Description of guineapig++, the C++ upgraded version of the GUINEA-PIG beam-beam simulation program

G. Le Meur, P. Bambade, Cécile Rimbault, F. Touze, F. Blampuy, D. Schulte

Description of *guineapig*++, the C++ upgraded version of the GUINEA-PIG beam-beam simulation program

G. Le Meur, P. Bambade, C. Rimbault, F. Touze
F. Blampuy (summer intern)

LAL, Univ. Paris-Sud, CNRS/IN2P3, Orsay, France

D. Schulte

CERN, Geneva, Switzerland

Abstract

The development of *guineapig*++, an object oriented version of the program GUINEA-PIG is presented. Used computer tools and methods are described. Improvements and new developments are detailed.

1 Introduction

GUINEA-PIG is a program to simulate beam-beam interactions in high-energy e^+e^- colliders, written in C language by D. Schulte [1] twelve years ago. It includes electromagnetic pinch and disruption effects, beamstrahlung, and secondary interactions leading to additional low-energy background particles such as pairs or hadrons. This program is a fundamental tool for R&D on future linear colliders.

In order to allow updating and introducing new functionalities needed for these challenging machines, an object-oriented C++ version of the program, *guineapig*++, has been developed. The object-oriented computer environment moreover allows benefiting from versatile computer tools for committing, versioning, distributing and documenting the code.

In Section 2, the main features of this development are described and in Section 3 the implementation of several new functionalities.

2 Computer environment of the program

2.1 Structure

The original algorithms from the C-version have been kept in *guineapig*++. Most of the C-structures were changed into C++ classes. Execution of the program is managed in terms of interactions between classes. The program is callable from any C++ framework : one has to instantiate an object of the class GUINEA and launch the method ‘run’ with the desired arguments (actually those of the command line of the C-version). There is also an executable to run in the same way as with the C-version. In both cases one has to prepare the input file ‘acc.dat’ with keywords as in the original version (including a few new keywords which were added). The main class GUINEA has, as private attributes, instances of others classes like FFT_SERVER, PARAMETERS, SWITCHES, BEAM, BEAM_PARAMETER, etc… It gives names to input or output files. The class GUINEA also sets the class GRID, which is the main class for field and beam-beam effect calculations. The C-method ‘simulate’ has been translated into a method of the class GUINEA. It achieves some initializations and launches the main loop of the program.

A novelty with respect to the C-version is the C++ inheritance of classes. As an example, the inheritance scheme for a ‘particle’ is illustrated in Figure 1. The general class ABSTRACT_PARTICLE defines all common attributes, such as the particle’s position, momentum and energy. The inherited classes carry specific properties needed in specific cases. The class PARTICLE_WITH_SPIN is for instance used in case of calculations with polarisation. An object of the class PARTICLE_WITH_SPIN is then a PARTICLE, but with a spin added. The object ‘PARTICLE_INTERFACE’ is used for input/output from or into a beam file, etc... All old C-structures have been reinterpreted in terms of such inter-related classes. In such a way, the program gains in clarity and versatility, and allows easier coding. Maintenance and implementation of new features are also facilitated.

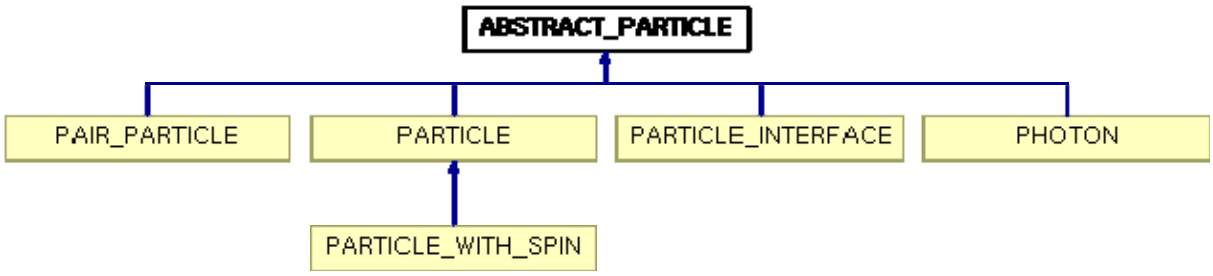

Figure 1: Inheritance scheme for 'particles' in *guineapig*++.

2.2 Updates

2.2.1 Random generation

In the original code, the random number generator assumed a 32-bit computer. A new algorithm, Haynes' algorithm [2], has been implemented in order to enable the code to be run on 64-bit machines. The most suitable random number generator is checked before the computation. In addition, the new keyword 'rndm_seed' allows one to choose a specific seed for the random generation. This is a useful feature to generate in parallel runs with independent random sequences.

