Summer 2001: Project Report

Anthony Del Frari

August 31, 2001

Contents

1 GEANT Exercises **3**
- 1.1 Purpose . . . 3
- 1.2 Overview . . . 3

2 Pair Production Beam Monitor **4**
- 2.1 Purpose . . . 4
- 2.2 Geometry . . . 4
- 2.3 Results . . . 4
- 2.4 Conclusion . . . 6

3 Backscatter background check for Neutron Array **7**
- 3.1 Geometry . . . 7
- 3.2 Testing . . . 7
- 3.3 Conclusions . . . 9

4 Visit to Duke **10**

5 Neutral Pion Spectrometer Simulation **11**
- 5.1 Purpose . . . 11
- 5.2 Overview . . . 11
- 5.3 Procedure . . . 11
  - 5.3.1 Materials . . . 11
  - 5.3.2 Geometry . . . 12
  - 5.3.3 Pion reaction simulation . . . 12
  - 5.3.4 Resolution Factors . . . 12
- 5.4 Results . . . 14
  - 5.4.1 Effect of distance from Target . . . 14
  - 5.4.2 Effect of different geometry designs . . . 14
  - 5.4.3 Effect of different analysis methods . . . 15
  - 5.4.4 Pion Angular Resolution . . . 15
- 5.5 Conclusion . . . 16

6 XLucid Interface Design **18**
6.1 Purpose 18
6.2 Methods 18
6.3 Specific Issues 19
6.3.1 Main interface control buttons 19
6.3.2 Histograms 19
6.3.3 The histogram display 20

A Reaction Kinematics **22**
A.1 Generating the Decay Photons 22
A.2 Pion Reconstruction 23

B The Pion Program Files **25**
B.1 Fortran Files 25
B.2 Kumac Scripts 26

Chapter 1

GEANT Exercises

1.1 Purpose

The purpose of this portion of the project was to learn the Fortran programming language and to become familiar with the GEANT simulation package.

1.2 Overview

These exercises involved entering the GEANT code and extracting results. There were nine exercises.

1. Manipulation of Basic Variables: No Comments
2. Energy Loss Measurement: No Comments
3. Absorption Measurement: No Comments
4. Energy Deposition, etc: No Comments
5. Change Beam Direction: No Comments
6. Reducing the Acceptance: No Comments
7. Finite Beam Spot: No Comments
8. Finite Beam Divergence: No Comments
9. Beam Monitor: No Comments

Chapter 2

Pair Production Beam Monitor

2.1 Purpose

The purpose of this GEANT simulation was to determine the number of pairs that will be detected, and the best setup to detect them. This is done as a correction factor to a Compton scattering experiment.

2.2 Geometry

The geometry consists of a copper target and three 32cmX32cm detector packages (See figure 2.1). Each detector package is made up of a NE102 paddle counter, a NaI counter in an aluminum casing. The detector packages are setup with one on the axis and two symmetrically off-axis. The on axis detector is four meters from the target and the off-axis detectors are at three meters.The off-axis detectors have iron collimators with a seven inch diameter.

2.3 Results

The simulation was conducted with 5 MeV and 10 MeV gamma rays incident on the target. The data was collected in two ways. The first way was to use the fact that this is a simulation and simply count the electrons, positrons, and gammas as they strike the paddle counters. The program then determines if an electron and a positron have entered the opposite off-axis detectors and calls that a pair. The other method is more realistic, using the energy deposition in the paddle counters and call anything above a certain threshold a charged particle (i.e. an electron or a positron). Energy deposition histograms suggest that this threshold be placed at 0.0003 GeV(300keV). Both of these methods

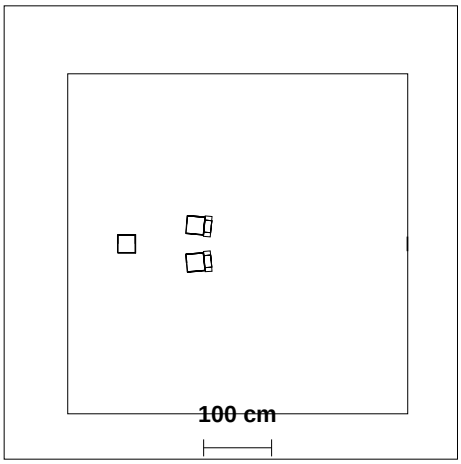


Figure 2.1: Pair Production Geometry

were actually counted using manual counters inside of the simulation. No post simulation analysis was done.

Three different sets of measurements were taken. The first two measurements were taken at off-axis angles of five and ten degrees.For the five degree simulation, the detectors were placed at a distance (3m from the target) such that they started to infringe on the gamma ray beam. The ten degree measurement was conducted to get an idea of the angular spread of the pairs. The last measurement was also conducted at ten degrees, but the detector was moved closer to the target so that they were at the same distance off-axis as the five degree case. This distance turned out to be 150.57cm from the target.

