GENERAL PARTICLE TRACER: A 3D CODE FOR ACCELERATOR AND BEAM LINE DESIGN

S.B. van der Geer, M.J. de Loos, Pulsar Physics,
De Bongerd 23, 3762 XA Soest, The Netherlands

Abstract

The General Particle Tracer (GPT) code is a well established simulation platform for the study of charged particle dynamics in electromagnetic fields. The code is completely 3D, including the space-charge model. Because of its modern implementation, GPT can be conveniently customized without compromising its ease of use, accuracy or simulation speed. In this paper we will present the latest version of GPT, version 2.40.

This newest release is twice as fast, is capable of simulating different types of particles simultaneously and includes many new elements. The new integration method is based on a fifth order embedded Runge-Kutta method with adaptive stepsize control to ensure both accuracy and speed in solving the particle's equations of motion in time domain. Furthermore any additional differential equations can be solved while tracking the particles.

GPT features also include complete freedom in the initial particle distribution and the flexibility to position and orient all beam line components. Separate utility programs calculate macroscopic quantities, produce ASCII and graphical output and automate parameter scans.

In this paper we report on the internal structure of the General Particle Tracer. In addition various pre- and postprocessors, combined in the integrated Windows 95/NT based graphical user interface, will be described.

1 INTRODUCTION

The General Particle Tracer (GPT) [1,2] is a software package developed to study 3D charged particle dynamics in electromagnetic fields. The purpose of GPT is to aid in the design of accelerators and beam lines by using modern, particle tracking techniques

GPT is capable of tracking any number of particles through complex electromagnetic fields taking all 3D effects and space-charge forces into account. This produces highly accurate and reliable results. GPT offers these capabilities in a very user friendly and easy to customize package. Due to the general character of the code and its flexibility, GPT is suited for a large number of purposes.

The fields of physical structures are represented by so called elements. GPT comes with a large set of standard elements for basic structures such as solenoids, quadrupoles and accelerating structures. However, for specific cases custom elements can easily be added.

Elements can be positioned anywhere and in any direction in 3D space. This gives the user complete freedom in constructing the experiment and also components placed off center or rotated can be simulated. The effect of fringe fields can be taken into account and measured or externally calculated fields can be used in the simulation.

The equations of motion for the macro-particles are solved relativisticly in the time-domain using a 5^{th} order embedded Runge-Kutta integrator with adaptive stepsize control. This allows the user to select the required accuracy, while the simulation time is always kept to a minimum.

The GPT package consists of the GPT kernel which performs the actual calculations, an extensive set of pre- and postprocessors including data analysis tools and a graphical user interface.

2 THE GPT KERNEL

A schematic of the GPT executable is shown in figure 1. The following subsections describe the individual components in detail.

2.1 Inputfile

The GPT executable starts by reading one or more ASCII inputfile(s) describing the simulation to perform. The inputfile specifies the initial particle distribution, the 3D electromagnetic field configuration (set-up), the required accuracy of the calculations and the output method. Standard expressions, functions and user defined variables can be used for convenience. Optionally, an MR-file (Multiple Run) can be used to automatically scan any number of parameters.

2.2 Initial particle distribution

The initial particle distribution consists of a number of macro-particles, each typically representing a large number of elementary particles. The distribution can be specified by using any combination of the built-in particle generators or an external file.

Complicated particle distributions can be composed of any number of separate particle distributions, i.e. a beam with a halo or a mixture of different ion species each having their own distribution function.

