PARTICLE TRACKING AND BUNCH POPULATION IN TraFiC^4 2.0

Andreas C. Kabel*
Stanford Linear Accelerator Center, Stanford, CA, USA
andreas.kabel@slac.stanford.edu

Abstract

Coherent Synchrotron Radiation (CSR) plays an important role in the design of accelerator components with high peak currents and small bending radii, such as magnetic bunch compressors, wigglers, and compact storage rings. The code TraFiC4 has been developed to design such elements[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12] ; it simulates CSR effects from first principles. We present a re-write of the tracking and user interface components of TraFiC^4 . Extensions and corrections include: expanded input language; generalized bunch populations (rectangular, Gaussian, user-specified function); new element types; truly three-dimensional dynamics (i. e., the restriction to a single plane of motion has been abandoned), vastly expanded documentation; documented C++ class interface; and improved dynamic load-balancing for parallel computers.

FULLY THREE-DIMENSIONAL TREATMENT

Generalized Local Coordinates

TraFiC^4 now handles fully three-dimensional problems; there is no limitation to one plane of movement. This makes it necessary to choose a more general system of local coordinates: while TraFiC^4 does all of its tracking in laboratory coordinates, as it needs to store the history of all particles to calculated the retarded fields, accelerator physics is usually done in a co-moving frame.

As a local system, we choose the Frenet coordinate system associated with an orbit particle's trajectory $\vec{r}_0(s)$, where s is the arc length. The co-moving frame is spanned by $\vec{r}_0'(s)$, $\vec{r}_0''(s)$, and $\vec{r}_0'(s) \times \vec{r}_0''(s)$; the associated normalized vectors are $\vec{t}(s)$, $\vec{n}(s)$, and $\vec{b}(s)$. Given another particle trajectory $\vec{r}(s(t))$, parametrized by the lab time t, we find that particle's local coordinates x, y, l by $\vec{r}(s(t)) = x\vec{n}(s(t)+l) + y\vec{b}(s(t)+l)$. Note that this decomposition is not unique, as there might be several l for which $\vec{t}(s_0(t)+l) \cdot (\vec{r}(s(t)) - \vec{r}(s_0(t)+l)) = 0$. TraFiC^4 starts looking around $l = 0$, however.

Note that this generalized prescription leads to some unfamiliar effects, such as x and y coordinate flipping their sign when the curvature does or switching roles when a sideways bend turns into an upward bend.

Also note that the prescription is not unique on drifts, as $\vec{r}'' = 0$. We use the parallel-transported $\vec{n}$ from the last curved section in these cases; by convention, we start our beamline with $\vec{t}(0) \parallel x_{Lab}$ and $\vec{n}(0) \parallel y_{Lab}$.

Generalized Dipoles

It is now possible to rotate Bending magnets around the axis of the incoming particle by an arbitrary angle. This makes TraFiC^4 fully three-dimensional, as the movement of the orbit particle is not restricted to a plane any more.

Furthermore, a dipole can be tilted upward and downward, i. e., rotated around the curvature vector. This turns the trajectory of the orbit from a circular arc into a helix segment; this generalization might be useful for studying the behavior of bunches moved out of the radiation cone of their own CSR by such an arrangement.

Also, the magnet's entry and exit faces may be rotated away from a sector-bend setup; the (de)focusing effects induced by fringe fields components occurring by tilting the magnet and its faces in the x and y phase-space planes, resp., is modeled by sandwiching the dipole between thin-lens (de)focusing elements. Neighboring wedge elements of the same strength, but opposite signs, are canceled automatically. Neighboring dipoles need to have compatible exit and entry faces; otherwise, an arbitrarily small drift space in between is required.

OBSERVER GRIDS

Sometimes it is of interest to calculate the fields due to CSR not only within the bunch, but at observation points far away from the bunch. TraFiC^4 now allows to specify such observation points. It is most convenient to do that with reference to the beam-line coordinates. A new beamline element type `ObserverGrid` was introduced for this purpose; it has parameters n_t, t_{min}, t_{max};n_s, s_{min}, s_{max}; n_x, x_{min}, x_{max}; n_y, y_{min}, y_{max}; $\beta_s, \beta_x, \beta_y$. It will create four-dimensional lattice of size n_t, n_s, n_x, n_y, spanning the spacetime interval $[t_{min}, t_{max}] \otimes [s_{min}, s_{max}] \otimes [x_{min}, x_{max}] \otimes [y_{min}, y_{max}]$, where t refers to the laboratory time and s, x, y refer to the local tangential, radial, and transverse direction, resp. The observation grid moves with a speed of $c\vec{\beta}$. The quantities calculated are the total force per charge $\vec{E} + c\vec{\beta} \times \vec{B}$ and the energy change per charge $\vec{E} \cdot \vec{\beta}$. An arbitrary number of `ObserverGrids` can be defined.