The results are in the following table. The results are normalized to per one million events. The numbers in parentheses are the 5 MeV results

Trial	1	2	3
Events(millions)	68(10)	110(60)	10(10)
deltaE Counts	669(453)	285(141)	1214(585)
Pairs(E)	8.16(2.1)	0.64(0.25)	6.2(2.3)
Positrons	151(27)	71(7.73)	323(48)
Electrons	503(358)	205(109)	867(445)
Gammas	2133(1653)	479(395)	3154(2382)
Coincidence	8.07(1.5)	0.57(0.1)	6.3(1.3)

Table 2.1: Results of the Trials (normalized per million)

These results show that while there is a significant amount of spread involved most of the pairs go very straight down the center. The smaller number of

particles detected in the second simulation shows that the density of the particles goes down at the higher angles. This explains why the third simulation has fewer pairs than the first even though it encompasses a larger solid angle. This shows that if the first setup with the off-axis detectors at five degrees will detect the most pairs. The numbers also indicate that the 10MeV gamma rays are more efficient at producing pairs than the 5MeV gamma rays are. The 5MeV gammas produce fewer particles of all types, especially positrons.

The above results are for for a target of a sixteenth of an inch thick. Measurements were taken at the five degree off-axis setup for target thicknesses of an eighth and a quarter of an inch.

Thickness	1/16	1/8	1/4
Events(millions)	68(10)	10(10)	10(10)
deltaE Counts	669(453)	747(446)	718(365)
Pairs(E)	8.16(2.1)	8(3.3)	6.7(2.8)
Positrons	151(27)	154(28)	150(25)
Electrons	503(358)	508(344)	478(317)
Gammas	2133(1653)	2531(1892)	3304(2360)
Coincidence	8.07(1.5)	7.8(2.5)	6.5(2.4)

Table 2.2: Results for different thicknesses (normalized per million)

The above results show that the number of charged particles produced is unchanged by the thickness of the target. The number of gammas produced increases as the thickness of the target is increased. The number of pairs produced decreases as the thickness of the material is increased. This may just be from the positrons annihilating in the thicker material but then one should see a change in the number of positrons detected which is not seen. The most pairs are detected at the 1/16in. thickness.

2.4 Conclusion

From the above results the most pairs will be detected with the five degree setup and the sixteenth inch target thickness.

Chapter 3

Backscatter background check for Neutron Array

Reference: http://nucleus.usask.ca/Geant/Summer2001/nback.html

3.1 Geometry

The geometry consists of a hollow sphere of BC505 with inner radius of 40.64cm and an outer radius of 50.64cm. Cylinders of air are placed in either side of the sphere to allow the beam to travel through it. The cylinder's radius is defined so that they come through the inner surface of the sphere at a angle of 25 degrees from the horizontal, this turns out to 17.18cm. Then a 4in. thick lead plate is placed in front of the beam 5m from the center of the sphere. Behind this a NaI detector is placed inside of a aluminum casing. Between the sphere and the lead plate a 10cm thick lead shield is used to absorb the backscatter. The shield is a square plate as large as the sphere. A 10cm square hole in placed in this shield to allow the beam to pass through it. The beam used is a 5MeV gamma ray beam. A 30keV energy deposition cutoff is placed on the BC505. The beam passes through approximately 1m of air before passing through the sphere. As shown in figure 3.1.

3.2 Testing

First a run was conducted without the shielding in place. Then the shielding was placed at various positions to see the varying amount of backscatter absorbed. Without the shield in place 4846/million hits were detected in the BC505. Almost all of this radiation came from the gamma ray beam interacting with the air it is passing through. The simulation was then run with the air replaced by a vacuum. When this was run without any shielding the BC505 detected 847/million.

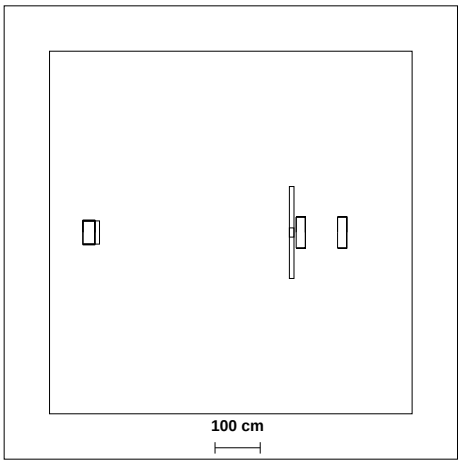


Figure 3.1: Neutron Backscatter Geometry

Next, the shielding was moved to different positions to determine the best placing of it to reduce the backscatter. The shield was placed in three positions: right against the lead plate, 310cm from the center of the sphere, and approximately 10cm from the edge of the sphere. The first position had almost no effect on the amount of backscatter with 837/million detected. The second position 3/million were detected by the BC505. The last position did the best with only 8/10million being detected. These three were conducted in a vacuum. All further runs use the shielding in the third position.

To then isolate the affect of the air in front of the sphere from the air behind the sphere columns of air 10cm square were alternately placed in these positions. The column of air in front of the sphere produced 2317/million. The column behind the sphere produced 27/million. As an alternative the air in these columns was replaced by helium gas and the tests done again. With He the front produced 298/million hits. The rear column of He produced 4.8/million.