2.2.2 Fast Fourier Transform

An interface with the latest version (3.1) of the high performance fast Fourier transform FFTW was implemented in order to accelerate the calculation of the beam field.

2.2.3 I/O interface

To enable large files to be manipulated, e.g. when running the program as part of a long-term simulation of the integrated collider performance (involving using input from and providing output to other software tools), the I/O functions have been strictly separated from the algorithms. Specific classes were thus designed in order to 'abstract' the I/O functions. New file formats can hence easily be plugged in. For the moment, the present ASCII format is however kept.

2.3 Performance and validation [3]

The performance of *guineapig*++ in terms of computing time was studied and shown to be similar to that of the C-version. The results obtained were also compared. An illustration of the agreement obtained is shown in Figures 2 for the horizontal beam divergence and for the transverse momenta of the secondary incoherent e^+e^- pairs produced during interactions.

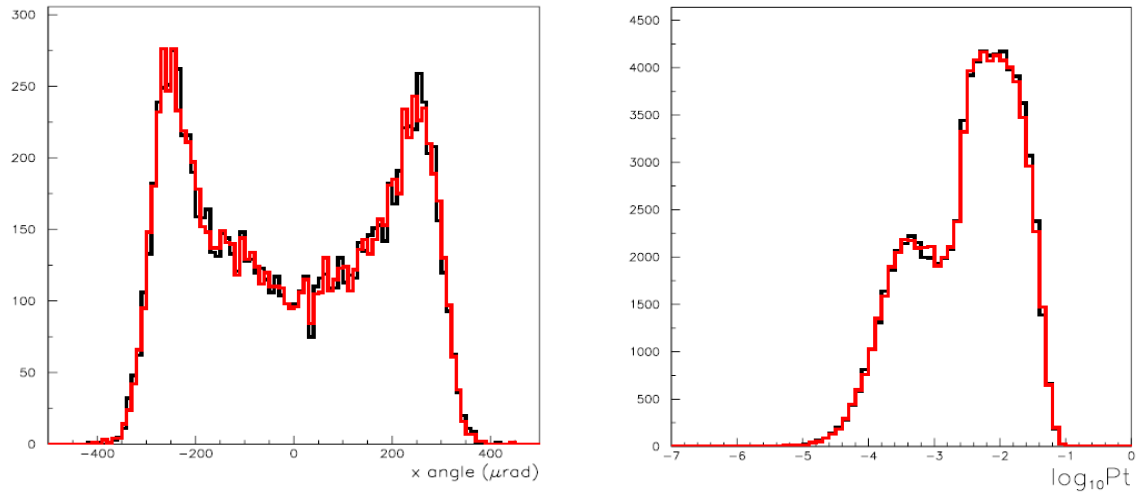

Figure 2: Comparison in *guineapig*++ (red) and in the C-version of GUINEA-PIG (black) of (a) horizontal beam particle angles and (b) transverse momenta of secondary incoherent e^+e^- pairs, after a full beam-beam interaction with Nominal ILC beam parameters [3].

2.4 Maintenance and distribution

2.4.1 Configuration

The configuration is achieved through the tool *CMT*[1] (*configuration tool management*). This tool was first developed at LAL and is used by several High Energy Physics experiments, in particular the ATLAS collaboration. It allows one to compile the whole software package without elaborating any makefile. A 'requirements' file specifies all the information needed by CMT to build a virtual makefile, and subsequently compile and link, in particular links to the FFTW library and information to recognize the current machine.

2.4.2 Versioning

The evolution of the package is managed through the use of the collaborating version manager free software *Subversion*[2] (SVN). Taking into account the experience of CVS, this versatile computer tool is designed for collaborating work, through a web interface. Its efficiency is much improved by the use of the *TRAC* interface (see below).

2.4.3 Documentation

All the information concerning development, documentation, bug reporting etc… is provided through a web interface, thanks to the tool *TRAC*[3], which is conveniently interfaced to *Subversion*, integrating also a wiki and easy reporting facilities via tickets.

1 http://www.cmt.site.org
2 http://subversion.tigris.org
3 http://trac.edgewall.org

Figure 3 shows the home page of the *guineapig*++ TRAC interface. The web address shown is: https://trac.lal.in2p3.fr/GuineaPig. The software *doxygen* provides a documentation of the classes of the whole package.

