Developing an Improved Machine Model for ATLAS

Taylor Clarke

Mentor: Dr. Richard Pardo

**Introduction:**

As one of the most highly regarded low-energy nuclear physics facilities in the world, ATLAS can satisfy a great variety of beam requests for many different purposes. Because of this variability, computer simulations of the ATLAS beam can prove tedious to perform. To perform simulations of the beam, a computer program called TRACK [1] can be used. With this program, a user can define the order of the accelerator elements through which the beam passes, as well as many different parameters of these elements. Effectively, given some initial parameters about a particle distribution and some parameters of the accelerator, TRACK can simulate the passage of the particles all the way through the accelerator.

However, TRACK was not being used to its full capacity at ATLAS because it is difficult to simulate the actual accelerator configuration in the TRACK program. Understanding how TRACK works is not complicated, but to use it one must create an input text file consisting of many lines with many parameters on each line. Creating this file by hand can quickly become tedious. Because many of the parameters needed for TRACK are kept in a database on ATLAS, it was decided that a simple program and interface were needed so that TRACK could be run with greater automation by importing data directly from the accelerator database.

**TRACK:**

Before the program could be designed, it was necessary to learn the basics about the TRACK program itself. Described as “a code for beam dynamics simulation in accelerators…” [1], TRACK has the ability to simulate beam transport through many different types of elements that might appear in an accelerator, such as resonators, magnetic and electrostatic quadrupoles, magnetic dipoles, and so on. TRACK reads these elements along with their respective parameters, such as length or field strength, from an input text file called ‘sclinac.dat’. Along with ‘fi\_in.dat’, which contains the relative phases of the resonator cavities, ‘sclinac.dat’ contains all of the parameters about the accelerator itself. But one of the important features of TRACK is its ability to read electric or magnetic field data that correspond to actual or computed field distributions of accelerator elements. Therefore one can accurately simulate resonators, solenoids, or quadrupoles of many different types, as long as all the field data files are available for each type.

Aside from the information about the accelerator, TRACK also requires information about the beam to be accelerated. In a text file named ‘track.dat’, TRACK reads data values describing the incoming beam such as particle mass, particle charge, starting energy, and beam emittance. This is also where one might specify if the beam is bunched or DC. With these parameters along with the parameters of the accelerator, TRACK is able to plot the progression of the particle stream through the accelerator, at the end of each element giving the emittance ellipse in all three dimensions, the energy of the beam, and the phase of the beam with respect to the element.

Emittance is one of the most important characteristics of a particle beam and yet one of the most difficult characteristics to model or predict. A single point on an emittance graph gives both the position and the trajectory of a particle in one dimension, so three graphs (one for each dimension) can give a very adequate description of the size of the beam and whether it is focusing or defocusing. One of the advantages of TRACK is that it can measure emittance very precisely given accurate parameters for the accelerator elements. Knowing the emittance of a beam within the accelerator is useful for troubleshooting, which will be one of the future uses of this program.

