP module) in arbitrary machine elements and file names. After simulation, output files can be packed in zip files (ZIPF module) to simplify data retrieval from remote running environments [27].

3 Branches

Compilation flags allow to compile SixTrack with specific functions or to choose the size of the allocated memory. In addition, few branches of the code exist, implementing additional functionalities. Not all functionalities enabled by the compilation flags and/or in the branches are mutually compatible, which is the reason why they are kept as such. This is non-ideal for those functionalities that could be merged offering a richer simulation setup choice. A development programme to merge the following functionalities has started:

Collimation : This option integrates scattering routines, management of collimator geometries, aperture checks, tracking of multiple sets of 64 particles at the same time; however, it breaks the post-processing routines for dynamic aperture and numerical reproducibility. Unlike other branches, this one is implemented as compilation flag, and is maintained in the same repository and updated together with the main code.

FLUKA coupling : This branch offers the possibility of a mutual exchange of particles with FLUKA [33, 34] through a TCP/IP socket during tracking; hence, the dynamics in the accelerator lattice can be coupled with the accurate scattering engine of FLUKA, allowing functionalities ranging from those similar to that of the collimation flag up to full on-line energy calculations; this branch comes with the addition of an on-line aperture checking, an early version of the DYNK and DUMP modules (afterwards imported into the main branch of SixTrack), the latter being a module for simple statistics on the tracked beam, and a generalised external engine for beam sampling.

Multiple ion species : This is a sub branch of the FLUKA coupling branch, in which the particle coordinates and tracking routines are extended to correctly compute the trajectories of particle with different mass and charge with respect to those of the reference particle.

4 Documentation

SixTrack documentation includes a user manual, a physics manual published as a draft in the SixTrack website, a developer manual, and a developer Wiki. The user manual [36] contains instructions on how to setup the simulation and how the quantities in the input files relate to the underlying physical model. It always refers to the latest production version of the code. A development version of the manual [37], synchronised with the latest commit in the master branch, is published in parallel. The physics manual [38] presents a short, but complete and self-contained, exposition of the physical models and mathematical methods used by SixTrack. The developer manual [39] guides the reader in the development platform and contribution strategy, as well as giving practical guidance on how to build, test, and submit patches (“pull request” in `GitHub` jargon) to the main branch. The developer Wiki [40] contains a collection of notes written by the developers explaining the meaning and the usage of the internal variables and subroutines.

5 Code Developments and testing

SixTrack is developed by a community of expert users who implement and document new physics, and maintainers who coordinate the parallel developments, advise and review the code of the developers, schedule releases, and maintain the infrastructures.

The source code and development platform is hosted on GitHub [41]. The old commit history from the previous SVN repository, which has been discontinued, has been preserved. Developers are encouraged to contribute by forking the project and submit pull requests. The tool has greatly improved support for working with branches compared to SVN.

SixTrack code can build on several OS (Linux, Windows, OSX and recently FreeBSD, NetBSD and Android) with different compilers (gfortran, Intel Fortran compiler, NAG, Lahey, PGI) using a CMAKE build system [27].

A test suite is available and covers about 45% of the code in the main branch. Results of each test are identical in all platforms. The test suite runs through `CTest` and results are reported on `CDash` [42]. Tests using the collimation flags and other branches are not yet integrated in the test suites.

6 Conclusions and Outlook

SixTrack is a code with a long history of development, intimately linked to the optimization of the LHC. The main assets are the computational speed, thanks to a careful optimization of the main tracking loop, and the integration in the complex simulation environment for LHC, relying on the high-performance computing resources available at CERN and on the LHC@Home BOINC project. In the past few years, a large set of new features to cover the needs of new projects (e.g. HL-LHC and FCC) have been implemented thanks to the work of many developers, and an increased level of support in the code management.