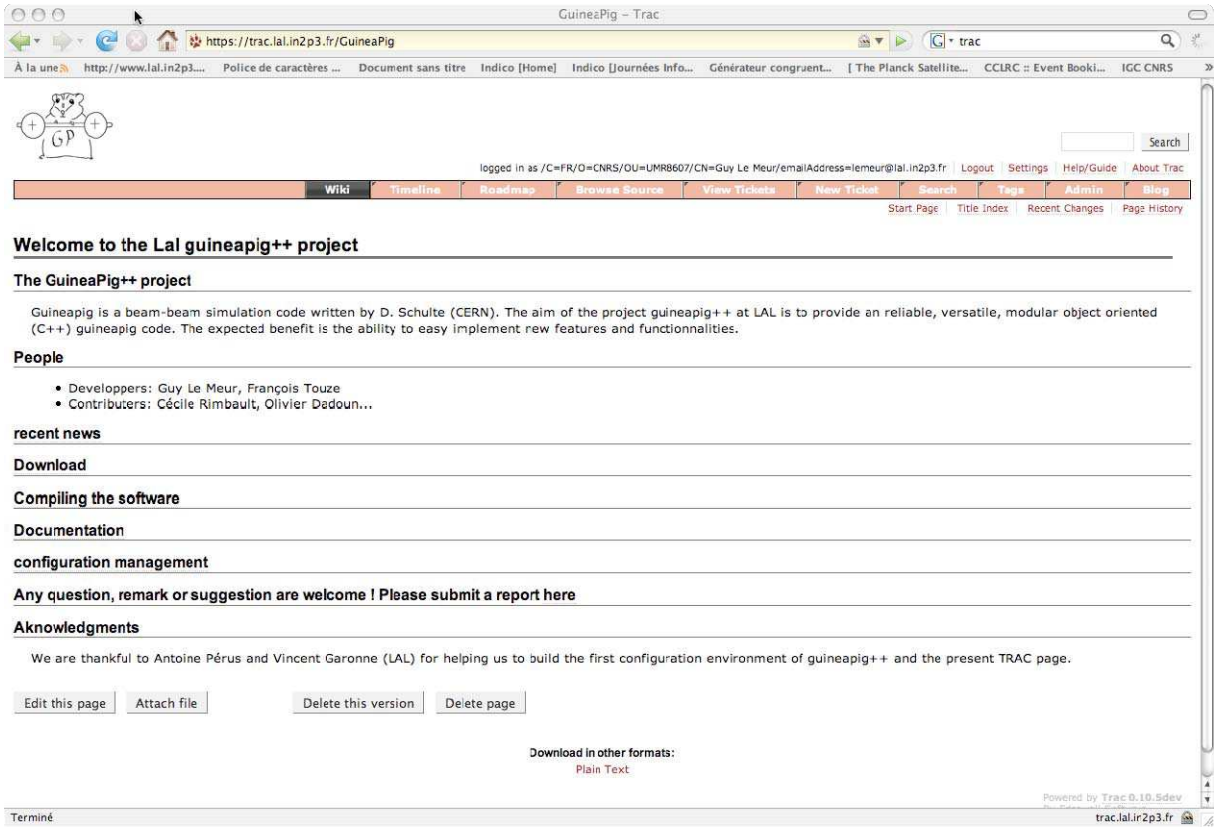

Figure 3: TRAC interface for *guineapig*++.

3 New developments

3.1 Bhabha events

The keyword 'do_bhabhas' allows applying beam-beam space charge effects to a sample of Bhabha events produced with an external generator. This involves tracking the scattered particles in the field of the colliding bunch after taking into account modified kinematics from both beamstrahlung radiation and beam-beam deflections of the initial state particles. This is important to quantify changes in counting rates in the experimental acceptance used for precision luminosity measurements. This development and the corresponding studies are described in [4] and [5].

3.2 Automatic grid adjustment

In the program, the electric forces on particles are computed resolving a Poisson equation in the transverse plane. This equation is discretized on a rectangular grid (mainly by the FFTW fast Fourier transform). Such a calculation is made at each time step. The size and the refinement of this grid are fundamental to obtain accurate results. In the C-version of the

program, the total size of the grid is controlled by the user through the parameters *cut_x* and *cut_y*, while the refinement is specified by the number of cells *n_x* and *n_y*. The output file provides two indicators to estimate the adequacy of the range covered by the grid: *miss.1&2* and *out.1&2,* indicating the maximum relative amount and the maximum number of interacting particles which were outside the grid during at least one time step (for beams 1&2, respectively).

