RECENT IMPROVEMENTS OF PLACET*

A. Latina†, P. Eliasson, L. Neukermans, G. Rumolo, D. Schulte (CERN, Geneva, Switzerland)

Abstract

The tracking code PLACET[1] simulates beam transport and orbit correction in linear colliders, from the damping ring to the interaction point and beyond. It allows for the investigation of single- and multi-bunch effects, implements normal RF cavities as well as transfer structures specific to CLIC. It is a fully programmable and modular software, thanks to a Tcl/Tk interface and external modules based on shared libraries. Recent improvements of the code include the possibility to simulate bunch compressors, ground motion as well as to use parallel computer systems and an extended programmable interface toward mathematical packages (such as Octave).

SIMULATION AND ALGORITHMS

Longitudinal Motion

A module to include the longitudinal motion in PLACET has been developed in C++. The following elements have been implemented so far: DRIFT, QUADRUPOLE, DIPOLE, SBEND, BPM and MULTIPOLE. All linear elements are treated as thick elements according to the second order matrix formalism in the full six-dimensional phase space, but thin-lens formalism is also supported. In SBEND, the emission of Synchrotron Radiation is taken into account. This module was created as an independent loadable program, which communicates with PLACET via memory buffers, in order to provide more flexibility, as well as to enable the development of a version suitable for clusters of computers, as will be described in the next subsection. From the user's point of view, the implementation is transparent and the six-dimensional tracking can be enabled (or disabled) at any time by inserting the commands "Begin6dTracking" and "End6dTracking" in the lattice description. This module is already operational and has been used for simulations of both the CLIC and the ILC Bunch Compressors.

Parallel Tracking with Clusters of Computers

Two versions of the Longitudinal Motion tracking module were created: one, described in the previous subsection, was developed for single CPU machines, the other, described here, was designed and optimized for clusters of computers. The implementation of this module took advantage of the Message Passing Interface (MPI) library, a standard protocol for inter-process communications which has been designed for high performance computation on massively parallel machines and on workstation clusters. Parallelizing the code written for the single CPU version was straightforward because the code treats the beam like an array of independent particles (collective effects are not taken into account yet). First the particles are equally distributes between the CPU's of the cluster, then tracking is performed by each machine and finally the results are gathered together and saved. In order to give an idea of the increment in speed, one should consider that a tracking executed on a cluster composed of N CPU's will benefit of a speed up by a factor of N. Similarly to the single processor version, the user can enable and disable this module by inserting the commands "MPI_Begin6dTracking" and "MPI_End6dTracking" at any point in the lattice description.

Emittance/Luminosity Tuning Bumps

A module for simulation of tuning bumps has been developed. The user must define a number of *knobs* and a *tuning signal*, before the optimization can start. Knobs can modify quadrupole position, dipole strength, etc. and the tuning signal, which will be optimized, can be drawn from the emittance (either locally or at the end of the linac), from the luminosity, or from any merit function the user defines. The optimization of the settings is carried out by first taking a number of measurements for different knob settings,

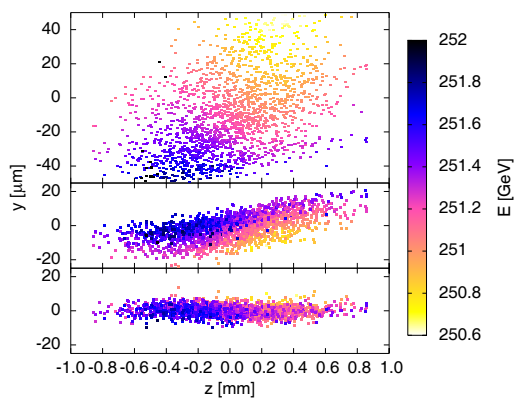

Figure 1: Example of tuning bumps in the ILC linac. The beam is plotted in the $z-y$ plane, with color corresponding to energy, for different correction methods. Top plot shows beam after one-to-one correction. Middle plot shows the beam after Dispersion Free Steering (DFS), while the bottom plot shows the beam after DFS and optimization of emittance tuning bumps.

* This work is supported by the Commission of the European Communities under the 6th Framework Programme "Structuring the European Research Area", contract number RIDS-011899.

† andrea.latina@cern.ch

then a parabolic fit is performed on the obtained values which gives an estimate of the optimal knob settings. In this way one knob after the other can be optimized. The optimization of all knobs can be iterated. Results of the emittance tuning bumps in the ILC linac are presented in [2] and shown in Fig. 1.