Test	Description	Hits(per million)
1	Original with air and no shielding	4846
2	Vacuum and no shielding	847
3	Vacuum, shielding as far back as possible	837
4	Vacuum, shielding 310cm from center of sphere	3
5	Vacuum, shielding 10cm from edge of sphere	0.8
6	Test 5 with column of air before sphere	2317
7	Test 5 with column of air after sphere	27
8	Test 5 with column of He before sphere	298
9	Test 5 with column of He after sphere	4.8

Table 3.1: Results of the tests

3.3 Conclusions

The amount of air the beam passes through before passing through the BC505 sphere has a large impact to on the number of counts. The best position for the shield to reduce backscatter is as close to the sphere as possible. The introduction of He to replace the air along the beam line causes a fairly large reduction in the number of hits detected.

Chapter 4

Visit to Duke

The purpose of this trip was to experience the joys of experimental physics first hand. The trip was to the High Intensity Gamma Source (HIGS) at the Duke Free Electron Laser Laboratory (FELL) at Duke University in Durham, North Carolina. The experiment being run used a piece of equipment known as "Blowfish" this was a spherical array of 88 scintillator detectors surrounding a target. The experiment was looking for asymmetries in detected neutrons that were created by a polarized gamma ray hitting the target.

I performed several duties while I was there. These included helping to run the experiment(starting/stopping data acquisition), doing calibration tests, and helping to search the equipment for light leaks. The calibrations were done so that the collected data could matched with an energy value. The light leak checking ensured that the correct amount of energy was detected.

Chapter 5

Neutral Pion Spectrometer Simulation

5.1 Purpose

The purpose of this project is to simulate the operation of a spectrometer to detect neutral pions. This is done to determine the detector resolution and efficiency in detecting the creation of neutral pions. The detector itself is being design to compare experimental results to predictions of QCD.

5.2 Overview

There are two types of detectors involved: cesium iodide (CsI) and lead glass (PbGl). The CsI detectors come in two prebuilt arrays of sixty individual blocks. There are another sixty lead glass blocks that can be arranged in any fashion.

5.3 Procedure

The simulations are done in the Fortran programming language using the CERN GEANT particle simulation package.

5.3.1 Materials

The two materials used are not perfect. The PbGl in particular has poor energy resolution. This is due to the fact that it is a Čerenkov detector. The GEANT package supports the generation of Čerenkov radiation but we were unable to get it to work. Thus as a simple method of simulating this effect the energy deposition in the PbGl was smeared with a Gaussian with σ=15%. The CsI is a much better detector, but it also is not perfect and so it was smeared with σ=2.5%.

5.3.2 Geometry

The CsI detectors come in prearranged 6X10 arrays. The PbGl detectors are free and and can be arranged in anyway chosen. To be approximately the same size as the CsI arrays they are grouped in arrays 3X5. This means up to 4 arrays can be made of the PbGl. When placed together in the spectrometer these arrays are called walls.

Three different geometries were constructed and tested as shown in Figure 5.1. The first geometry places the CsI walls 90° apart, two PbGl walls are placed opposite them. The second geometry places the CsI walls facing each other with the PbGl 90° apart from the two CsI walls. The third geometry uses the same CsI placement as the second but it brackets the CsI walls with two PbGl walls as sort of catchers.

5.3.3 Pion reaction simulation

The reaction studied here is a two step process. First a gamma ray impacts a target nucleus producing a neutral pion. Then the pion decays into two decay photons which are actually detected. The first part of the reaction was not simulated. Instead first the pion was randomly assigned a direction and an energy and then from that the two decay photons where created.

This simulation uses an input gamma ray energy of 155MeV and a deuteron target. This means that the neutral pion created will be low energy ($m_\pi = 135MeV$). Therefore the pion will have very little momentum. Thus to conserve momentum the decay gammas will come out going in almost opposite directions.

5.3.4 Resolution Factors

There are several factors that have to be taken into consideration in the simulation. The first is that the decay gammas will come out in almost opposite directions. Next, the energy resolution of the PbGl is very poor. Lastly, the angular resolution will be better with the detectors farther from the target.

The first factor is studied by the three different geometry setups. The third factor was tested by changing the detector distance in the first detector geometry Csi90°. The second factor is dealt with by calculating the pions mass by different methods. Two main methods were used. The first includes the energy of the PbGl and the CsI detectors. The second method removes the PbGl energy deposition from the equation and combines the incident gamma ray energy with the CsI energy deposition and the angles from both. This method uses the formulas below.

