A MULTI-PARTICLE ONLINE BEAM DYNAMICS SIMULATOR FOR HIGH POWER ION LINAC OPERATIONS*

X. Pang†, L.J. Rybarcyk, S.A. Baily
Los Alamos National Laboratory, Los Alamos, NM, 87544, USA

Abstract

A fast multi-particle online beam dynamics simulator has been developed at LANL. It is a marriage of multi-particle beam physics algorithms and graphics processing unit (GPU) technology. It combines the execution efficiency of the C/C++ programming language and a powerful yet flexible user interface via Python scripts. Therefore, it is not only accurate and fast, but also very easy to use. We have used this simulator at LANSCE to guide linac tuning, explore optimal operational settings, and test new ideas.

INTRODUCTION

Why Another Simulator?

Accelerator control rooms are usually equipped with online beam modeling tools to help guide machine tuning. These tools, which typically have access to machine set points through the control system, can not only help physicists and operators set up the machine faster, but also provide information on the beam properties in areas where no measurements can be made. However, almost all of the existing online modeling tools today are either based on single-particle tracking or on envelope models. While they might perform sufficiently well for nicely formed beams, they cannot predict the nonlinear motions of a real beam or estimate losses, especially in high-power operations when beams can be highly nonlinear and chaotic.

The logical next step to improve the status quo is to use a multi-particle beam dynamics code to provide more realistic predictions. However, most of the existing multi-particle simulation tools need either significant computational time or supercomputer resources. This makes them impractical to use during real world machine operations where fast turn-around is required and where they may be in use for long periods of time. In addition, they are typically not configured to have ready access to online machine specific set points.

One can clearly see the gap that exists between the over-simplified but fast models used in control rooms and the highly sophisticated yet slow multi-particle simulation tools which are usually used during the design process. The goal of our development is to fill this gap by providing a multi-particle simulation tool that is both accurate and fast enough to be used in real world accelerator tuning and operation.

Why Use a GPU?

The graphics processing unit (GPU) is at the frontier of high performance computing [1]. It powers several of the world's most powerful supercomputers and it has also democratized super-computing by enabling cluster performance on people's personal desktops. For us, the GPU offers outstanding parallel performance and it is also the most cost effective way to provide 24/7 availability for our online simulator. With around a $600 USD investment in the GPU hardware, one can get up to 100 times speedup compared to a single threaded CPU. And this GPU workstation can be dedicated to accelerator operations 24/7.

How to Use It?

This is where the users can freely apply their creativity. We have applied this tool to guide turn-on of the LANSCE linac, to test what-if scenarios, to optimize operational machine settings by combining it with the multi-objective optimization algorithms, and to test a new automatic tuning/control scheme. More details will be covered in later sections.

THE SIMULATOR

Code Design

The goal of our code design is to ensure fast execution and ease of use. This led us to adopt a combination of a low-level compiled language, i.e. C++/CUDA and a high-level scripting language, i.e. Python. The number-crunching is efficiently carried out by CUDA and C++, however, the users don't have to deal with the complex syntax and the lengthy compilation processes associated with them, but instead can configure and execute a simulation with a high-level script. Figure 1 shows the code hierarchy. The shallow learning curve of Python and the richness of its application libraries allow users with the minimal programming experiences to quickly prototype their ideas.

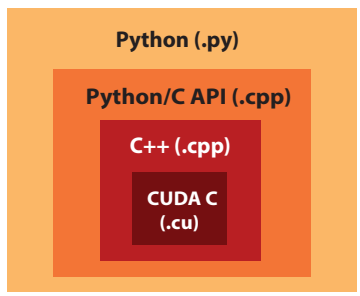

Figure 1: Lower-level CUDA and C++ are wrapped up by Python/C API and compiled into a shared library that can be imported in Python.

