A HIGH-LEVEL PYTHON INTERFACE TO THE FERMILAB ACNET CONTROL SYSTEM*

P. Piot[1,2] and A. Halavanau[1,2]
[1] Department of Physics and Northern Illinois Center for Accelerator & Detector Development, Northern Illinois University, DeKalb, IL 60115, USA
[2] Fermi National Accelerator Laboratory, Batavia, IL 60510, USA

Abstract

This paper discusses the implementation of a PYTHON-based high-level interface to the Fermilab ACNET control system. The interface has been successfully employed during the commissioning of the Fermilab Accelerator Science & Technology (FAST) facility. Specifically, we present examples of applications at FAST which include the interfacing of the ELEGANT program to assist lattice matching, an automated emittance measurement via the quadrupole-scan method and tranverse transport matrix measurement of a superconducting RF cavity.

INTRODUCTION

The ACNET accelerator-control system was originally developed for the start of TEVATRON operations in 1983 [1]. It was eventually extended into a comprehensive unified control system for the accelerators in operation at Fermilab. Over the years ACNET has undergone substantial improvement [2]. Additionally, ACNET is complemented with the Accelerator Command Language (ACL), a scripting environment developed as a versatile scripting tool for developing high-level application involving ACNET devices [3].

The ACL environment is extensively used at the Fermilab Accelerator Science and Technology (FAST) facility and while it provides excellent capabilities in accelerator control, it may be challenging for non experts to learn. Additionally, ACL does not enjoy the simplicity, large user base, and extensive open-source packages available from high-level scripting languages such as PYTHON. Based on these observations we have developed a PYTHON interface to ACL. The interface – dubbed PYACL – enables control of accelerator devices within the PYTHON framework (as a script or an IPYTHON notebook). Such a development enables the rapid scripting of high-level applications by non-ACL experts while interfacing the accelerator control and data analysis with the extensive set of available PYTHON packages.

IMPLEMENTATION

Our current implementation of PYACL consists of a PYTHON-based interface providing a set of elementary functions to directly execute ACL commands via operating-system calls; see overview in Fig. 1. Our current tests were performed under the linux operating system but PYACL is platform independent. The PYACL package currently consists of a `acnet.py` toolbox which provides a set of elementary functions to read and write into ACNET devices (either to single devices or a list of devices). Another functionality of `acnet.py` is to handle image acquisition (from a frame grabber) and storage of corresponding arrays as python NUMPY arrays.

In the context of the FAST facility, an additional toolbox (`fast.py`) was developed to include functions specific to the FAST accelerator. These especially include magnet-specific current-to-strength conversion (e.g. I to k_1 for quadrupole) and other functions relevant to beam-dynamics calculations or often use during operation (e.g. automated single-magnet degaussing procedure). Likewise, the script can also write input files for the ELEGANT program [4] thereby providing on-line beam-dynamics modeling of the FAST beamline. In the following sections we present examples of successful application of the PYACL toolbox.

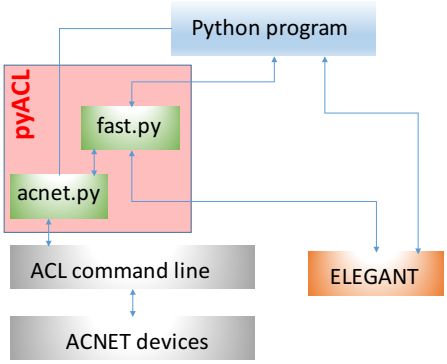

Figure 1: Overview of the interfacing of PYTHON scripts with the Fermilab's ACNET control system.

AUTOMATED SCANS

A simple example of the application of PYACL involves scanning routine commonly used in accelerators. For instance, consider a simple example of hysteresis removal for a dipole or quadrupole magnet. In low-energy accelerators such a hysteresis removal (or "degaussing") is accomplished with an oscillating exponential decaying current excitation. The current is varied in time as $I_k = I_0 \exp(-k/\tau)\cos[(-)^k\pi]$, where $k \in N$ is the step number, τ is the decay time (measured as a number of steps),

* Work supported by the by the US Department of Energy (DOE) contract DE-SC0011831 to Northern Illinois University. Fermilab is operated by the Fermi research alliance LLC under US DOE contract DE-AC02-07CH11359.

I_0 is the saturation current. Such a degaussing procedure can be simply written using the following PYTHON snippet:

```
import acnet
device='N:Q120'
tolerance=1.E-2
imax=5.0
acnet.devzero (device, imax, tolerance)
```

where the quadrupole magnet `N:Q120` was set to zero after being degaussed, imax is the value of I_0 in the previous equation. The `tolerance` parameter corresponds to the difference between the measured and set value of I_k below which the magnet is set to its next current I_{k+1}. Such a simple implementation can actually be encapsulated in its own function and enable automatic degaussing every time a magnet is set to a new value. Such an approach was adopted in our script to ensure all quadrupole consistently remain on the right hysteresis branch to provide more reliability.