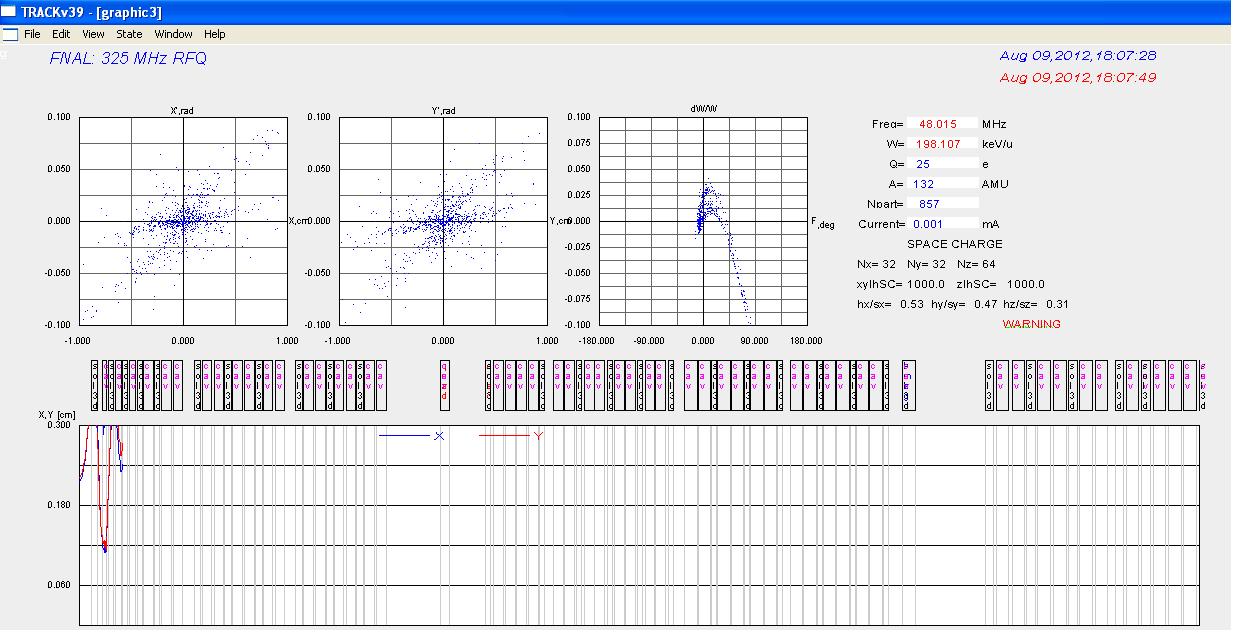


Fig. 1: TRACK output showing emittance graphs, beam energy, and other values

**Initial Decisions:**

After TRACK was thoroughly understood, it seemed logical to start designing the program and interface that would make the operation of TRACK much more efficient for ATLAS users. A few requirements for the program were developed. First, the human interface must be simple to use. Since TRACK is time-consuming to use for those not experienced with it, it would not make sense to create a program equally as tedious or difficult to master. Next, it must be able to use real-time data from ATLAS in order to simulate the current state of the accelerator. This was the main thrust of the project. Finally, the program must be compatible with the many types of computers that are used at ATLAS, so it must be as cohesive as possible.

**The Program:**

As soon as the requirements for the program were decided, the program was ready to be developed. It was decided that the program should be coded in C, since access to the ATLAS databases were most readily performed in C or FORTRAN. The ATLAS databases use Vista as a system for storing and accessing parameters of the accelerator. These databases store hundreds of values, ranging from quadrupole currents to cryostat temperatures. However, the databases did not contain all the values necessary to run TRACK. Luckily, creating additional elements in the realtime database in the Vista system is simple, so it was decided that for each accelerator element a list of parameters could be initialized in the database. These parameters could then be defined as whatever was necessary for TRACK, such as effective length or field strength. A combination of parameters already in use and the parameters we could define would prove sufficient to run TRACK.

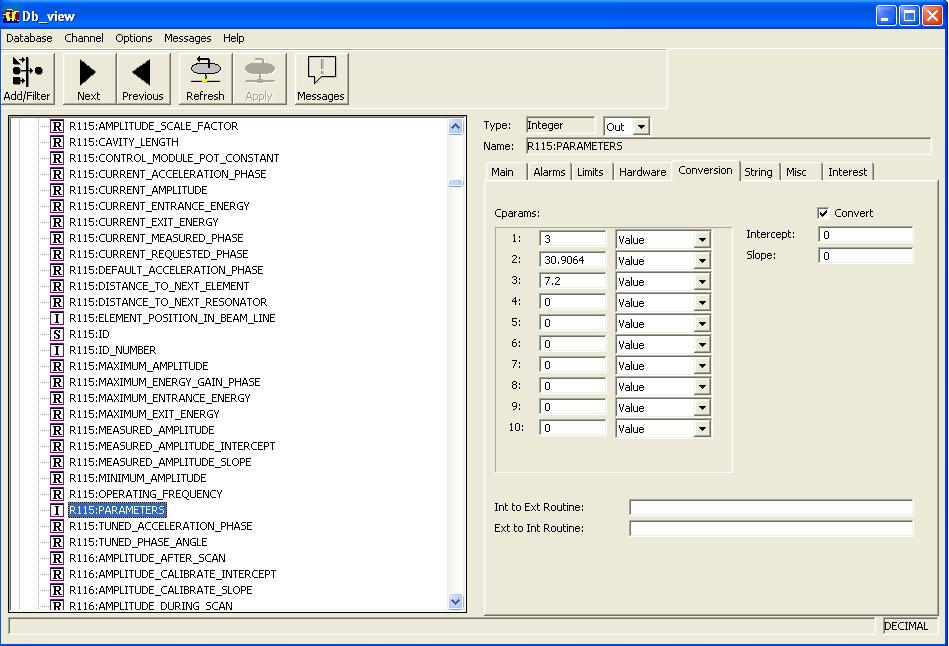


Fig. 2: Example of database showing parameters needed for program

After it was determined what parameters were needed, the program was coded. This program was developed in C using Microsoft Visual Studio so that it could easily be linked to the databases. The program was developed using a test server that held a previous instance of the ATLAS databases so that the values could be read and changed without interfering with ATLAS operations. To obtain the values from the database, routine access commands for Vista are employed in the code.

After the program obtains the values, it converts them to whatever form is required by TRACK. For instance, it uses a slope calibration, an intercept calibration, and a current to determine the field inside the quadrupoles, resonators, and solenoids. This field is then used as an input value for TRACK. Once the values are ready, they are written in the form required by TRACK to the ‘sclinac.dat’ file. The same process is used for the ‘track.dat’ file. In this case, however, an entirely new database was created that stores only the parameters required for TRACK. This database stores values describing the input beam and must be updated manually whenever the beam characteristics change. But the program still reads the values from the database and writes them in the correct format to the ‘track.dat’ input text file.

**The Interface:**

After designing the program, it was decided that a simple interface was needed in order to easily modify the values in the database. For many of the controls on ATLAS computers, Vista command interfaces are used, and this was found to be a sufficient method of controlling the parameters for TRACK. There are two interfaces: one for viewing the beam parameters, and one for viewing the accelerator element parameters. Theoretically one would not want to alter the accelerator element parameters actually used on TRACK, but one can create a mirror database with the current parameters and then alter those to test simulations with different parameters.