The major components of the code structure design are shown in Figure 2. The components that are shaded in blue

* Work supported by U.S. DOE, NNSA under contract DE-AC52-06NA25396. LA-UR-14-28658
† xpang@lanl.gov

ISBN 978-3-95450-173-1

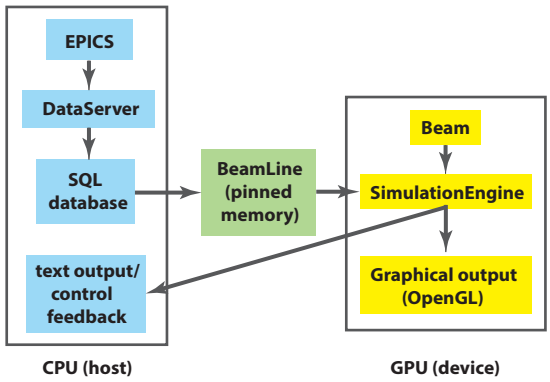

Figure 2: High level code structure and data flow indicated by the arrows.

are generated and stored by the CPUs (host), while the ones in yellow are generated and stored in the GPU (device) during the simulation. The C APIs of the Experimental Physics and Industrial Control System (EPICS) [2] have been used to get real-time machine data from the accelerator control system. The data are then stored in a SQLite database and processed into model physics units that are required for the simulations. The pinned memory (shaded in green) physically sits on the CPU side however, can be accessed by both the CPU and the GPU. It is used in our case to store the beam line information which has to be accessed by the GPU to simulate a beam, and also needs to be updated by the CPU using the information queried in real-time. More detailed description about the code structure can be found in [3].

GPU Performance

One can refer to [3] for more details about the GPU algorithms and optimization we have applied. We compared the code performance using an Intel Xeon E5520 2.27GHz CPU and a NVIDIA GTX 580 GPU (Fermi architecture). For a section of beam line at LANSCE, up to 112 times speedup has been achieved for beam transported without space charge. A speedup factor up to 45 has been achieved for the space charge routine.

Our initial test with the LANSCE CCL (4960 RF gaps + 206 quads + 460 drift spaces) on a NVIDIA Tesla K20 GPU showed that it took the simulator about a second to push 32K particles through it without space charge, and about 10 seconds with space charge (> 6000 space charge kicks). It is likely that this performance will improve as we do not need to apply space charge kicks this often in the CCL due to the lattice design and the diminishing effects of the space charge for the higher energy beam. With a more intelligent space charge routine and further GPU optimization, we can expect the front-to-end simulation for the half mile 800-MeV LANSCE linac to finish within just several seconds.

APPLICATIONS

Tuning Guide

At LANSCE, the half-mile long 800-MeV linac provides both H^+ and H^- beams for user programs. The tune-up procedures usually begin with direct low-power beam measurements and set up of the beam lines based on the predictions from an envelope model. However, in the transition to high-power operations, due to the lack of direct measurement of the beam and a good modeling tool for the high power beam, machine settings are empirically adjusted by operators to achieve minimal beam loss along the linac. These adjustments are usually done in a high dimensional parameter space, which can make the tuning process lengthy, and the machine settings subjective and inconsistent. This is where the simulator can help. Once correctly calibrated, one can use the simulator to predict beam properties at any location along the linac for any beam condition. Therefore, the operators and physicists would no longer be tuning with limited information, but instead are provided with new insight into the beam propagation along the linac. Moreover, they can simulate real beam distributions that are generated directly from emittance scans.

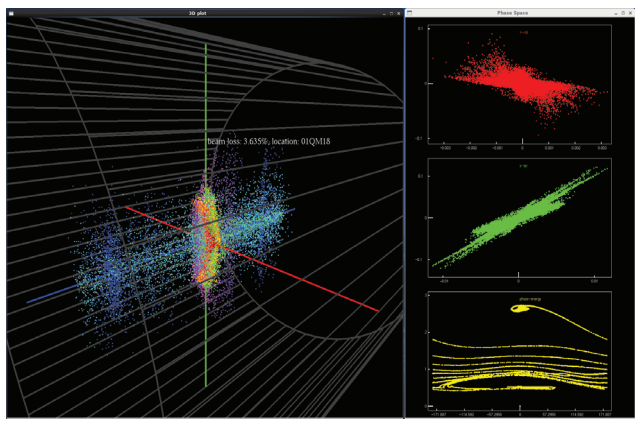

Figure 3: Screen shot of the online simulator graphical output. Left: 3D beam display in x, y, phase coordinates. Right: transverse and longitudinal phase space plots.