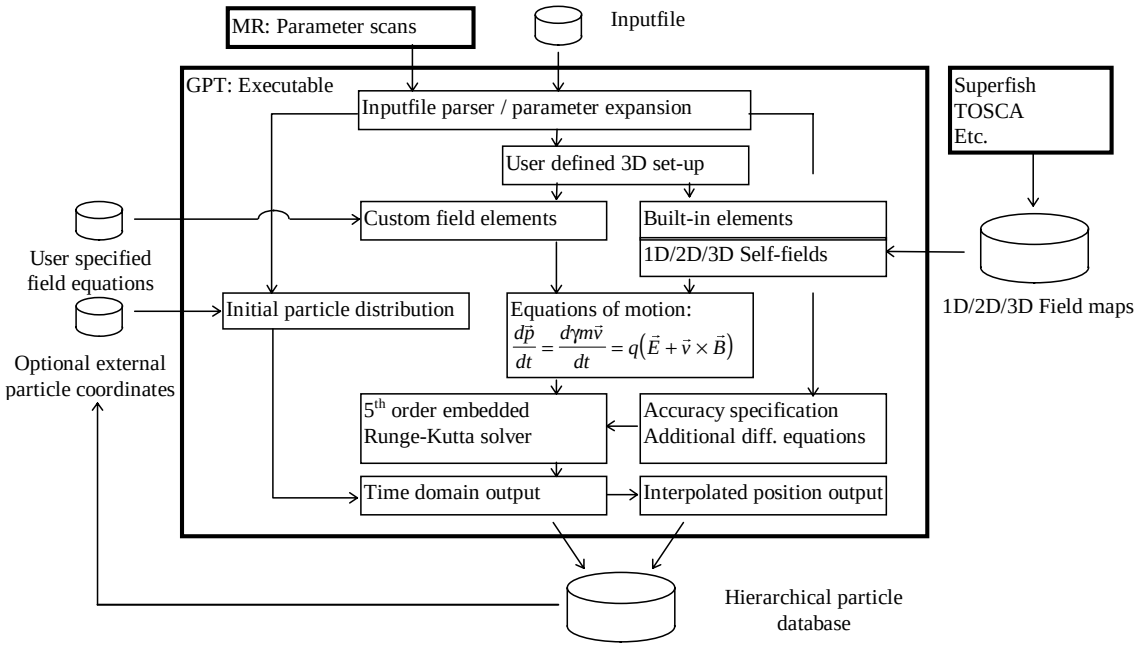

Figure 1: Schematic of the GPT executable.

2.3 Set-up

The set-up defines the 3D electromagnetic field configuration as generated by the beam line components. It can be composed of any number of built-in elements, external 2D or 3D field-maps and user defined expressions in custom elements. There is no limit to the number, location and orientation of the elements.

The field-map elements can read electromagnetic field configurations calculated by external programs like TOSCA and the SUPERFISH set of codes. Also measured field information can be used in the calculations.

To increase the flexibility of GPT, users can easily write their own elements for specific configurations. The interface for writing custom elements is very broad and covers the development of user defined electromagnetic field configurations, custom initial particle distributions, interfaces to other codes and additional differential equations.

All GPT elements are developed in their own coordinate system and written in separate source files. These files are platform independent and allow users to freely exchange elements with each other without having to modify the GPT kernel. When an element is positioned in 3D space, the GPT kernel takes care of all required coordinate transformations.

2.4 Space-charge

The self-fields of the particles are also part of the electromagnetic fields through which the particles are tracked. Space-charge effects can be calculated using a 1D, 2D or 3D model depending on the type of simulation.

The one dimensional model can only be used for beams with a constant radius. For the simulation of cylindrical symmetric (semi)continuous beams the 2D space-charge model can be used. It is a relativistic point-to-ray interaction model and it interprets particles as moving rays with a homogeneous linecharge density. The 3D routine consists of a 3D relativistic point-to-point model [3]. This version makes no assumptions about particle distribution, and is therefore well suited for all design problems.

2.5 Equations of motion

The equations of motion for the macro-particles are solved relativsticly in the time-domain using a 5^{th} order embedded Runge-Kutta integrator with adaptive stepsize control [4]. When required, the individual particle coordinates can be obtained with an accuracy of 10^{-10}. The adaptive timestep mechanism modifies the stepsizes according to the gradients of the electromagnetic fields to ensure that all output satisfies the user specified accuracy.

Optionally, the equations of motion are combined with additional differential equations. This mechanism can be used to calculate beam-loading or FEL interaction completely self-consistently.

2.6 Output methods

GPT has two available output modes: time and position output. Time output writes all particle coordinates at user specified time(s). Position output writes all particle coordinates passing any plane in 3D space. This output mode is also known as “nondestructive screen” output. Time and position output can be freely mixed and any number of time and position outputs can be specified.

Optionally, the electric and magnetic fields at the particle coordinates are also output. This can greatly aid in understanding the particle dynamics.

3 PRE- AND POSTPROCESSING

Being able to simulate charged particle dynamics in time-dependent 3D electromagnetic field configurations is usually far from sufficient for serious accelerator and beam line design. The simulation data must be analyzed, parameters must be scanned and typically a comparison must be made between different scenarios.