Future developments aim at reducing incompatible branches by absorbing functionality in the main branch. This should leave only a small set of selectable features which introduce strong simulation speed or memory penalties for standard simulations. The testing platform should include more test cases to increase the tested code coverage. The physics and build manuals are in the pipeline for completion and publication. The following additional physics models are currently under development: extend DYNK to cover beam-beam lenses, implement an on-line aperture checking, extend the initial conditions module to cover distributions (BDIST module), complete and test solenoid, implement generic high order RF-multipoles, and introduce radiation effects.

SixTrackLib [43] is a new development effort to provide a library which exposes SixTrack tracking maps to into external programs. It is designed to support CPU and GPU through CUDA and OpenCL and it is written in portable C99. The long-term goal is to replace the internal Fortran tracking routines with those of SixTrackLib in order to take advantage of GPU resources on LHC@Home volunteer machines and HPC clusters.

References

[1] F. Schmidt *et al.*, “SixTrack Version 4.2.16 Single Particle Tracking Code Treating Transverse Motion with Synchrotron Oscillations in a Symplectic Manner,” CERN/SL/94-56, 2012.

[2] R. Bruce *et al.*, “Status of Sixtrack with collimation”, these proceedings.

[3] A.F. Wrulich, “RACETRACK: a computer code for the nonlinear particle motion in accelerators,” DESY 84-026, 1984.

[4] G. Demolaize, *et al.*, “A New Version of SixTrack with Collimation and Aperture interfaces”, in *PAC’05 Proceedings*, USA, 2005.

[5] McIntosh E. and Wagner A., “CERN Modular Physics Screensaver or Using Spare CPU Cycles of CERN’s Desktop PCs”, in *Proceedings of 14th International Conference on Computing in High-Energy and Nuclear Physics*, CERN Geneva 2005.

[6] E. McIntosh, R. De Maria, “The SixDesk Run Environment for SixTrack,” CERN-ATS-Note-2012-089 TECH, 2012.

[7] CERN Batch service, http://information-technology.web.cern.ch/services/batch

[8] M. Giovannozzi *et al.*, “LHC@Home: A volunteer computing system for massive numerical simulations of beam dynamics and high energy physics events” in *IPAC’12 Proceedings*, New Orleans, LA, USA, May 2012, paper MOPPD061.

[9] MAD-X Project website, `http://cern.ch/mad`

[10] SixTrack Project website, `http://cern.ch/sixtrack`

[11] G. Ripken, F. Schmidt, “A symplectic six-dimensional thin-lens formalism for tracking,” DESY 95-63 and CERN/SL/95-12(AP), 1995.

[12] K. Heinemann, G. Ripken, F. Schmidt, “Construction of nonlinear symplectic six-dimensional thin-lens maps by exponentiation,” DESY 95-189, 1995.

[13] M. Fjellstrom, “Particle Tracking in Circular Accelerators Using the Exact Hamiltonian in SixTrack”, CERN-THESIS-2013-248, 2013.

[14] D.P. Barber, *et al.*, “A non-linear canonical formalism for the coupled synchro-betatron motion of protons with arbitrary energy,” DESY 87-36, 1987.

[15] A. Latina, R. De Maria, “RF multipole implementation,” CERN-ATS-Note-2012-088 TECH, 2012.

[16] D.R. Brett, *et al.*, “Accurate crab cavity modeling for the high luminosity Large Hadron Collider,” Phys. Rev. ST Accel. Beams 17, 104001.

[17] J. Barranco Garcia, *et al.*, “Long term dynamics of the high luminosity Large Hadron Collider with crab cavities,” Phys. Rev. Accel. Beams 19, 101003.

[18] R. Calaga *et al.*, “LHC crab-cavity aspects and strategy”, in *IPAC’10 Proceedings*, Japan, 2010.

[19] L.H.A. Leunissen, *et al.*, “6D beam-beam kick including coupled motion,” Phys. Rev. ST Accel. Beams 3 (2000) 124002.

[20] V. Previtali, *et al.*, “Numerical Simulations of a Hollow Electron Lens as a Scraping Device for the LHC,” in *IPAC’13 Proceedings*, Shanghai, China, May 2013, paper MOPWO044.

