IONIZATION PROFILE MONITOR SIMULATIONS - STATUS AND FUTURE PLANS

M. Sapinski*, P. Forck, T. Giacomini, R. Singh, S. Udrea, D. M. Vilsmeier, GSI, Darmstadt, Germany
B. Dehning, J. Storey, CERN, Geneva, Switzerland
F. Belloni, J. Marroncle, CEA/IRFU, Gif-sur-Yvette, France
C. Thomas, ESS, Lund, Sweden
R. M. Thurman-Keup, FNAL, Illinois, U.S.A.
K. Satou, J-PARC, Tokai, Japan
C. C. Wilcox, R. E. Williamson, STFC/RAL/ISIS, Oxford, United Kingdom

Abstract

Nonuniformities of the extraction fields, the velocity distribution of electrons from ionization processes and strong bunch fields are just a few of the effects affecting Ionization Profile Monitor measurements and operation. Careful analysis of these phenomena require specialized simulation programs. A handful of such codes have been written independently by various researchers over the recent years, showing an important demand for this type of study. In this paper we describe the available codes and discuss various approaches to Ionization Profile Monitor simulations. We propose benchmark conditions to compare these codes among each other and we collect data from various devices to benchmark codes against the measurements. Finally we present a community effort with a goal to discuss the codes, exchange simulation results and to develop and maintain a new, common codebase.

INTRODUCTION

The Ionization Profile Monitors were first buildt in the 1960s as simple devices to measure transverse profiles of particle beams without affecting them. The basic idea of the device is that the distribution of electrons or ions from the rest gas ionization mirrors the original beam distribution, however the effects of guiding field nonuniformities, beam space charge or initial velocities due to the ionization process can affect the measurement. A number of numerical simulations have been written dealing with those aspects. Because of their specificity - for instance tracking of low energy electrons or ions, beam charge distributions - the established codes, like Geant4 [1] or CST Studio [2], are usually not applicable to IPM simulations.

In this paper first we present the most important stages of an IPM simulation. These logical stages can be used to modularize the simulation code. In the second part we present simulation codes known to us. These codes were discussed during the Ionization Profile Monitor simulation kickoff workshop [3]. They show a variety of approaches to IPM simulations. Finally we discuss the collaborative tools prepared in order to compare various codes, benchmark them against measurements and share the results. More information can be found on the collaboration's TWiki pages [4].

It should be stressed that other beam devices, for instance Beam Fluorescence Monitors or even electron lenses, can be simulated using similar techniques.

SIMULATION COMPONENTS

The simulation can be divided into the following stages, which cover various physics phenomena and can be used to modularize the simulation code:

- **Ionization** - The purpose of this component is to generate the initial momenta of the particles to be tracked. The most promising approach is to use - if available - a realistic double differential cross section (DDCS) for obtaining the energies and scattering angles of ionization products. The ionization process depends on the beam particle type, beam energy and the residual gas species, therefore an appropriate cross section model for each beam configuration can be chosen.

- **Guiding fields** - The purpose of this component is to provide the externally applied electric and magnetic fields in the volume where the particles must be tracked. Usually either uniform fields are used or a field map is imported from an EM-solver. In most applications the field nonuniformities are just sources of errors, but in some cases fields are deeply nonuniform and their precise knowledge is fundamental in order to reconstruct the real beam profile.

- **Beam fields** - This module provides the electromagnetic fields generated by the beam. For highly relativistic beams the electric field is mainly transverse to the beam and its longitudinal component can be neglected. In such case a '2D' approach in which the longitudinal shape of the beam is modelled by a simple shifting of the bunch charge distribution with time is often used. Other approaches include solving Poisson's equation analytically or using EM solvers. This allows for the creation of a three-dimensional field map. The magnetic field of the beam is usually neglected because its impact on slowly-moving ionization products is much smaller than the electric field.

* m.sapinski@gsi.de

ISBN 978-3-95450-177-9

- **Particle tracking** - The purpose of this module is to update the positions and velocities of the tracked particles at each time step of the trajectory before they reach the detector. The motion depends on the initial velocities and on the electric and magnetic fields of the IPM chamber and the beam. Most codes consider only non-relativistic motion of particles. Various approaches exist such as analytic solutions of the equations of motion for special cases or numerical solutions using a Runge-Kutta-method.