Figure 3 shows part of the graphics interface of the simulator. Figure 4 shows a simulated beam distribution at the end of the 100-MeV DTL. One can clearly see the low-energy particles in the longitudinal phase space that make it to the end of the DTL. However, they become beam losses in the chicane following the DTL. The bottom right plot shows the appearance of the nonlinear beam core. The tail of the core can also produce loss in the subsequent elements. These are the features that only multi-particle simulations can provide. They will be very useful in machine tuning especially in high power operations. Figure 5 shows that the simulations can reproduce the actual phase scan experiments for the DTL and CCL to very high accuracy.

Multi-objective Optimization

We have used the simulator in combination with the multi-objective optimization techniques to find optimal operational settings in a high-dimensional parameter space [4]. Figure 6 shows the 2D projection of the estimated Pareto front in the 3D objective space obtained both by the multi-objective genetic algorithm (MOGA) (the left column) and the multi-

ISBN 978-3-95450-173-1

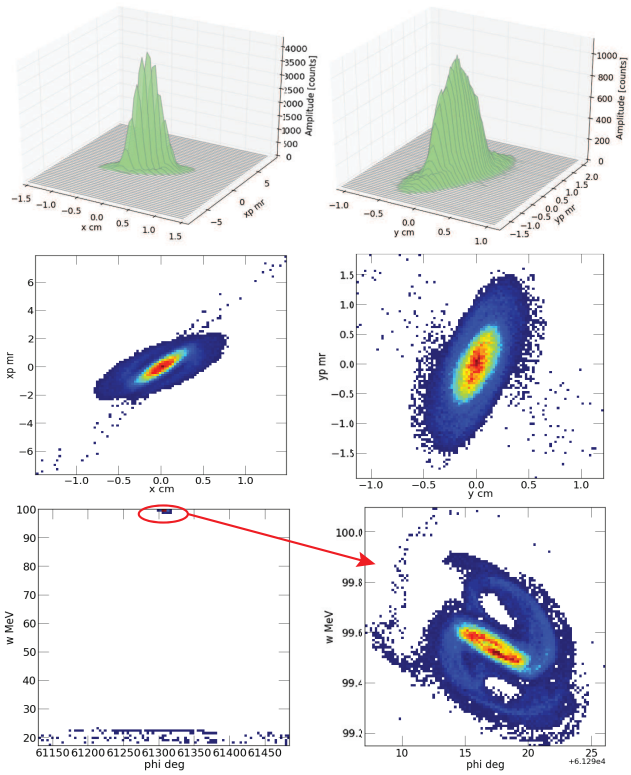

Figure 4: Beam distribution at the end of the LANSCE DTL (100 MeV). Top: beam distribution in x,x' and y,y'. Middle: phase space in x and y. Bottom: longitudinal phase space, ϕ, w showing low-energy particles well below design energy, and a closer look at the beam core around 100 MeV.

objective particle swarm optimization (MOPSO) (the right column). For this graph, the three objectives the MO algorithms are trying to minimize are the longitudinal phase space and phase width at the end of the LANSCE DTL, and power of the lost beam throughout the DTL. There are 11 free parameters for the algorithms to adjust including quadrupole gradients and RF phases and amplitudes of the four DTL tanks. The simulator acts as a virtual experimental environment and provides a cost functions for the MO algorithms to minimize in the process. From this study, we were able to find optimal operating conditions and the fact that the MOPSO converges much faster than the MOGA.

Accelerator Automatic Tuning

An automatic accelerator tuning method has been developed using the simulator as its test bed [5]. This real-time method can simultaneously tune several coupled components of an accelerator to achieve good beam quality. Using the simulator, it has proven to be very efficient and robust to noise. It can even quickly adjust the beam to its best condition with failing elements in the accelerator.

CONCLUSION

The GPU-based online simulator has been proven to be a very useful tool for accelerator operations. By combining