Orbit Correction Algorithms

A new module introducing the MICADO orbit correction algorithm has been added to PLACET's existent Linear System Solver (LS) correction algorithm. This module is now operational and has been tested in the feedback loop of a dynamic simulation of a CLIC linac which is misaligned due to the ground motion. The results are in accordance with expectations.

The PLACET Orbit Correction package was further extended by introducing an orbit correction module based on the Kalman Filter. The Kalman Filter descends from the digital control theory formalism and consists of a set of mathematical equations that provides an efficient computational tool to estimate the state of a process. The Kalman Filter is very powerful in several aspects: it supports estimations of past, present, and even future states, it takes into account the noise in the measurement vectors and it does so even when the precise nature of the modeled system is unknown. In our implementation, the Kalman Filter is used to support the standard LS algorithm that was already present in the code. More precisely, the vector of measured values (i.e. the state of the machine, in this case the vector with the BPM readings) which was feeding the existing LS algorithm, is now replaced by the Kalman Filter's *prediction* for it (see Fig. 2), which takes into account both the history of the system and the noise affecting the measurements. The convergence of this improved KF+LS algorithm seems to be faster than the plain LS implementation, as it is shown in Fig. 3.

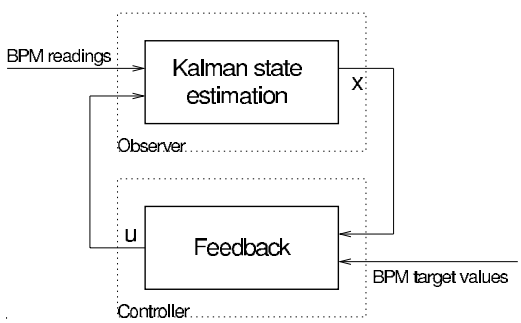

Figure 2: Block diagram of the Kalman Filter orbit correction feedback loop.

Ground Motion and LiCAS Interface

A program to include ground motion simulation in PLACET has been developed. This ground motion package is based on a model provided by Andrei Seryi[3] and

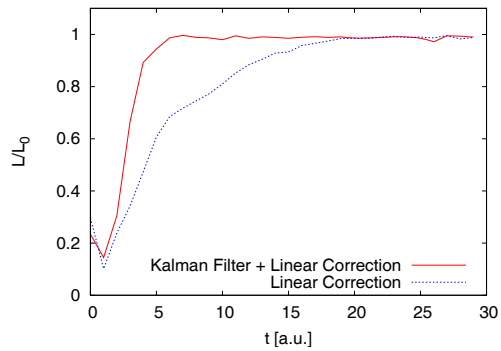

Figure 3: Kalman Filter+Linear Solver vs standard Linear algorithm, recovering the luminosity of a CLIC BDS subject to ground motion. Notice that KF+LS convergence is faster than plain LS.

calculates the misalignment along a given part of the machine. All the data exchange with PLACET is performed via files. The program needs two files: one containing the information about the amplitudes, frequencies and wavelength of the ground motion modes, and another that specifies the longitudinal locations of the points at which the misalignment must be calculated. Based on these two files, the program outputs the misalignments that PLACET reads to misalign the beamline. A seed for the random number generator can be specified to ensure that calls with different positions along the same machine yield consistent motions. In order to be efficient, the program does not calculate the misalignment at a single time but rather at a number of steps in time.

A program interfacing PLACET to the "Linear Collider Alignment and Survey" project (LiCAS, see [4]) has been also developed. The LiCAS group is developing a measurement system to provide an optical metrology system for the survey and alignment of a linear collider. It exploits a novel combination of optical techniques to overcome the limitations of traditional open air survey. This technique should increase the range and precision to meet the demanding alignment tolerances of future linear colliders.

Collimator Wake Fields

A module for simulating single bunch collimator wake fields has been constructed and implemented in PLACET. For this purpose, a new element-type, COLLIMATOR, has been added to the set of PLACET lattice definition commands. In order to use it, the user must provide the parameters describing the material and the geometry of the collimator. Based on these numbers, the module can determine the working regime of the collimator, i.e. inductive, diffractive, or intermediate for the geometric wake, and long- or short-range (DC or AC-conductivity) for the resistive wall wake. When tracking, the wake field kick is calculated and applied to the particle. The code evaluates the kick on each particle of the bunch as a function of its longitudinal posi-

tion (and transverse, if the collimator is flat and there is a quadrupole component of the wake) and applies it. A more detailed description is presented in [6].