Other components, whose effect is of minor importance for typical IPM, but which can be critical for other applications, are:

- **Gas dynamics** - Simulation of rest gas thermal motion or motion due to gas injection (for instance cold gas jet) or gas burnout - all these effects lead to a non-uniform gas density in vacuum;
- **Wakefields** - Simulation of transient electric and magnetic fields due to mirror charges on beamline components;
- **Synchrotron radiation** - For relativistic beams gas ionization can be caused by synchrotron radiation generating additional profiles;
- **Multiple beams** - In cases when multiple beams are present in the vacuum chamber (eg. electron lens) their impact on the measurement should be estimated.

KNOWN SIMULATION CODES

In Table 1, existing codes are summarized including their approach to the main simulation components. Most of the codes are not public. A short description of each code is provided below.

GSI Code

The GSI code, developed around 2002, is used for a quick estimation of the influence of the electromagnetic field of ion bunches on the transverse profile measurements with IPM. The bunches have the shape of a prolate spheroid, with the spherical case treated separately. The charge density distribution within bunches may be chosen to be homogeneous or parabolic $\rho(r, z) \propto 1 - [(r/b)^2 + (z/a)^2]$. These assumptions allow for computation of the electrostatic field of the bunches using analytic formulas [5]. The field in the laboratory frame is computed through the Lorentz transformation. The guiding fields are uniform and can be separately activated. Both electrons and ions may be tracked. These particles are randomly generated within the volume of the bunches and have a spatial distribution in accordance with the bunch charge density. No interaction is considered between them, thus each one is tracked independently according to the classical laws of motion until it reaches the detector plane or a user defined time limit gets exceeded, in which case the particle is discarded. While ions are generated at rest, for electrons a simple double-differential cross-section (DDCS) model can be applied to generate their velocities.

PyECLOUD-BGI

This code is an adaption of the PyECLOUD package used to study the electron cloud built-up [6]. It has been created in 2012 to address the issue of beam space-charge influence on profile measurements for LHC monitors [7]. It assumes an ultra-relativistic beam and uses an analytic formula to compute the transverse electric field of a two-dimensional elliptical Gaussian charge distribution [8]. For a circular beam Gauss' law is used to compute the electric field. The simulation incorporates electric and magnetic guiding fields which are assumed to be uniform and perfectly aligned. The initial velocities of electrons are computed according to the DDCS [9]. The tracking of electrons is performed according to an analytic formula that is obtained by solving the equations of motion of a charged particle in a constant transverse electromagnetic field. The code is public [4].

FNAL Code

The IPM simulation code at FNAL was originally developed to track low energy electrons in an electron beam profiler for the proton beams. The code was adapted to solve for electron trajectories in the IPMs. Of particular interest was the behavior of the ionization products in the new, gated IPMs [10]. The simulation is written in MATLAB, utilizing its parallelization capabilities, and implements a numerical solver for the relativistic equations of motion of the electrons or ions. The solver is a second order ODE solver with the added feature that the momentum is scaled to preserve its magnitude when applying the solver to the magnetic portion of the guiding fields. The guiding fields are typically calculated via interpolation of externally pre-calculated 3D field maps. For the FNAL IPM, a 2D electric and a 3D magnetic field maps are used. In the case of the fields of the beam, only a single bunch fields are evaluated however, with the proper time shifts in the solver, one can effectively simulate a train of bunches. Since the bunch fields are evaluated externally, the bunch shape is not limited to any particular functional form. Typically a Gaussian shape is used and there is a separate MATLAB function to produce the fields for that shape. The initial momenta of the ionization electrons are chosen from a $1/E^2$ distribution [11]. The initial momenta of the ions are taken from the thermal energy distribution of an ideal gas. The simulation is not very user friendly with many separate functions or scripts that must be called to setup it up correctly.

ISIS Code

The ISIS IPM simulation begins with beam data being recorded from the machine, providing both an input for the simulation and also a benchmark to compare the results against. The 2D beam profile is measured with SEM grids located close to the IPM, and this is used alongside intensity measurements to define the beam within the simulation. Alternatively, purely theoretical beam distributions can be

ISBN 978-3-95450-177-9

Table 1: The Current Simulation Codes. See Text for the Details.