A procedure was developed to automatically compute the parameters *cut_x* and *cut_y* as well as *n_x* and *n_y*. This can be beneficial to reduce the computing time in chained runs, e.g. for simulations of the feedback system to maintain the luminosity at the interaction point, in which input beam distributions can vary for each successive collisions to be computed. Fixed grid parameters common to all collisions may then be too conservative for a majority of collisions, hence significantly slowing down execution as compared to adapting them to changing beam sizes.

The implemented method is based on guessing the maximum excursion of beam particles during a collision from the initial beam sizes, divergences, offsets, bunch population, and based on computing the disruption angle.

3.2.1 Algorithm

The parameters *cut_x* and *cut_y* specifying the range of the grid are expressed as:

$$cut\_x = x_0 + \Delta x + 4\sigma_x + \Delta_{xdisrup}$$
$$cut\_y = y_0 + \Delta y + 4\sigma_y + \Delta_{ydisrup}$$

x_0 and y_0 are the means of the two beams' positions, namely:

$$x_0 = 1/2(x_{0beam\,1} + x_{0beam\,2})$$
$$y_0 = 1/2(y_{0beam\,1} + y_{0beam\,2})$$

where $x_{0beam1,2}$ and $y_{0beam1,2}$ represent the average positions of the particles of beams 1 and 2.

Δx and Δy are the offsets between the two beams in the x and y directions, namely:

$$\Delta x = 1/2(x_{0beam\,1} - x_{0beam\,2})$$
$$\Delta y = 1/2(y_{0beam\,1} - y_{0beam\,2})$$

Finally, $\Delta_{xdisrup}$ and $\Delta_{ydisrup}$ are offsets added to account for disruption effects in both planes, derived as follows, based on the calculation of the center of mass disruption angle by Yokoya and Chen [6]. Given an offset δ between the two beams, expressed in units of sigma, the center of mass disruption angle can be written in terms of a form factor $F(\delta)$:

$$\Theta = \frac{1}{2}\theta_0 F(\delta)$$

where

$$\theta_0 = \frac{2Nr_e}{\gamma\left(\sigma_x + \sigma_y\right)}$$

is a parameter characterizing the disruption angle, with N the number of particles per beam and γ the Lorentz factor. The form factor expression from Yokoya and Chen was used:

$$F(\delta) = \delta\left[C_1 + C_2\delta^2 + \frac{1}{\pi^2}\delta^4\right]^{-1/4}$$

where

$$C_1 = \left(1 + A^2\right)\left[1 + \frac{0,5}{0,6 + \left(\sqrt{D} - 2,5\right)^2}\right]^2 \qquad C_2 = \left[\frac{1,2D^2}{D + 10}\right]^2 \qquad A_{x,y} = \frac{\sigma_z}{\beta_{x,y}}$$

and

$$D_{x,y} = \theta_0 \frac{\sigma_z}{\sigma_{x,y}}$$

is the disruption parameter. From this Θ angle, the additional offsets needed horizontally and vertically to take into account disruption can be estimated:

$$\Delta_{xdisrup} = f\sigma_x\Theta \qquad \Delta_{ydisrup} = f\sigma_y\Theta$$

where a factor f was included to provide some margin. This arbitrary factor was determined empirically during tests, as a function of $h_{x,y} = 2\Delta_{x,y} / \sigma_{x,y}$:

- if $h > 2$ $f = 3$
- if $2 > h > 1.25$ $f = 4$
- if $1.25 > h > 0.75$ $f = 5$
- if $0.75 > h$ $f = 6$

For the number of cells *n_x,y*, the power of 2 nearest to *cut_x,y* / $\sigma_{x,y}$ was adopted, as this maximizes the performance of the fast Fourier transform.

The procedure for automatic grid adjustments described above was tested with respect to the output parameters *miss*, giving satisfactory results. It can be improved by further testing and tuning.

3.3 Depolarising effects

The spin motion in an electromagnetic field is influenced by two effects: the classical spin precession, described by the Thomas-Bargmann-Michel-Telegdi (T-BMT) equation and the spin-flip effect, which occurs when electrons emit beamstrahlung radiation. In the following, the implementation of these effects in *guineapig*++ is described. The treatment available in the CAIN simulation [7] was closely followed. A first evaluation of the results obtained and comparison with CAIN can be found in [8].