$$E_\pi = \frac{2k_A^2 - 2k_A^2 \hat{k}_A \cdot \hat{k}_B - E_0^2 + M_N^2}{2[k_A(1 - \hat{k}_A \cdot \hat{k}_B) - E_0]} \tag{5.1}$$

$$E_0 = k + \sqrt{k^2 + M_N^2} \tag{5.2}$$

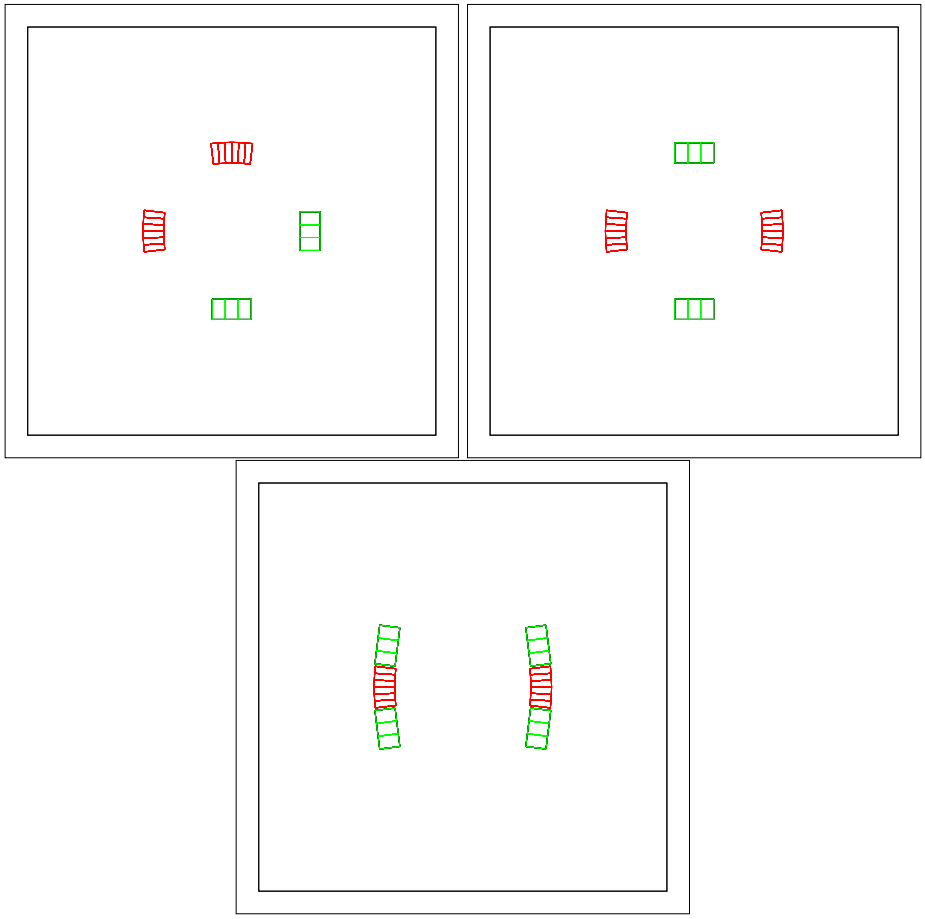


Figure 5.1: The Pion Spectrometer Geometries. The red walls are CsI, the green are PbGl. The geometries labeled left to right are: Csi90°,Csi180°, and Catchers

$$p_\pi = \sqrt{E_\pi^2 + E_0^2 - 2E_\pi E_0 - M_N^2} \tag{5.3}$$

5.4 Results

5.4.1 Effect of distance from Target

The distance from the target to the walls has two major effects. First, it increase the angular resolution of the array as the the individual detectors take up a smaller solid angle. Secondly, because the detectors take up a smaller solid angle the array is far less efficient at detecting the pions. The results of a distance test on the first geometry setup Csi90° is shown below in Table 5.1.

Distance(m)				
CsI	PbGl	FWHM(MeV)	Mean(MeV)	Total Counts(/million)
.29	2	10.5	0.5939	80670
1	2	4.0	1.046	4480
1	1	5.0	1.272	17756
2	2	3.5	0.1398	1001

Table 5.1: Effect of Wall Distance on missing mass spectra

From Table 5.1 one can see that the wall distance has a major effect on both the resolution(FWHM) and the total counts. From the table above as a compromise between resolution and count efficiency all the runs with the other geometries where run with walls at 1m from the target.

5.4.2 Effect of different geometry designs

The different geometry designs are mostly meant take advantage of the high resolution CsI detectors. Only those events where one of the decay photons hit a CsI detector are accepted as valid events. Table 5.2 below summarizes the results.

Geometry Setup	FWHM(MeV)	Mean(MeV)	Total Counts(/million)
1. CsI 90°	5.0	1.272	17756
2. CsI 180°	5.5	-1.661	9695
3. Catchers	6	0.9344	49213

Table 5.2: Effect of different geometry designs on missing mass spectra

The table above shows that the Csi90° geometry has the best energy resolution while the Catchers geometry has the greatest detection efficiency.

5.4.3 Effect of different analysis methods

There are two ways to analyze the data the first uses the energy of both decay photons and their angles. The second uses only the energy of one of the decay photons and the incident gamma ray energy. This allows you to remove the smearing due to the poor energy resolution of the PbGl. These two methods were tested the results are shown in figure 5.2.

