Putting "Physics" in the CLEO Event Display

Terry Legette

Computer Science, Wayne State University, Detroit, Michigan, 48202

Abstract

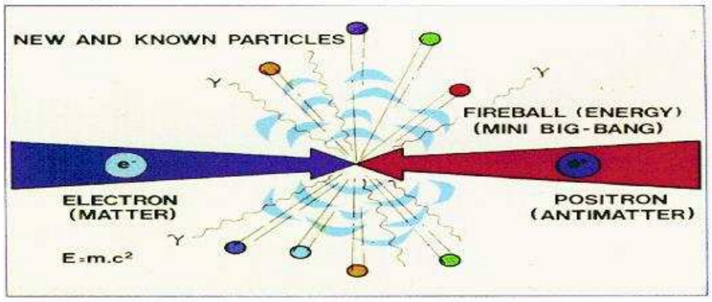
My project while physics based, was more computer science related. In an attempt to help out the curiosity of physicists when dealing with decays and the event in which it came, we began to develop a computer program that will perform all the calculations of possible decay combinations. By developing this graphic window, called a "View", it was our hopes to have an efficient manner in which the physicists can analyze data by viewing the physics behind each event. *inline*

Introduction

The problem that I assisted in addressing this summer deals with the viewing of decays from the CLEO III Detector that came from an event. While there are other "views" that currently exist to help physicists view the data from an event, Hierarchy and 2D Views, we wanted to create another view that allowed events to be viewed by the particles mass. One must keep in mind that the data that will be viewed from this program is simulated, but by inserting actual data we can observe just how close good our program works. This new view was to be created by using Qt, which is a C++ library that makes it easier to develop the graphic portion of our window. So now you can see that that the bulk of this project was developing, modifying, and correcting C++ code, which I had never used prior to this summer! This project can be best viewed by looking at the physics and computer programming aspects.

The Physics involved in this project

The purpose of this project is to add a new graphic display window view to CLEO's, the detector used at Cornell, visualization program. By careful analysis of the information gathered from the detectors, physicists can determine the speed, momentum, mass, and charge of a particle. For example, whether a particle has a positive or negative charge is known by the direction the particle curve. A physics event is when an electron and positron collide and from the energy of that collision a new set of particles is created.



event.eps

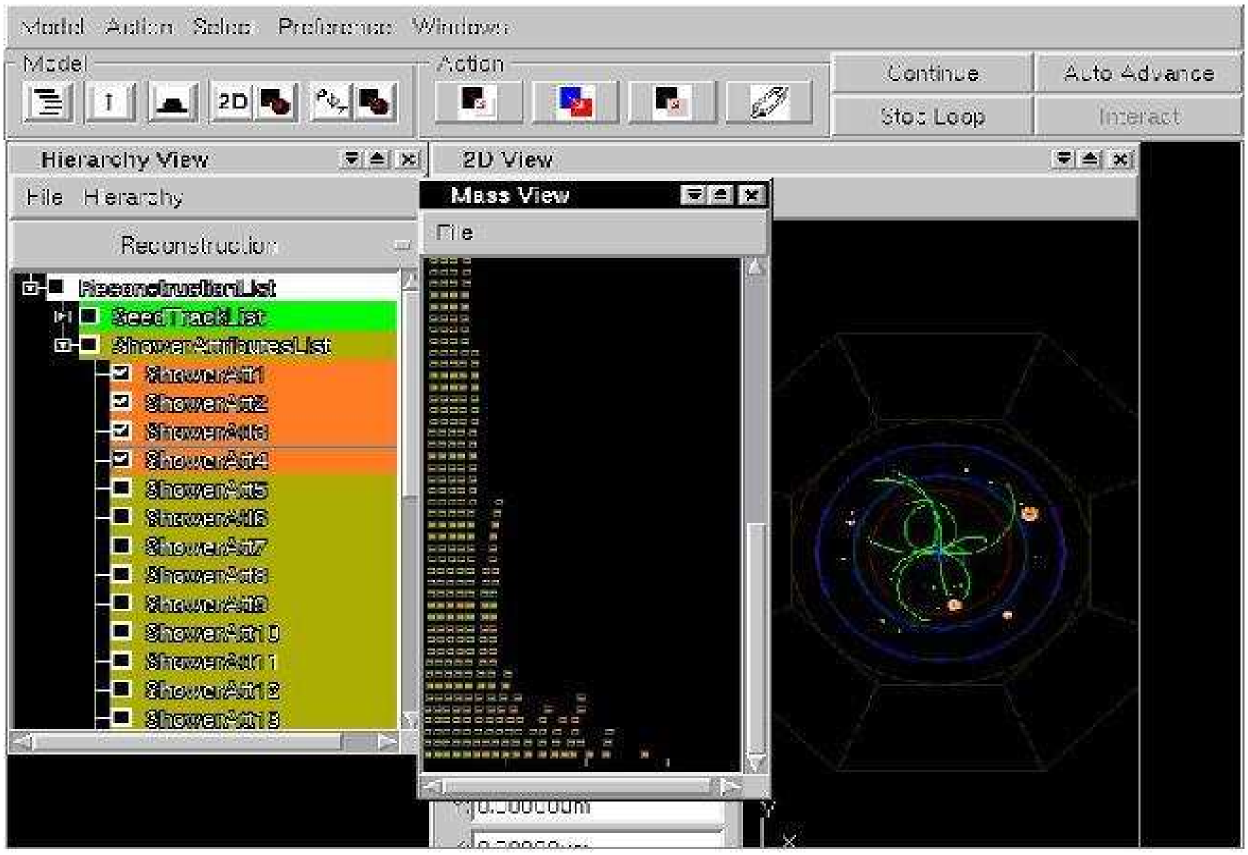
Usually these new set of particles have a short life span and they decay into other particles. A typical reaction might be the following:

B -¿ D* + pi
D* -¿ D + pi
D -¿ K pi

The above data shows a decay that resulted in a total of four particles, a K and three pi's, stemming from the original B. However, the original B was created with a mate an anti-matter B. Not only would our data contain the four particles from the original B, it would also include the number of particles the anti-matter B decayed into. So, what we want to do is to make it easier for physicists to find all possible combinations that could have resulted from the visible particles from the event to see if it may have come from the initial B. Since it would take a considerable amount of time to view, analyze and attempt to compute all of the possible combinations, it would only make sense to make use of technology by developing a program that does all the computations for the physicists. By developing such a program, it would save valuable time and make the computations less prone to human error when attempting to reconstruct or simulate the decays of an event.

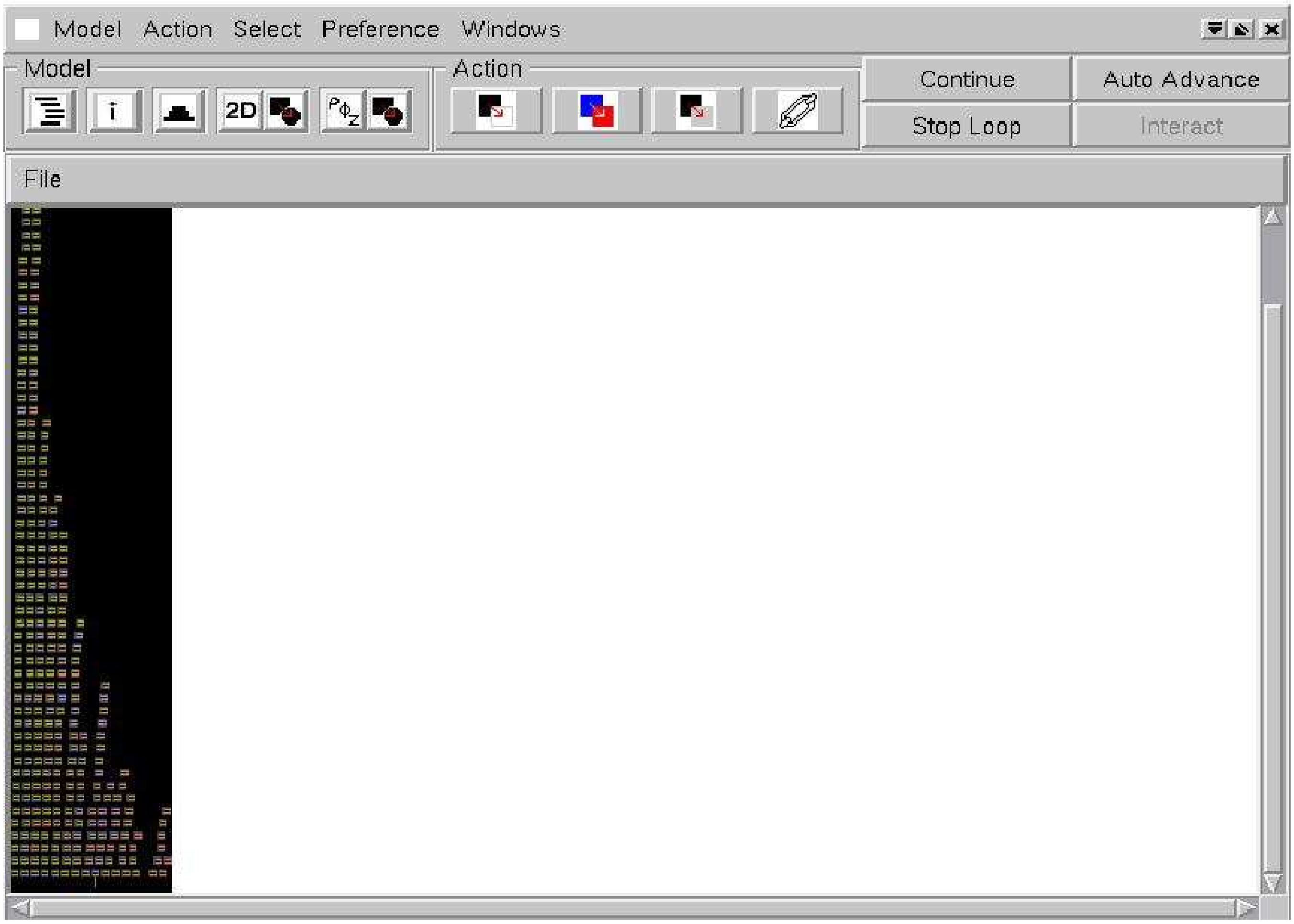
Computer Programming

Up until the recent past, most physics programs had been written using Fortran. However, to keep up with the advancements in technology and to become more user friendly, programs are being written using object oriented programming languages. The language that we used to create the view was C++ that seems to be the standard these days for



eventdisplay.eps

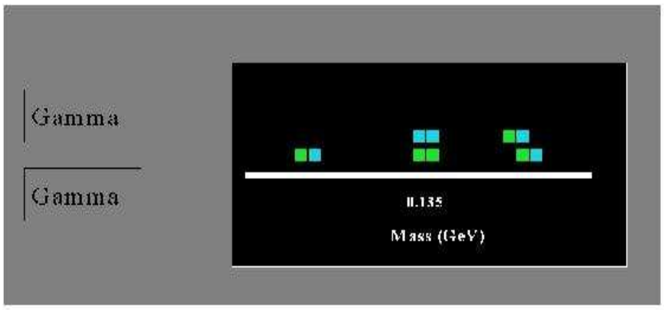
programs dealing with objects. Since there were two prior programs that had been created for viewing events, it made coding a new view slightly easier. Once the logic for the program had been thought out, we began by making sure we could obtain a list of the possible mass combinations by having the code print out the masses in text form to a standard terminal. Once that was accomplished, the next step was to develop the window to hold the graphics that would represent those masses and the possible combinations. To develop the graphics, it was decided that we would make use of Qt which is a C++ library that deals with developing graphical user interface's (GUI's). So, by making use of Qt, an Icon was created to go onto the toolbar that would represent the window in which we were creating. Following the creation of the menubar and window, it was time to make the masses appear in