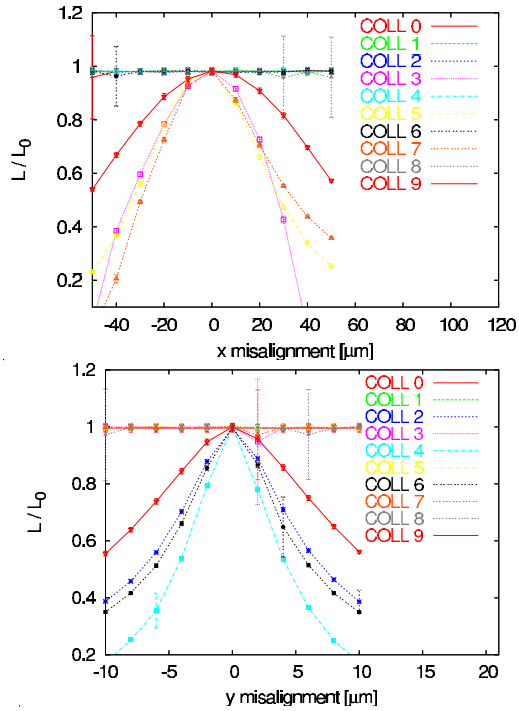

Figure 4: Luminosity reduction curves due to horizontal and vertical misalignment of collimators in the CLIC BDS.

Beam Halo and Tail Generator

In order to help for halo and collimation studies, some new features have been implemented in PLACET [7]. A beam-gas scattering fast generator has been interfaced. It simulates elastic and inelastic collisions with beam-gas atoms. Scattered particles are stacked into a secondary beam structure and tracked trough the beamline. A new geometrical description of the element aperture allows to localize hits of secondaries in the elements. These hits are now described as hard collimation - where the particle is lost - or soft collimation where particles interacts with material trough multiple scattering interactions. A photon tracking module has been also implemented. It allows to estimate synchrotron radiation losses trough the beam line.

CONVERSION TOOLS

In order to benchmark PLACET against other tracking codes, two conversion tools have been created: `mad2placet`, which fills the conversion from the MAD lattice description language into PLACET, and `xsif2placet`, which converts into PLACET the Standard Input Format (SIF). The SIF was created in 1984 in respondence to the need for a more user-friendly *lingua franca* for the accelerator world[8]. Our converter uses the LIBXSIF library, a standalone package for parsing the extended Standard Input Format decks in use at SLAC[9]. Besides PLACET, this format is also adopted by programs such as LIAR, MAD, DIMAD and TRANSPORT.

INTERFACE TOWARD OCTAVE

An interface toward the mathematical package Octave[5] is under development. Octave is a high-level interactive language, primarily intended for numerical computations, that is mostly compatible with Matlab©. Octave can do arithmetic for real and complex scalars and matrices, solve sets of nonlinear algebraic equations, integrate functions over finite and infinite intervals, integrate systems of differential equations; it includes also a set of optimization algorithms and a toolbox for controlling dynamic systems. The integration proceeded through an *embedding* of Octave into PLACET: the original Octave's command set has been enriched with special commands controlling the PLACET *state machine* while, on the other hand, the Octave's workspace and symbols have been extended to *see* the PLACET's data structures in memory. This integration will allow the PLACET users to process, elaborate and display the results of their simulations directly within the PLACET/Octave environment, as well as to control the simulation programme itself using the powerful Octave's statements and optimization tools.

ACCESSIBILITY

PLACET is free software and it is available on the Internet from:
`http://savannah.cern.ch/projects/placet`

REFERENCES

[1] D. Schulte et al. CERN/PS 2001-028 (AE) and PAC 2001

[2] P. Eliasson, D. Schulte *Dispersion Free Steering and Emittance Tuning Bumps in the ILC Linac*, EUROTeV-Report-2005-021

[3] A. Seryi. Private communication.

[4] G. Grzelak et al. Private communication. See also `http://www-pnp.physics.ox.ac.uk/~licas/`

[5] John W. Eaton et al. See `http://www.octave.org`

[6] G. Rumolo et al. *Effects of wake fields in the CLIC BDS*, These proceedings.

[7] H. Burkhardt, L. Neukermans *Halo and tail generation study for linear collider*, These proceedings.

[8] D.C. Carey et al. *A Standard Input Language for Particle Beam and Accelerator Computer Programs*, Proceedings of the 1984 Summer Study on the Design and Utilization of the Superconducting Super Collider, Snowmass, Colorado, 1984.

[9] P. Tenenbaum *LIBXSIF, A STANDALONE LIBRARY FOR PARSING THE STANDARD INPUT FORMAT*, Proceedings of the 2001 Particle Accelerator Conference, p. 3093-3095, Chicago