Fig. and show field profiles over time and transverse (i. e., perpendicular to the curvature of the magnet) coordinates in the middle of a dipole of curvature $1/m$ and length $2m$. The 100×100 grid was specified by writing the beamline as

* Work supported by Department of Energy contract DE–AC03–76SF00515.

```
param Beamline=Group(Bend(1., 1.),
   ObserverGrid(100, -bunchlength, 5.*bunchlength,
    1, 0., 0.,
    1, 0., 0.,
    100, -bunchlength, bunchlength,
    0. , 0., 0.
    ),
    Bend(1., 1.)
  ));
```

The first field profile is due to a Gaussian bunch, the second is caused by a rectangular bunch, softened by a Fermi-Dirac distribution.

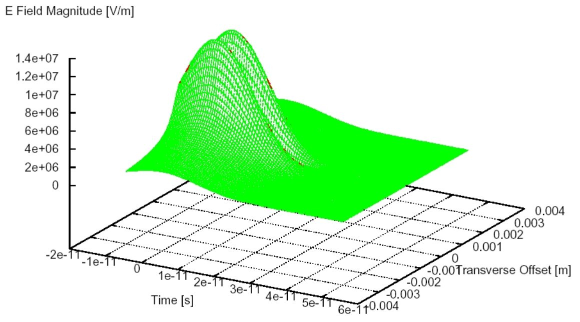

Figure 1: Field vs. time, transverse offset plot Beam parameters: $E = 150MeV, Q = 1.0nC, \sigma_s = 3mm, \sigma_y = \sigma_s/10$

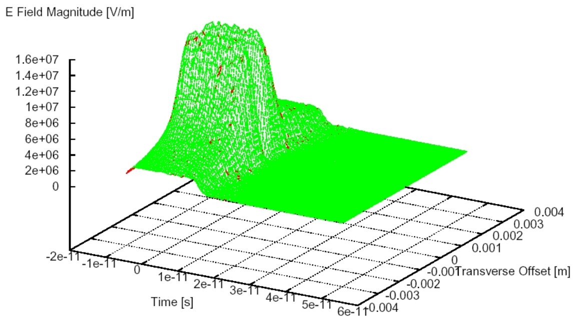

Figure 2: Same as , but for an almost rectangular charge distribution (softened by a Fermi-Dirac distribution with T=1/15)

BUNCH SETUP

One of the biggest hurdles to successfully use TraFiC^4 in its previous versions was the awkward way of setting up bunches. TraFiC^4 knows three kinds of bunches: *generating* bunches, comprising weighted, smoothed-out macro particles which generate the fields and may or may not move under the influence of their own fields, depending on whether or not the user expects these effects to be important; *optical* bunches, which do not generate fields, but feel the perturbations of the fields caused by the generating bunch(es), and *sampling* bunches, which sample the fields of the generating bunches, but are not influenced by them (they can be viewed as co-moving `ObserverGrids`).

Generating bunches consist of one-, two-, or three-dimensional extended Gaussian charge distributions whose centroids are tracked according to the magnetic lattice and possibly the bunch's fields. All other bunches comprise point particles, which can bear a charge (or statistical weight), which will be used in calculating collective quantities such as dissipated power, rms values, Twiss parameters. As sampling and optical bunches do not contribute to the fields, their setup needs not coincide with the setup of the generating bunch(es), so they can be used to study the behavior of sub-ensembles of the bunch.

A bunch in TraFiC^4 is now set up by specifying a particle class (point, pencil, sheet, or cylinder) and a Generator. A Generator is a sequence of extendend phasespace vectors (extended meaning 6 phasespace coordinates + 1 statistical weight); pre-defined generators are cartesian Grids, quasi-random sequences, and input from a text file. Generators can be transformed into new Generators using a set of predefined functions (transform to beam ellipses with given Twiss parameters, Scale, Shift, transform to gaussian normal distribution), Selectors (first n particles from a generator, particles lying on either side of a hyperplane), and arbitrary user-defined functions. All transformations can be concatenated, allowing for high flexibility in populating the bunch.

SELF-CONSISTENT CALCULATIONS

TraFiC^4 has been augmented by a new algorithm for self-consistent calculations. The user now has the choice of two algorithms:

- A “pong” algorithm, in which two initially identical copies of the generating bunch correct their trajectories by applying the field of the respective other bunch. This mutual correction is repeated a user-defined number of times for each timestep. The deviation of the copies' trajectories give a rough estimate of the error made in the process. This algorithm, however, sometimes will converge only for ungainly small timesteps or high particle numbers [].

- A self-correcting algorithm, in which all fields of a bunch on itself in a given timestep are collected and subsequently applied at once to all particles. The trajectories are then corrected up to the end of the beamline, and the process repeats (again, the number of iterations is user-defined) or advances to the next timestep. This does not give an error estimate, but avoids having macroparticles of the generating and receiving bunch have a very small but finite distance (as generating and receiving bunch coincide).

OTHER NEW FEATURES

Memory Requirements

Memory requirements for TraFiC^4 have drastically reduced, as a number of quantities that had to be stored in the previous version are now computed on the fly and discarded. This was made possibly by an overall restructuring of the code. The asymptotic memory requirements are now `11⋆sizeof(double)+ sizeof(void⋆)` (=92 bytes on most machines) per particle and timestep, as opposed to 234 bytes in previous versions.