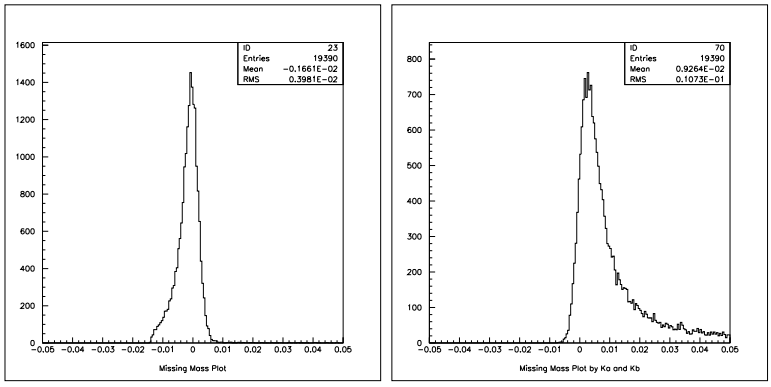


Figure 5.2: Effect of different analysis methods

One can see that the second method greatly increases the resolution of the calculation. This method was used in all of the other tests. The energy-energy calculation is very bad. The above calculations were done for the second geometry setup Csi180°.

5.4.4 Pion Angular Resolution

It is desired in this experiment to be able to observe angular asymmetries for the pion. Thus it is important to know the pion angular acceptance of each detector geometry. In the simulation in actual pion angular spectra are even in ϕ and in $cos\theta$.

As seen in figure 5.3 none of the geometries is able to reproduce the $cos\theta$ distribution.However the last geometry is the smoothest as appears to due the best job of detection. The first two geometries produce odd forward peaked theta spectra. Figure 5.4 shows that the Catchers geometry has the smallest error. In particular it removes almost completely the odd structure occurring at $\theta = 1rad$ in the error spectra.

Figure 5.5 shows the phi response of the three detector geometries. The first geometry has a very odd response. It is asymmetric with hight peaks where the CsI detectors are and low peaks where the PbGl detectors are. This is probably due to the preference to events that include the CsI detectors. The Csi180° geometry has two high peaks where the CsI detectors are and essentially nothing in between. The last geometry, Catchers, also has peaks where the CsI

detectors are but it also has a finite number of counts in the gaps where there are no detectors. The error plots shown in figure 5.6 show that the last geometry has a slightly wider base error than the other two. This is likely due to its increased use of the poor angular resolution of the PbGl detectors.

5.5 Conclusion

The different geometries have different advantages. The first, Csi90° has the highest resolution allowing for the lower energy features to be seen. However, it is not very efficient. The Catchers geometry setup loses slightly in the resolution but gains highly in the efficiency (2.77 times). This would allow the experiment to be run for a fair amount less time. The detectors are more accurate the farther they are from the target but space and efficiency constraints put limits on this. A reasonable compromise on both leads to accepting the detectors at 1m from the target. The energy resolution of the PbGl cells is really bad thus removing this number from the calculation greatly increases the resolution of the array. The Catchers geometry provides the greatest angular pion acceptance and would thus show asymmetries more clearly.

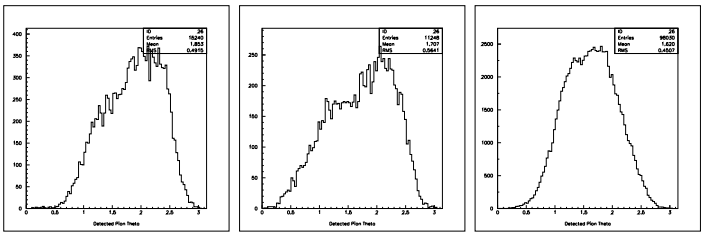


Figure 5.3: Pion Theta Spectra

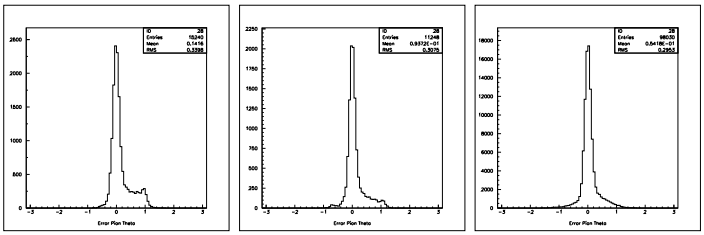


Figure 5.4: Pion Theta Error Spectra

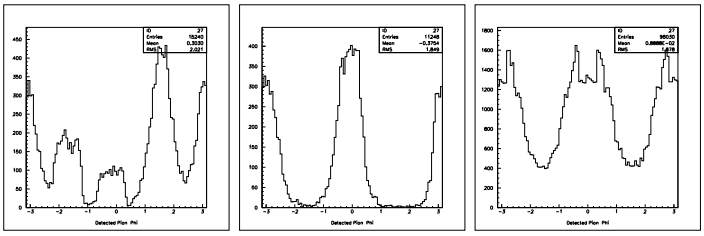


Figure 5.5: Pion Phi Spectra

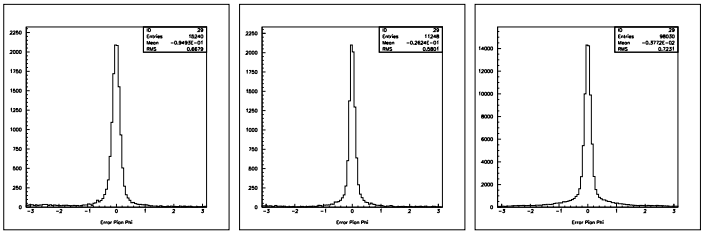


Figure 5.6: Pion Phi Error Spectra

Chapter 6

XLucid Interface Design

Reference: The gtk home page: http://www.gtk.org
The glade home page: http://glade.gnome.org