Name/Lab	Language	Ionization	Guiding field	Beam shape	Beam field	Tracking
GSI code	C++	simple DDCS	uniform E,B	parabolic 3D	3D analytic relativ.	numeric R-K 4th order
PyECLOUD-BGI /CERN	python	realistic DDCS	uniform E,B	Gauss 3D	2D analytic relativ. only	analytic
FNAL	MATLAB	simple SDCS	3D map E,B	arbitrary	3D numeric relativ. (E and B)	num. MATLAB rel. eq. of motion
ISIS	C++	at rest	CST map E only	arbitrary (CST)	2D numeric (CST) non-relativ.	numeric Euler 2nd order
IFMIF	C++	at rest	Lorenz-3E map E only	General. Gauss	numeric (Lorenz-3E) non-relativ.	
ESS	MATLAB	at rest	uniform E,B	Gauss 3D	3D numeric (MATLAB) relativ.	numeric MATLAB R-K
IPMSim3D /J-PARC	python	realistic DDCS	2D/3Dmap E, B	Gauss 3D	2D numeric (SOR) relativ. only	numeric R-K 4th order

defined. The beam's field is calculated using CST EM Studio [2], in which the beam is modelled as concentric elliptic cylinder charge distributions with the same aspect ratio as the measured beam. The charge levels are chosen to represent an elliptic distribution within the beam and are calculated to match the measured beam intensity. This beam is placed inside a 3D model of the IPM that also includes the guiding field. The internal fields of the monitor are then calculated and exported for use in a C++ ion tracking code. To approximate a time dependent space charge, a second electric field is calculated with the beam removed from the model. The tracking code itself generates a uniform distribution of ions within the beam's volume, and tracks the motion of these through the electric fields calculated in CST. The equations of motion are solved using a 2nd order Euler method. For the extracted beamline IPM the electric field is swapped for the guiding field after 200 ns to model the beam leaving the monitor. When modelling synchrotron IPMs, an electrostatic approximation of the average charge within the monitor is used. During post-processing an elliptic weighting is applied to each ion based on its initial position to compensate for the initially uniform ion distribution. A further weighting is applied based on the longitudinal angle of incidence at which each ion reaches the detectors, to account for the variation in detection efficiency. This weighting was measured using an in-house vacuum tank test at ISIS [12].

IFMIF Code

The IFMIF code was created to design the IFMIF/LIPAc monitor and to investigate space-charge correction algorithm [13]. The beam transverse profile $\overrightarrow{P}$ is described by a Generalized Gaussian Distribution (GGD). In this case the space-charge impacts only on the 2nd (σ) and the 4th (kurtosis) moments of the distribution. The distortion of the profile is described in matrix formalism: $\overrightarrow{P}_{\text{meas}} = \text{M} \times \overrightarrow{P}_{\text{real}}$. A set of M-matrices for various beam parameters is prepared using the tracking procedures written in C++. Guiding (only E) and beam fields are calculated using Lorenz-3E [14]. The correction procedure uses the σ and kurtosis of the measured profile, beam intensity, beam energy and the value of the guiding field to select the proper M-matrix. The procedure is iterative.

ESS Code

The ESS code was developed initially in MATLAB, for the purpose of investigation of the space charge effect on the performance the IPM in the ESS Cold Linac sections. The code solves the classical equations of motion for arbitrary charged particles exposed to electromagnetic fields. The equations of motion are solved using the standard MATLAB's ODE solver, based on non-linear Runge-Kutta method. The fields are composed of uniform and static guiding fields and dynamic fields given by a 3D Gaussian distribution of moving moving charges. The field generated by the bunch is calculated at the position of the particles and given time by the ODE solver. It is generated in the bunch rest-frame and then transformed to the lab-frame using Lorentz transformation. Bunches periodically cross the IPM interaction volume, and this is taken into account in case of particles with large mass which can "see" several bunches before leaving the beam. The initial spatial distribution of tracked particles is randomly generated following the bunch 2D transverse distribution, and linearly along the longitudinal axis. The detector is modelled by means of an event function used by the ODE solver. It calculates the time at which the particles cross the detector surface and returns their positions and speeds as an additional output to the time – phase space output of the ODE solver. The limitation of the code is extremely high electric field, which would drive the particles to relativistic motion already in the interaction region. Finally, in order to improve the code performance, MATLAB's multi-processing or rewriting the code in a C++ are considered.

ISBN 978-3-95450-177-9

J-PARC Code - IPMsim3D

The IPMsim3D code assists IPM design works. It simulates charged particle trajectory in 3D, traveling in guiding fields and beam fields of circulating bunched beams. The electric and magnetic fields maps are calculated by using external code POISSON/Superfish [15] in case of 2D and CST Studio [2] in case of 3D. Uniform fields can also be selected. The charge density of beam is defined as 3D Gaussian distribution and the beam is assumed to be relativistic. At first, 2D grids are set internally and unit charge is distributed on each grid according to the transverse charge density. Then iterative calculations determine the self-consistent potential and the electric field on each grid. The longitudinal charge distribution is used to normalize the line density. This method, called succesive over-relaxation (SOR), will be adapted to a more general density distribution.