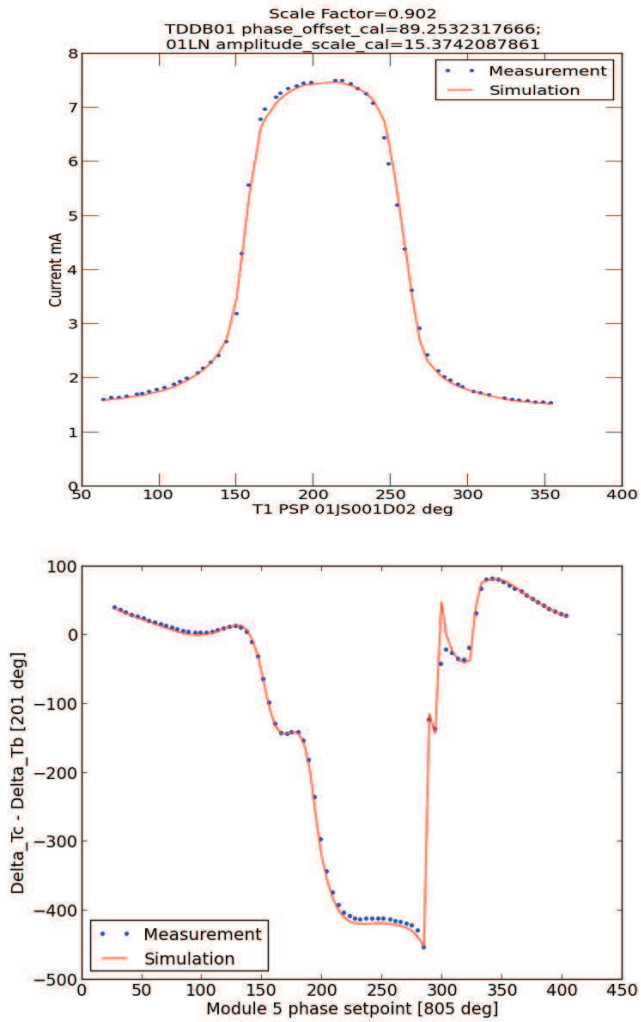

Figure 5: Top: DTL tank 1 phase scan which utilizes an absorber/collector diagnostic. Bottom CCL module 5 phase scan, which utilizes beam phase measurement diagnostics.

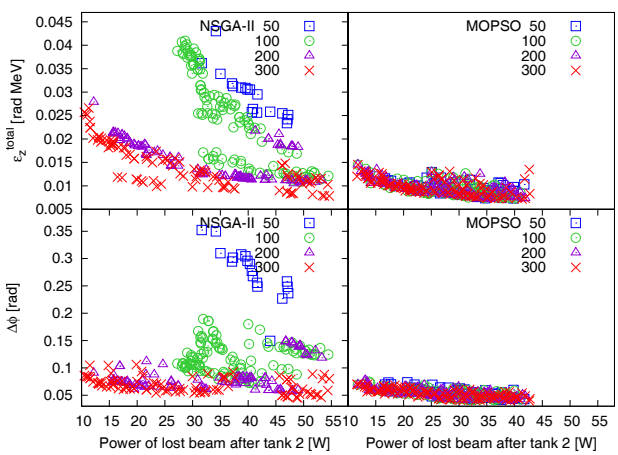

Figure 6: The 2D projections of the estimated 3D Pareto front in the objective space obtained by the NSGA-II and MOPSO at different iterations.

multiparticle beam physics algorithms with GPU technology we are able to bring high-fidelity beam dynamics modeling

ISBN 978-3-95450-173-1

capabilities to the control room environment in a cost effective way. Furthermore, by enabling real-time acquisition of machine parameter setpoints, the simulator is able to track the operation of the accelerator and provide operators and physicists with new insight into beam performance in an operational setting. The list of possible applications is by no means limited to the examples given above. Instead, the high-level Python interface enables the user to easily create and explore new application. Further development of the simulator and its applications are ongoing.

REFERENCES

[1] NVIDIA Corporation, CUDA C Programming Guide, 2014.

[2] EPICS, http://www.aps.anl.gov/epics

[3] X. Pang, L. Rybarcyk, GPU Accelerated Online Beam Dynamics Simulator for Linear Particle Accelerators, Computer Physics Communications, 185, pp. 744-753, 2014.

[4] X. Pang, L. Rybarcyk, Multi-objective Particle Swarm and Genetic Algorithm for the Optimization of the LANSCE Linac Operation, Nuclear Instruments and Methods in Physics Research Section A, 741, pp. 124-129, 2014.

[5] A. Scheinker, X. Pang, and L. Rybarcyk, Model Independent Particle Accelerator Tuning, Physics Review Special Topics - Accelerators and Beams, 16, 102803, 2013.

ISBN 978-3-95450-173-1