6.1 Purpose

The interface to the Lucid data acquisition system(XLucid) was originally created on Sun workstations using the Window Manager package XView. The system is now being run on Linux boxes running Red Hat 6.2. This means that it will not run without installing the XView packages and when it does the hacks to get things to work in XView do not all work in GNOME or KDE desktop environments. The group would like to know how difficult it would be to port the system to GNOME(gtk).

6.2 Methods

The windowing package for the GNOME window manager is known as gtk. The creation of a graphical user interface (GUI) in gtk can be done graphically using the program GLADE. GLADE allows the programmer to visually lay out the interface and it can then generate the C code in gtk that will create this interface. GLADE can also generate the code in C++, Perl, or Eiffel if one of those languages is preferred.

GLADE generates four main files that are used in the GUI. The first, main.c, contains the main function of the program. It takes in user arguments which it passes to the gtk system, it then creates the first window, shows it, and starts up the gtk loop program that will wait for events the occur.

The second file is interfaces.c. This file is where the code to create each window widget is stored. This is where the interface is actually created. It contains a creation function for every window that you design in glade. Caution: Glade overwrites this file so any changes you make to it are non-permanent. This means that if you want change something dynamically to a window on

each creation it cannot be done here without some effort. One way of getting around this is to create a wrapper function around the creation function for each window in a separate file. This will be tedious if the number of windows grows large. The other option is to use a signal that is raised when a window is created and put the dynamic code in its callback function. However, I don't think that there is a "creation" signal, but there are signals "draw","show",etc that may be raised during or after creation. In particular the "show" signal is probably sent before the window is shown.

The third file that glade creates is support.c. This file contains a number of useful functions that ease the use of glade and the creation of a GUI. As with interfaces.c this file is overwritten by glade.

The fourth file is callbacks.c. This file contains all the signal handlers for the GUI that are created by glade. This file is not overwritten. This is where the actions of the GUI are done and where custom code should go. Each widget created in glade has a "signals" tab in the properties window. Any signals created here will be given a function in callbacks.c.

In addition to the files created by glade three other files were created for the test implementation. The first of these as mentioned before is windowwrappers.c/h. This is the file where the wrapper functions for the window creation were written. The file enterhistos.c/h is where the histogram names are added into the CList for the data window. The last file is windows.h this header file contains the definitions for the histogram structs, the array to store them, and a pointer for every window in the project. The pointers allow for easy testing if a window has been created and provides a convenient place to store the window pointers.

6.3 Specific Issues

6.3.1 Main interface control buttons

It is desired that the main interface buttons interchange like radio buttons. Infact, in the test implementation they are radio buttons. However, the default signal for these is a "toggle" signal for each button. This signal is then raised whenever the state of a button changes. This means that when you press a button the signal is raised for that button and whichever button was previously selected. Some ideas for getting around this is to use a "pressed" or a "released" signal instead. Another idea is to use one callback function for all of the buttons, and then to carry out the actions based on which widget raised the signal and what its previous state was.

6.3.2 The histogram list and creation of the windows

One of the most used portions of the XLucid interface is the histogram displaying available from the data window. This window features a list of histograms that can be displayed and selected.

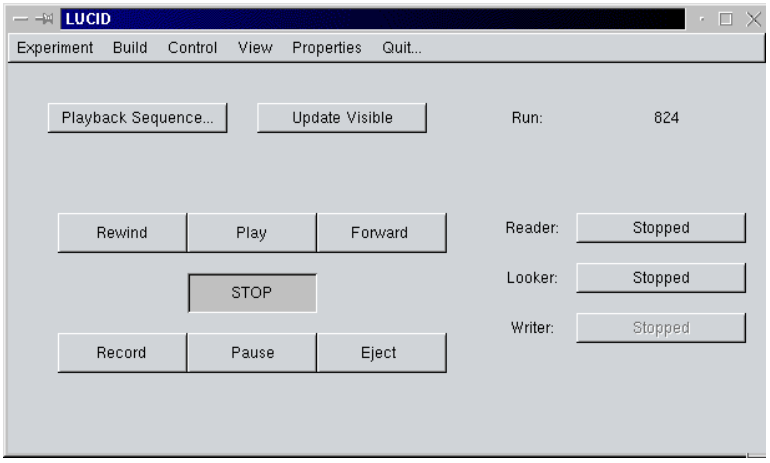


Figure 6.1: The main interface to XLucid in gtk

In the test implementation this list was created as a CList or Columned-List. The addition of items to this type of list is done by simply supplying a string to a function. There are signals for selecting and unselecting items that allow for the histograms to be kept track off. The test implementation used an array of special structs to keep track of the selected histograms and the windows that went with them. However, the CList specification allows for data to be attached to items through pointers. This should allow for a more efficient method of keeping track of the histograms.