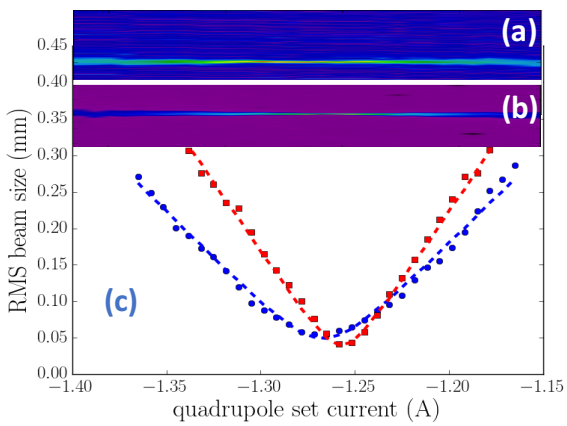

Figure 2: An example of quadrupole scan. Insets (a) and (b) show a 2-D image of the stack horizontal (a) and vertical (b) profiles. The associated rms beam sizes (in this example obtained from Gaussian fits of the profiles) are shown in (c).

Another important application of PYACL was the automatization of transverse-emittance measurement based on the quadupole-scan technique [5, 6]. The program scans a user-specified quadupole magnet, records the corresponding images formed as the beam impinged on a downstream screen, and provides the beam size (which are stored for further analysis). Additionally, the interfacing with ELEGANT enables the query of transfer-matrix elements (between the magnet and observation screen) as the quadrupole magnet is varied. The matrix elements are then used in the emittance analysis. An example of a quadrupole scan outputs are shown in Fig. 2.

BEAM-BASED CONTROL

When operating an accelerator it is often required to explore the beam response to a given perturbation and accordingly tune some of the beamline parameters. An example regards beam-based alignment in quadrupole magnets. Using PYACL we developed simple scripts that automatically center the beam in quadrupole magnets by (1) varying the magnet within a defined range, (2) measuring the resulting beam motion at a downstream beam-position monitor (via the `acnet.ReadAllBPMspec()` function), and (3) steering the beam accordingly. These three steps are repeated until the beam motion recorded at the BPM is minimized. In our implementation we use a conjugate gradient minimization algorithm available in SCIPY to optimize a pair of steerer upstream of the quadrupole magnet such that it minimizes the quadrupole-induced steering. This local correction can in fact be generalized to a global correction scheme where all the response matrix are measured and the steerer are tuned to globally minimize the steering in all quadrupole magnets within a specified region of the beamline.

INTERFACING WITH BEAM-DYNAMICS PROGRAMS

One of the most important aspects of PYACL is that it provides an interface between ELEGANT and ACNET. Several functions were implemented to enable the rapid simulation of current beamline settings or to rest the magnet according to an optimized ELEGANT simulation. The interfacing of ACNET to ELEGANT (that is passing current magnet settings to ELEGANT) is performed as follows. First the magnet currents are converted to normalized strength and save to a file using the function `fast.Machine2Elegant(EMeV,'fast.param')`. The output SDDS-compliant file (in the present case `fast.param`) is formatted for use in the ELEGANT `load_parameter` command. This provides a way to directly load the current beamline settings in ELEGANT. Conversely, it is desired to convert a magnet's settings to optimize using ELEGANT and set the values in the accelerator. This is accomplished in a similar way as described above using a function `fast.LoadFile2Machine(file, scale=1)`, where the `scale` parameter enable to arbitrary scale the magnet (e.g. to account for a change of energy) and `file` is an SDDS parameter file written by ELEGANT.

These functions were extensively used during the commissioning of the FAST facility and provided a method to seamlessly match the lattice given the measured Courant-Snyder parameter. Such an example appears in Fig. 3 where PYACL was used to match the beam to provide a small waist at the X121 YAG screen location. The beam was initially divergent and the corresponding spot size at X121 was on the order of $\sim 1 \times 1$ mm^2; see Fig. 3(b,c). The initial beam parameters were measured (using the previously described emittance measurement tool) and the quadrupole magnet optimized in ELEGANT to yield a waist at X121. The optimized betatron functions are shown in Fig. 3(d) and associated beam transverse density measured at X121 appears in Fig. 3(e).

DEVELOPMENT OF EXPERIMENTAL SEQUENCES

Beam experiments often rely on repetitive tasks that can be executed using scripts. The PYACL toolbox was used to develop procedures and assist in the measurement of transfer matrices. A specific example is our recent attempt to characterize the transfer matrix on a superconducting (SRF) TESLA-type cavity; see Ref. [7]. The script was used to automatically center the beam in one of the FAST SRF cavity [CC2 in Fig. 3(a)] using the same technique described above, but with a target function that minimizes the difference between beam centroid positions for different cavity phases. The beam positions were recorded for a large number of settings for dipole magnets (steerers). Such a measurement was automated and repeated for different operating phases to measure the transfer matrix of the cavity as function of phase [7].