Load Balancing

The automatic load balancing in the multi-processing version has been improved. TraFiC^4 allows for the automatic distribution of particles to processors according to the measured speeds in previous timesteps. The load balancer will react to changes in speed according to a user-selectable inertia. Moreover, the load balancer associated with each bunch and `ObserverGrid` can learn from the speed behavior of calculations for bunches treated before, distributing particles accordingly.

Documentation and Source Code

The documentation of the user interface and tracking part has been vastly extended [13]. Also, the class structure of the tracking part has been extensively documented, which should make it possible for programmers to add their own element types or transformer functions [14, 15].

The source code for TraFiC^4 is available from [16]. TraFiC^4 requires a recent, ANSI-compliant C++ compiler, a Fortran 77 compiler, some components of the “boost” extension library [17], and an MPI multiprocessing library.

Acknowledgements

I wish to thank T. Limberg, H. Schlarb and F. Stulle for useful discussions. I also wish to thank M. Dohlus to make his new field calculation algorithm available to me.

REFERENCES

[1] A. Kabel. Coherent synchrotron radiation calculations using trafic4: Multi-processor simulations and optics scans. Presented at IEEE Particle Accelerator Conference (PAC2001), Chicago, Illinois, 18-22 Jun 2001.

[2] M. Borland, H. Braun, S. Doebert, L. Groening, and A. Kabel. Recent experiments on the effect of coherent synchrotron radiation on the electron beam of ctf ii. Presented at IEEE Particle Accelerator Conference (PAC2001), Chicago, Illinois, 18-22 Jun 2001.

[3] A. Kabel, M. Dohlus, and T. Limberg. Numerical calculation of coherent synchrotron radiation effects using TraFiC4. *Nucl. Instrum. Meth. A*, A455:185–189, 2000.

[4] A. Kabel, M. Dohlus, and T. Limberg. Numerical calculation of coherent synchrotron radiation effects. 1999. Invited talk given at the International Symposium on New Visions in Laser Beam Interactions: Fundamental Problems and Applications of Laser Compton Scattering, Tokyo, Japan, 11-15 Oct 1999.

[5] H. H. Braun, R. Corsini, L. Groening, F. Zhou, A. Kabel, T. Raubenheimer, R. Li, and T. Limberg. Coherent synchrotron radiation measurements in the CLIC test facility (CTF II). *eConf*, C000821:TH206, 2000.

[6] H. H. Braun, R. Corsini, L. Groening, F. Zhou, A. Kabel, T. Raubenheimer, R. Li, and T. Limberg. Emittance growth and energy loss due to coherent synchrotron radiation in a bunch compressor. *Phys. Rev. ST Accel. Beams*, 3:124402, 2000.

[7] M. Dohlus, A. Kabel, and T. Limberg. Efficient field calculation of 3D bunches on general trajectories. *Nucl. Instrum. Meth.*, A445:338–342, 2000.

[8] M. Dohlus, A. Kabel, and T. Limberg. Coherent effects of a macro bunch in an undulator. *Nucl. Instrum. Meth.*, A445:84–89, 2000.

[9] M. Dohlus, A. Kabel, and T. Limberg. Optimal beam optics in the TTF-FEL bunch compression sections: Minimizing the emittance growth. In *Proceedings of the IEEE Particle Accelerator Conference (PAC 99), New York, NY, 29 Mar - 2 Apr 1999*, 1999.

[10] M. Dohlus, A. Kabel, and T. Limberg. Uncorrelated emittance growth in the TTF-FEL bunch compression sections due to coherent synchrotron radiation and space charge effects. In *Proceedings of the 6th European Particle Accelerator Conference (EPAC 98), Stockholm, Sweden, 22-26 Jun 1998*, 1998. DESY-M-98-060l.

[11] M. Dohlus, A. Kabel, and T. Limberg. Design consequences of coherent synchrotron radiation beam dynamic effects on the TTF-FEL bunch compression system. In *Proceedings of the 19th International Conference on Free Electron Lasers, Beijing, China, 18-21 Aug 1997*, 1997. DESY-TESLA-FEL-97-06B, Oct. 1997.

[12] M. Dohlus, A. Kabel, and T. Limberg. Wake fields of a bunch on a general trajectory due to coherent synchrotron radiation. In *Proceedings of the 17th IEEE Particle Accelerator Conference (PAC 97): Accelerator Science, Technology and Applications, Vancouver, Canada, 12-16 May 1997*, 1997. DESY-M-97-10J.

[13] A. Kabel. A short guide to trafic4 2.0. 2003. http://www.slac.stanford.edu/ akabel/TraFiC4-2.0/doc.pdf.

[14] A. Kabel. Trafic4 2.0 – programmer’s manual. 2003. http://www.slac.stanford.edu/ akabel/TraFiC4-2.0/TraFiCdoc/latex/refman.pdf.

[15] A. Kabel. Trafic4 2.0 – programmer’s manual. 2003.

[16] http://www.slac.stanford.edu/ akabel/TraFiC4.

[17] http://www.boost.org.