One of the major issues with the old XLucid implementation in the GNOME environment is that when a histogram window is destroyed (i.e. by pressing the "X" button at the top of the window) it is not unselected from the histogram list. Gtk allows for this problem to be easily solved. When a widget is destroyed it raises a "destroyed" signal. By attaching a callback function to these signals for histograms windows the histogram list can be changed when a histogram window is destroyed.

6.3.3 The histogram display

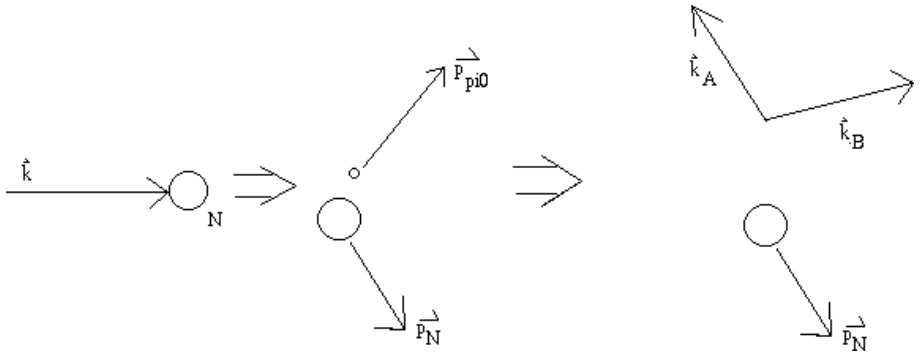
The hardest part of porting the XLucid interface to gtk is probably going to be the creation of a custom widget for displaying the histograms. The main difficulty in doing this depends on how the histograms are drawn in the old implementation. If they are drawn by direct calls to X then it will probably not be that hard. However, if they are drawn by functions in XView or related to it then this will have to be ported too. This will involve changing function calls to XView into calls to the lower level gtk drawing package known as gdk. Other than this the creation of new signals is fairly simple and should not be a major problem. The gtk tutorial recommends that you study the code for

the precreated gtk widgets to get an idea of how a widget is created. It also recommends that you start as a basis with a precreated widget and build on top of it.

Appendix A

Reaction Kinematics

The reaction studied in this simulation is a two part reaction. First the incident gamma ray impacts a target nucleus producing a neutral pion then the pion decays into two photons.



A.1 Generating the Decay Photons

Conservation requires that:

$$k + M_N = E_\pi + E_N \tag{A.1}$$

$$\vec{k} = \vec{p}_\pi + \vec{p}_N \tag{A.2}$$

By isolating the nucleus' variables, squaring then and then subtracting $(E_N^2 - p_N^2)$ one gets:

$$M_\pi^2 + 2kM = 2[E_\pi(k + M_N) - kp_\pi cos\theta_\pi] \tag{A.3}$$

The following substitutions one can simplify the equation.

$$\epsilon_1 = \frac{M_\pi^2 + 2kM}{2(k + M_N)} \tag{A.4}$$

$$\beta = \frac{k}{k + M_N} \tag{A.5}$$

One can then replace p_π by its energy equivalent, and square on gets

$$\beta^2(E_\pi^2 - M_\pi^2)cos^2\theta_\pi = E_\pi^2 - 2E_\pi\epsilon_1 + \epsilon_1^2 \tag{A.6}$$

Solving for E_π one gets:

$$E_\pi = \frac{\epsilon_1 + \beta cos\theta_\pi\sqrt{\epsilon_1^2 - M_\pi^2(1 - \beta^2 cos^2\theta_\pi)}}{1 - \beta^2 cos^2\theta_\pi} \tag{A.7}$$

Then the pion's momentum is given by:

$$p_\pi = \sqrt{E_\pi^2 - M_\pi^2} \tag{A.8}$$

It's direction is chosen randomly. Next one must calculate the momentum of the decay photons. In the pion's center of mass the momentums of the two photons are simply:

$$k_A = k_B = \frac{M_\pi}{2} \tag{A.9}$$

The direction of one photon is chosen randomly, while the other is then chosen to conserve momentum. Then the photons have to boosted back into the lab frame.

The boost equations from a frame A to a frame B moving with $\vec{\beta}$ with respect to A are given by:

$$E_B = \gamma(E_A - \vec{\beta} \cdot \vec{p}_A) \tag{A.10}$$

$$\vec{p}_B = \vec{p}_A + \gamma\vec{\beta}\left(\frac{\gamma\vec{\beta} \cdot \vec{p}_A}{\gamma + 1} - E_A\right) \tag{A.11}$$

These can be performed by the GEANT function GLOREN. Then the photons have to be rotated from the pion frame to the lab frame. This can be done with the GEANT function GVROT. This completes the photon generation.