IMAGE ACQUISITION & ANALYSIS

In linear accelerators, there is often a need to collect and analyze images from beam-density monitors such as scintillating (e.g. YAG:Ce) or optical transition radiation (OTR) screens. Additionally, these images need to be processed to extract statistical properties associated to the beam distribution. A typical example is the measurement of the transverse distribution of a beam such as needed to measure the transverse emittance. The PYACL toolbox includes provided features to grab images from the FAST frame grabber and store them as `numpy` arrays. The capability to acquire multiple image along with the automated acquisition of "background" images (i.e.turning off the photoemission laser and taking images) is also provided. The detailed analysis of the image is, however, performed in an independent package IMAGETOOL.PY. The package can be used to display images with associated projections; see Fig. 4, but also to extract its parameters. The rms beam sizes and coupling parameters can be obtained from various methods including Gaussian and SuperGaussian fits, or from a statistical analysis such as developed in Ref. [8]. This capability was used in emittance measurements; see Fig. 2. (c)

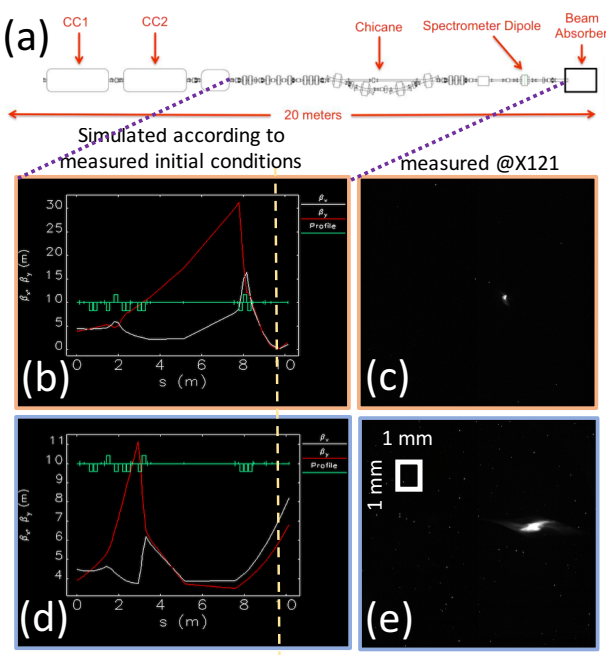

Figure 3: Overview of the interfacing of FAST injector beamline (a) and corresponding evolution of the betatron function before (b) and after (d) beam matching to achieve a small beam size at the location of X121. Image (c) and (e) are the corresponding measured beam density at X121.

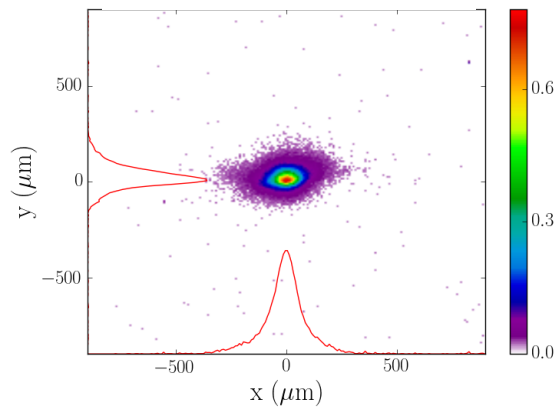

Figure 4: Image acquired and displayed with the IMAGETOOL.PY image analysis toolbox.

SUMMARY

In summary, we presented some applications of the PYACL toolbox. The toolbox enables users to rapidly develop scripts morphed with sophisticated analysis algorithms leveraging on the PYTHON scripting language. This toolbox, despite still being developed, has proven to be very useful during commissioning of the FAST facility to measure emittance, perform beam-based alignment, or take and rapidly analyze large data sets. Finally, PYACL also provides an seamless way of interfacing beam dynamic program with the operating accelerator. The toolbox in its current implementation is available via GITHUB [9] and is being further expanded. The `ImageTool.py` toolbox provides a generic suite of tools to analyze images from screens.

REFERENCES

[1] F. Nagy, G. Johnson, C. Briegel, "ACNET Network Services: VAX/VMS User's Manual," Software Documentation Memo No. 64.2, revision of March 16, 1994 available from Fermilab at https://www-bd.fnal.gov/controls/networks/vaxnetusr.txt (1994).

[2] J. Patrick, Proceedings of ICAP 2006, Chamonix, France p. 246 (2016).

[3] B. Hendricks, “ACL an introduction," Femilab BeamDocs 929-v3; available at http://beamdocs.fnal.gov/AD-public/DocDB/ShowDocument?docid=929 (2005).

[4] M. Borland, “Elegant: A flexible SDDS-compliant code for accelerator simulation," APS internal report LS-287 (2000).

[5] J. Rees, L. Rivkin, “On Measuring Emittances and Sigma Matrices," SLAC-PUB-3305, SLAC/AP-18 (1984).

[6] A. Green, “Development of a Python Based Emittance Calculator at Fermilab Science & Technology (FAST) Facility," MS Thesis Northern Illinois University; available as FERMILAB-MASTERS-2016-03 (2016).

[7] A. Halavanau, et al. to appear in the proceedings of the 2016 linac conference, in press (2016).

[8] F. Löhl, et al., Phys. Rev. ST Accel. Beams **9**, 092802 (2016); see also Y.-E Sun, PhD dissertation U. of Chicago (2005).

[9] see https://github.com/NIUaard