GPT is accompanied by a number of pre- and postprocessing programs as well as interfaces to other software packages to ease this design process. These tools are combined with GPT and integrated into the Windows 95/NT based graphical user interface, GPTwin. GPT on UNIX machines uses command-line versions of these programs. The typical data flow within GPTwin is shown in figure 2. The following subsections describe the individual components in detail.

3.1 GPTwin

GPTwin is the all-integrated Windows 95/NT based user interface for GPT. It provides a standard editor with on-line help capabilities for editing the inputfile. Inputfiles can be run and combined with any number of pre- and postprocessing tools. GPTwin is able to directly plot the output of GPT as well as the results of data analysis tools like GDFA.

Any number plots can be shown in separate windows within the GPTwin user interface. When more windows show different views of the same outputfile, the hierarchy between the different views always remains synchronized.

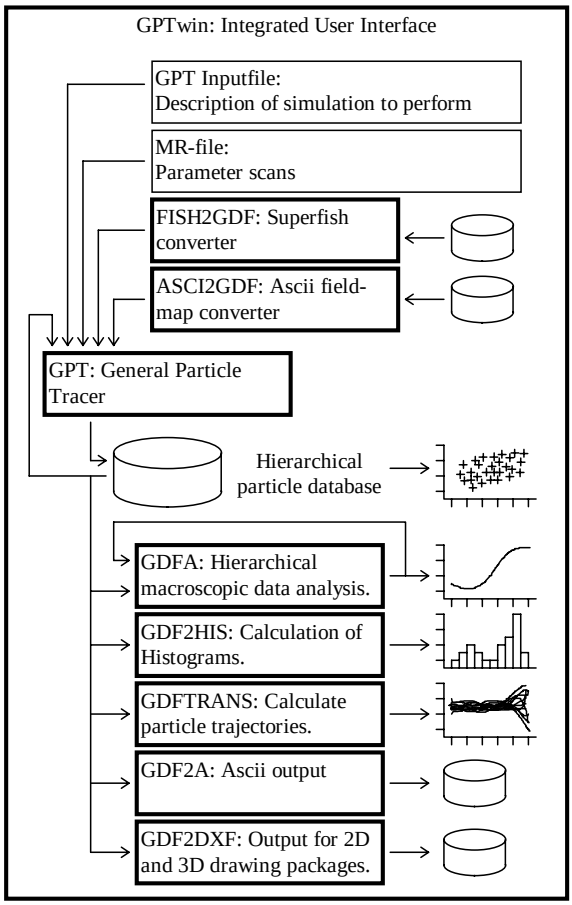

Figure 2: Typical data flow within GPTwin.

3.2 The GDF format

The GPT results are written to a binary file for off-line analysis and interpretation. The file is written in the General Datafile Format (GDF), a multi-purpose, hierarchical file format specifically designed for efficiently storing large quantities of numerical data. It allows the post-processing components to easily extract information. The GDF format is used throughout the GPT kernel, the analysis tools and the GPT elements to keep the code compact and efficient. As shown in figure 2, various conversion utilities are available to convert to and from the GDF format.

3.3 GDFA

The main analysis program for GPT output is GDFA. It calculates macroscopic beam parameters as function of simulation time, position or any scanned parameter. Typical macroscopic quantities like emittance, bunch length, average energy and beam radius can be obtained, but GDFA is capable of calculating many other useful quantities. When needed, the list of GDFA programs can be extended with custom code to calculate the beam parameter of interest. GDFA maintains the original hierarchy in the GDF file when more parameters are scanned simultaneously.

4 CONCLUSION

The high accuracy, the possibility to add custom code, the large number of standard elements and the user friendly interface, all make GPT an attractive tool for beam line and accelerator design. Especially the new field-map elements and the ability to simulate different particle species further enhances its capabilities. The current status of the GPT project can be found on the web at *http://www.pulsar.nl/gpt*

5 REFERENCES

[1] M.J. de Loos, et al, Proc. 5th Eur. Part. Acc. Conf., Sitges, (1996) pp. 1241.
[2] GPT User Manual, Pulsar Physics, Flamingostraat 24, 3582 SX Utrecht, The Netherlands.
[3] Ch. Bourat, CGR-MeV, Thesis 1988.
[4] W.H. Press, et al., Numerical Recipes in C, Cambridge Univ. Press, 2nd edition, (1992) pp. 714.