A charge particle equation of motion is solved based on the 4th order Runge-Kutta method. The initial position of the tracked particle is generated according to the transverse and longitudinal charge density. The initial momentum is calculated from analytic DDCS, the same as in PyECLOUD-BGI. The semi-empirical single differential ionization cross sections for some gas species are also used, where the electron emission angle is assumed to be perpendicular to the beam axis. The tracking is stopped when the particle crosses the detector surface. This code is public [4] and it was used to design IPM for CERN PS [16].

COMMON TOOLS

In order to facilitate the comparison of various codes against each other and against measurements a specialized data format, based on the W3C XML standard, was developed. During development the focus was not only on introducing a common data format but also on creating a convenient way of storing metadata about measurements. A file contains a number of beam profiles and/or images together with important information about the type of the beam and the configuration of the device - IPM for example - such as extraction voltage and magnetic field. Specification of the proposed format is available [4].

A Python GUI has been developed for visualization and processing the XML data [4]. A set of benchmark beams have been proposed. A comparison of one such beam (CERN PS case) using two codes is shown in Fig. 1.

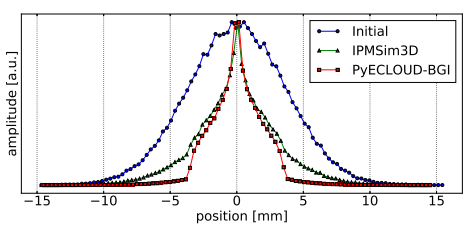

Figure 1: Comparison of profiles obtained with two simulation codes for PS benchmark beam (electron detection).

CONCLUSIONS AND PLANS

A collaboration effort to simulate effects related to beam interaction with rest gas and measurement of the beam profile using the ionization process have been launched. The goals are to exchange information between researchers working in different laboratories and share experience in the design and understanding of these devices.

Currently a few of the described codes are still under independent development, however a common effort to develop a universal, modular multi-purpose approach is also ongoing.

REFERENCES

[1] IEEE Trans. Nucl. Sci. 53 No. 1 (2006) 270-278.

[2] CST - Computer Simulation Technology, `http://www.cst.com/`.

[3] IPM simulation kickoff workshop, CERN, March 2016, `https://indico.cern.ch/event/491615/`.

[4] IPMSim, `https://twiki.cern.ch/twiki/bin/view/IPMSim/`.

[5] P. Strehl, "The Electromagnetic Fields of Bunches" in "Beam Instrumentation and Diagnostics", pp. 341-375, Springer, 2006 (ISBN 978-3-540-26404-0).

[6] G. Iadarola et al.,"Electron Cloud Simulations with PyECLOUD", Proc. of ICAP2012, WESAI4.

[7] D. Vilsmeier, "Profile distortion by beam space charge in Ionization Profile Monitors", CERN-THESIS-2015-035.

[8] M. Bassetti et al., "Closed expression for the electrical field of a two-dimensional gaussian charge", CERN-ISR-TH/80-06, 1980.

[9] A.Voitkiv et al., "Hydrogen and helium ionization by relativistic projectiles in collisions with small momentum transfer", J.Phys.B: At.Mol.Opt.Phys.32 (3923-3937), 1999.

[10] J.R. Zagel et al., "Third Generation Residual Gas Ionization Profile Monitors at Fermilab", in Proc. of IBIC2014, TUPD04.

[11] F. Sauli, "Principles of Operation of Multiwire Proportional and Drift Chambers", CERN 77-09, 1977.

[12] C. C. Wilcox, "An Investigation into the Behaviour of Residual Gas Ionization Profile Monitors in the ISIS extracted Beamline", these proceedings.

[13] J. Egberts, "IFMIF-LIPAc Beam Diagnostics: Profiling and Loss Monitoring Systems", thesis, Univ. Paris Sud, 2012.

[14] INTEGRATED Engineering Software, `https://www.integratedsoft.com/`.

[15] J.H. Billen, et al., "POISSON/SUPERFISH on PC Compatibles", Proc. of Linear Accelerator Conf. TH4-60, 778, 1992.

[16] J.W. Storey et al.,"Development of an IPM Based on a Pixel Detector for CERN Proton Synchrotron", Proc. of IBIC2015, TUPB059.

ISBN 978-3-95450-177-9