[21] M. Fitterer, *et al.*, “Electron lens news”, 3rd General SixTrack Meeting, 7th December 2016, `https://indico.cern.ch/event/590257/`.

[22] M. Fitterer, *et al.*, “Implementation of Hollow Electron Lenses in SixTrack and first simulation results for the LHC,” *IPAC’17 Proceedings*, Copenhagen, Denmark, May 2017, paper THPAB041.

[23] A. Patapenka, “Wire implementation in SIXTRACK”, Beam-Beam and Luminosity Studies meeting, 3 August 2015, `https://indico.cern.ch/event/436670/`.

[24] A. Patapenka, “Wire model in Sixtrack - simulation results,” in *Simulations and Measurements of Long Range Beam-Beam Effects in the LHC,* 2015, `https://indico.cern.ch/event/456856/`.

[25] A. Patapenka, “First order transport map of wire element,” in *3rd General SixTrack Meeting, 7 December 2016,* `https://indico.cern.ch/event/590257/`.

[26] H. Mais and G. Ripken, “Theory of coupled synchro-betatron oscillations”, DESY-M-82-05, DESY, 1982.

[27] K. Sjobak, *et al.*, “New Features of the 2017 SixTrack release”, in *IPAC’17 Proceedings*, Copenhagen, Denmark, May 2017, paper THPAB047.

[28] M. Giovannozzi, *et al.*, “PLATO: a program library for the analysis of nonlinear betatronic motion”,

Nucl. Instrum. and Methods A, vol. 388, 1996.

[29] S. Kostoglou, *et al.*, "Development of Computational Tools for Noise Studies in the LHC", in *IPAC'17 Proceedings*, Copenhagen, Denmark, May 2017, paper THPAB044, https://doi.org/10.18429/JACoW-IPAC2017-THPAB044.

[30] K. Sjobak, *et al.*, "General functionality for turn-dependent element properties in SixTrack", in *IPAC'15 Proceedings*, Richmond, VA, USA, May 2015, paper MOPJE069.

[31] K. Sjobak, "Dynamic simulations in Sixtrack", these proceedings.

[32] S.J. Wretborn, *et al.*, "Study of off-momentum losses at the start of the ramp in the Large Hadron Collider", September 6th, 2017, CERN-ACC-NOTE-2017-0065, http://cds.cern.ch/record/2298696.

[33] A. Mereghetti, *et al.*, "SixTrack-Fluka Active Coupling for the Upgrade of the SPS Scrapers," in *IPAC'13 Proceedings*, Shanghai, China, May 2013, paper WEPEA064.

[34] R.B. Appleby, *et al.*, "Fluka Coupling for SPS Scrapers", these proceedings.

[35] P. Hermes, *et al.*, "Symplectic Tracking of Multi-Isotopic Heavy-Ion Beams in SixTrack," in *IPAC'16 Proceedings*, Busan, Korea, May 2016, paper TUPMW015.

[36] F. Schmidt, *et al.*, "SixTrack User Manual,", http://sixtrack.web.cern.ch/SixTrack/doc/manual/six.html, 2017.

[37] F. Schmidt, *et al.*, "Sixtrack Physics Manual (Dev)," http://sixtrack.web.cern.ch/SixTrack/doc/manual_dev/six.pdf, 2017.

[38] R. De Maria, *et al.*, "Sixtrack Physics Manual (DRAFT)," http://sixtrack.web.cern.ch/SixTrack/doc/physics_manual/sixphys.pdf.

[39] K. Sjobak, J. Molson, "Compiling, building, and testing SixTrack,", http://sixtrack.web.cern.ch/SixTrack/doc/building_sixtrack/building_sixtrack.pdf.

[40] "SixTrack Developer Wiki," http://twiki.cern.ch/twiki/bin/view/LHCAtHome/SixTrack.

[41] SixTrack source code repository, http://github.com/SixTrack/SixTrack.

[42] CDash website, https://abp-cdash.web.cern.ch/abp-cdash/index.php?project=SixTrack.

[43] SixTrackLib source code repository, http://github.com/SixTrack/SixTrackLib.