A.2 Pion Reconstruction

The simplest method of reconstructing the pions energy and momentum is by taking advantage of their conservation. This gives the equations:

$$E_\pi = k_A + k_B \tag{A.12}$$

$$\vec{p}_\pi = \vec{k}_A + \vec{k}_B \tag{A.13}$$

This is very simple but it means that you have to include the poor energy resolution of the PbGl. To deal with this another method is needed.

First notice that in the reaction center of mass frame you have the equations:

$$k + E_N = E_\pi + \acute{E_N} \tag{A.14}$$

$$\vec{k} + \vec{p}_N = \vec{p}_\pi + \vec{\acute{p}_N} = 0 \tag{A.15}$$

If one isolates the nucleus' variables and combines the above equations one gets:

$$E_\pi = E_0 - \sqrt{p_\pi^2 + M_N^2} \tag{A.16}$$

$$p_\pi^2 = E_\pi^2 + E_0^2 - 2E_\pi E_0 - M_N^2 \tag{A.17}$$

where

$$E_0 = k + \sqrt{k^2 + M_N^2} \tag{A.18}$$

Now in the reaction center of mass the pion decay gives equations:

$$E_\pi = k_A + k_B \tag{A.19}$$

$$\vec{p}_\pi = \vec{k}_A + \vec{k}_B \tag{A.20}$$

Let $\hat{k_A}$ be the unit vector for k_A. The squaring A.20 and solving for k_B one gets:

$$k_B = -k_B \hat{k_A} \cdot \hat{k_B} \pm \sqrt{k_A^2[(\hat{k_A} \cdot \hat{k_B})^2 - 1] + p_\pi^2} \tag{A.21}$$

If one then substitutes this into A.19 and then replaces $p_p i^2$ by A.20 one can solve for E_π to get:

$$E_\pi = \frac{2k_A^2 - 2k_A^2 \hat{k}_A \cdot \hat{k}_B - E_0^2 + M_N^2}{2[k_A(1 - \hat{k}_A \cdot \hat{k}_B) - E_0]} \tag{A.22}$$

$$p_\pi = \sqrt{E_\pi^2 + E_0^2 - 2E_\pi E_0 - M_N^2} \tag{A.23}$$

This allows one to replace the PbGl energy deposition with the input gamma ray energy. The use of the above formulas will require the use of the above mentioned boost formulas several times with β given by:

$$\beta = \frac{k}{k + M_N} \tag{A.24}$$

Appendix B

The Pion Program Files

B.1 Fortran Files

cerenkovsmear.f	This is where the PbGl energy deposition is smeared.
cleardep.f	Zeroes the energy deposition arrays.
csi.f	Creates the geometry for the CsI arrays.
csismear.f	Smears the energy deposition for the CsI.
fillarraydep.f	Fills the arraydep array with the sum of the energy depositions for the respective arrays.
finddep.f	Finds the energy deposition for a given cell.
gaussian.f	Utility function for finding a Gaussian value given a mean and a RMS.
gukine.f	Called by GEANT to setup the kinematics.
guout.f	Called by GEANT after all particles in an event have stopped.
guplsh.f	File created when trying to the Čerenkov radiation to work.
gustep.f	Called by GEANT on a step by step basis. Used to fill energy deposition arrays.
gutrev.f	Called by GEANT.
gxint.f	Sets up the GEANT interface.
gxphys.f	Controls by command line which physical processes are set.
pbgl.f	Creates the geometry for the PbGl arrays.
pirecon.f	Called from guout.f to reconstruct the pions mass.
uangle.f	Given a cell index it calculates the angles to that cell.
uangleinit.f	Same as uangle.f, but it works through all the cells and fills arrays with the values.
ugeom.f	Called by GEANT to setup the geometry.
uginit.f	Called by GEANT to setup user defined information.
uglast.f	Called by GEANT on exit from program.
uhinit.f	Sets up the histograms.
umax.f	Finds which cell has the maximum energy deposition.
umax2.f	Finds which cell has the maximum energy deposition if one cell is ignored.
upion2.f	Sets up the pion and decay photon kinematics.
xyzrtp.f	Utility function to convert rectangular coordinates to spherical coordinates.

B.2 Kumac Scripts

angdist.kumac	Used in the angle test scripts to determine the mean square values.
angledist.kumac	Same as above.
inhistos.kumac	Used to load the histogram output from the angle tests.
inputhisto.kumac	Allows for the input of a specified histogram from a specified file.
makeps.kumac	Creates a postscript file of a given histogram.
mean.kumac	Calculates the mean of a given histogram.
outhistos.kumac	Used by the angle test scripts to output their histograms.
phieffic.kumac	Used to find the detection efficiency by angle of a given phi angle test.
phitest.kumac	Tests the arrays at various angles of phi.
thetaeffic.kumac	Used to find the detection efficiency by angle of a given theta angle test.
thetatest.kumac	Tests the arrays at various angles of theta.
viewbank.kumac	Sets up the program for running.
writehisto.kumac	Outputs the data from a given histogram to a given file.