graphic form. Rectangles were used to represent the masses which was sorted from highest to lowest. After developing this code and having the masses appear in the canvas of the newly created window, we wanted to "flip" the axis's. Instead of being on the normal x and y axis's, going from bottom to top, the code's default was to have it go from top to bottom, which meant the code had to be corrected where it would resemble a normal x and y axis. Once the axis was changed, color was then added to the background of the canvas and the rectangles. Finally, to see if the window was working properly at this point, it was opened and used concurrently with the Hierarchy View and the 2D View.



Mass View.eps

Results

Although the current view does work with the others, it is quite a way from being completed. Due to time constraints, it does not have the same interactive capabilities as does the other views, and it does not respond to the mouse. However, plans are in place to continue working on this view to make it resemble the prototype as much as possible.



prototype.eps

Conclusions

I began this project with no knowledge of C++, now I can say that I have made progress with the help of my mentor, Chris Jones. As with the life of many programmers that revolve around the "compile, error, compile, run" cycle, I too had numerous frustrations when attempting to consider and code the logic necessary for this program. Once this project is completed I'm sure with the input of users this newly created "view" will perform as intended. tAlthough the current view, Mass View, does work with the others, it is quite a way from being completed. Due to time constraints, it does not have the same interactive capabilities as does the other views, and it does not respond to the mouse. However, plans are in place to continue working on this view to make it resemble the prototype as much as possible.

Acknowledgments

I am pleased to acknowledge Chris Jones, of Cornell University, and Prof. G. Bonvicin, of Wayne State University, who proposed this Research Experience for Undergraduates project and guided my effort...

This work was supported by a National Science Foundation REU grant.

Footnotes and References

References:

Chris Jones, Cornell University PROGRAMMING with C++, Schaum's Outlines, 2nd Edition Programming with Qt 1.4x and 2.0, O'REILLY