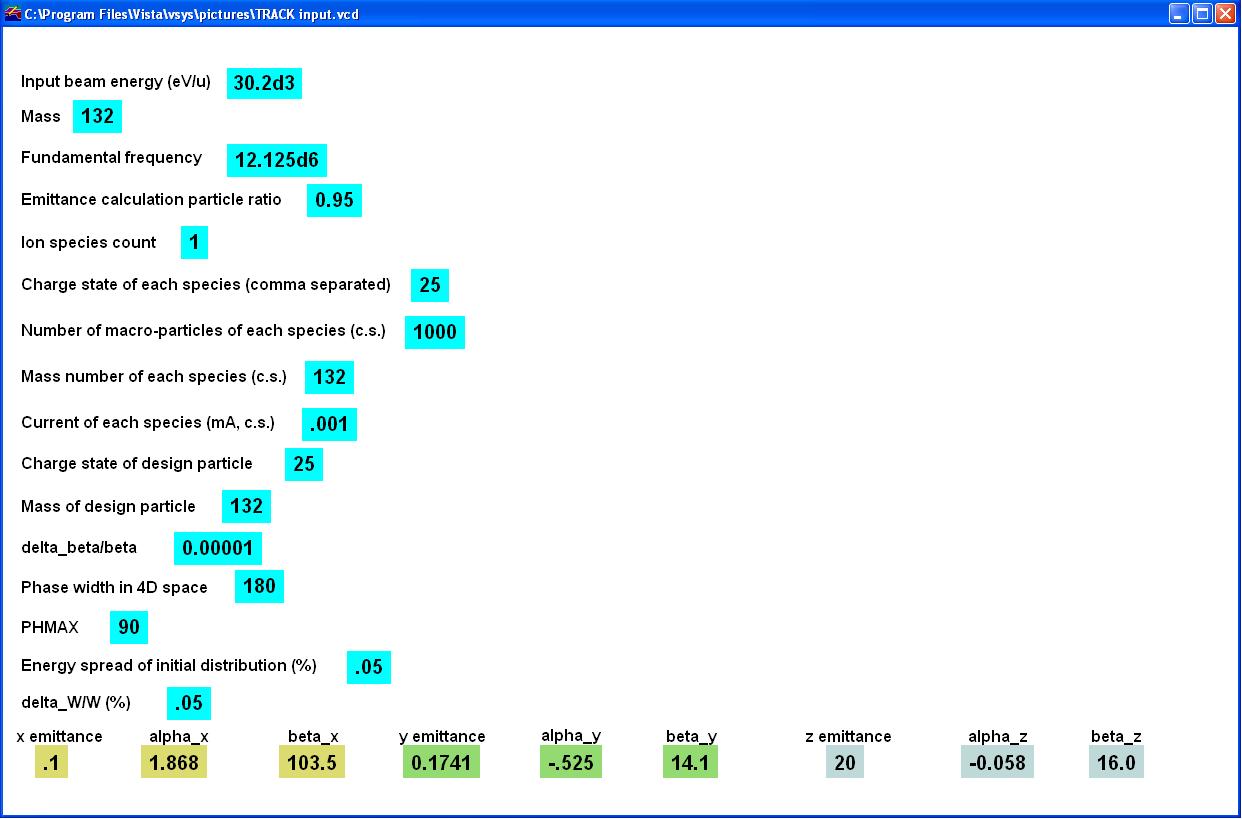


Fig. 3: Example of interface created to define parameters needed for TRACK input

**Conclusions and Future Work:**

Now that the tools have been developed to allow TRACK to be used more efficiently, experiments that utilize these tools can begin. Namely, scientists already using ATLAS will be able to better predict and understand their results. Not only will they be able to forecast the beam emittance and beam current at their target, they will be able to easily modify the accelerator parameters or test different beam designs to simulate different effects on their experiments. For the future, adding more features, such as functionality for more types of accelerator elements, would improve my program. It would also be beneficial to combine the disparate parts of my program into a more cohesive unit, in order to further streamline usage.

**Acknowledgments:**

I must thank my mentor, Dr. Richard Pardo, for being extremely welcoming to me and for always providing me with prudent and straightforward advice. Next, I am very grateful to Chris Peters, Maria Power, and Brahim Mustapha, without whom I would never have been able to get my project off the ground. I am greatly indebted to everyone with the Lee Teng Internship and USPAS, specifically Dr. Linda Spentzouris, Dr. William Barletta, Dr. Eric Prebys, and Elvin Harms, for making this summer one of the most educational and memorable I have ever had. To all of you and to all of my fellow interns, thanks for making this a fantastic summer and a great internship experience.

**Works Cited:**

[1] Ostroumov, P.N., V.N.. Aseev, and B. Mustapha. "TRACKv37." 2007. Print. <http://www.phy.anl.gov/atlas/TRACK/Trackv39/Manuals/tv39\_man\_index.html>.