3.3.1 Spin precession

As the simulation uses macro-particles to represent typically 50000 to 100000 or even 200000 'true' particles, polarisation vectors P can be used to describe their spin state, as probabilities for the spins of each statistical ensemble of particles to be oriented along a given axis. The time evolution of a polarisation vector is described as a precession by the equation:

$$\frac{dP}{dt} = \Omega_{BMT} \wedge P$$

together with the T-BMT equation:

$$\Omega_{BMT} = -\frac{e}{m_e c}\left[\left(a + \frac{1}{\gamma}\right)B - \frac{a\gamma}{\gamma+1}\beta\left(\beta.B\right) - \left(a + \frac{1}{\gamma+1}\right)\beta \wedge E\right]$$

where c is the velocity of light, β is the rapidity (velocity divided by c), e is the charge of the electron, E and B are the electromagnetic field vectors and a is the anomalous gyromagnetic moment of the electron.

The electromagnetic field created by the opposite beam and computed in *guineapig*++ is used to compute the precession of the macro-particles according to the T-BMT equation. Initial sets of polarisations for the colliding beams can be specified as additional parameters in input files with externally generated macro-particles. They can also be provided as initial polarization state vectors, valid in this case for all macro-particles generated internally in the program.

A deviation of the anomalous gyromagnetic moment of the electron from the usual value $\alpha/2\pi$ is expected in the case of strong electromagnetic fields. The *guineapig*++ program follows the treatment available in CAIN [7], based on the parametrisation provided by V.N. Baier as a function of the parameter Y to characterise the field strength[4].

3.3.2 The Sokolov-Ternov spin flip

The emission of beamstrahlung in the transverse magnetic field of the opposite bunch can be accompanied by spin transitions with probabilities specified according to the theory of Sokolov and Ternov [9]. These transition rates can be used to describe the statistical evolution of the polarisation vectors.

In the case of longitudinally polarised beams, the probabilities to flip the spins in either direction is the same and the overall effect is hence depolarising[5]. Effects at ILC are small in comparison with the T-BMT precession, but increase at higher energies.

---

[4] Y is defined as 2/3 of the synchrotron radiation critical energy divided by the beam energy

[5] The situation is different for storage rings, where a transverse polarisation gradually builds up as a result of the slight bias which exist in this case in the probabilities, in favour of the spins aligning themselves with the magnetic field.

The *guineapig*++ implementation currently uses the expressions for the probabilities derived for spin-flip and non spin-flip transitions as provided in the CAIN manual [7]. This allows technical comparisons and studies. Only electrons and positrons have been considered so far. A cross-check of the used expressions and evaluation of the treatment in the case of strong electromagnetic fields have however not been done and may be desirable in the future.

4 Conclusion

A new C++ version of the GUINEA-PIG program has been implemented and is now available for the design studies of ILC and other future colliders. The package is structured, maintained, distributed and documented through dedicated computer tools. It can more conveniently accommodate the development of new features. A couple of new features are already included such as automatic grid dimensioning and, principally, depolarization effects.

Acknowledgment

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.

References

[1] D. Schulte Ph. D. Thesis, University of Hamburg 1996. TESLA-97-08
[2] D. E. Knuth, "The Art of Computer Programming", Volume 2: Semi-numerical Algorithms, 3rd edition (Addison-Wesley, Boston, 1998).
[3] C. Rimbault et al., "GUINEA-PIG++: An upgraded version of the linear collider beam-beam interaction simulation code GUINEA-PIG", EuroTeV-Report-2007-056, LAL/RT-07-15.
[4] C. Rimbault et al., "Impact of beam-beam effects on precision luminosity measurements at the ILC", EUROTeV-Report-2007-017, LAL-07-157, JINST. 2:O9001.2007.
[5] C. Rimbault et al., "Bias on absolute luminosity measurements at the ILC from beam-beam space charge effects", EUROTeV-Report-2007-055, LAL/RT-07-14.
[6] K. Yokoya, P. Chen "Beam-beam phenomena in linear colliders", Lecture Notes in Physics 400. *Frontiers of Particle Beams; Intensity Limitations*, Springer Verlag. KEK Preprint 91-2.
[7] K. Yokoya, "User's manual of CAIN", http://lcdev.kek.jp/~yokoya/CAIN/cain235/CainMan235.pdf
[8] C. Rimbault, et al., "Implementation of depolarization due to beam-beam effects in the beam-beam interaction simulation tool guineap++", EUROTeV-Report-2008-066.
[9] A. A. Sokolov, M. Ternov "On polarization and spin effects in the theory of synchrotron radiation", Soviet Physics Doklady, volume 8, n° 12.