Rafael David Pena

Undergraduate Research Spring 2008

This paper documents the status of our progress in the high energy physics lab. We begin with our initial steps in developing a computing cluster and continue from our current state. Overall, our progress in developing a computing cluster is has been good. With loaned machines from the University of Florida, we managed to put together a 40 CPU cluster and in the spring of 2008 we finally connected to the Open Science Grid (OSG). Using our cluster we have also developed simulations for muon tomography using the Geant4 and Cosmic RaY (CRY) packages.

In the fall of 2005 when I began working in the high energy physics lab, we started with five computers. We attempted to configure them to work properly using Rocks downloaded from rockscluster.org. Rocks is a flavor of linux that is built on CentOS. It allows users to build a cluster of spare machines with different architectures and configurations. That fall we struggled with configuration file after configuration file as the seamless integration of Rocks was not as seamless as we had hoped.

At the end of 2005 we had very little to show for our efforts. Several times that fall we managed to configure and install Rocks at least half a dozen times. Our configurations were always flawed and it mainly had something to do with our hardware. When it was not a unsupported network card, it was a faulty hard drive.

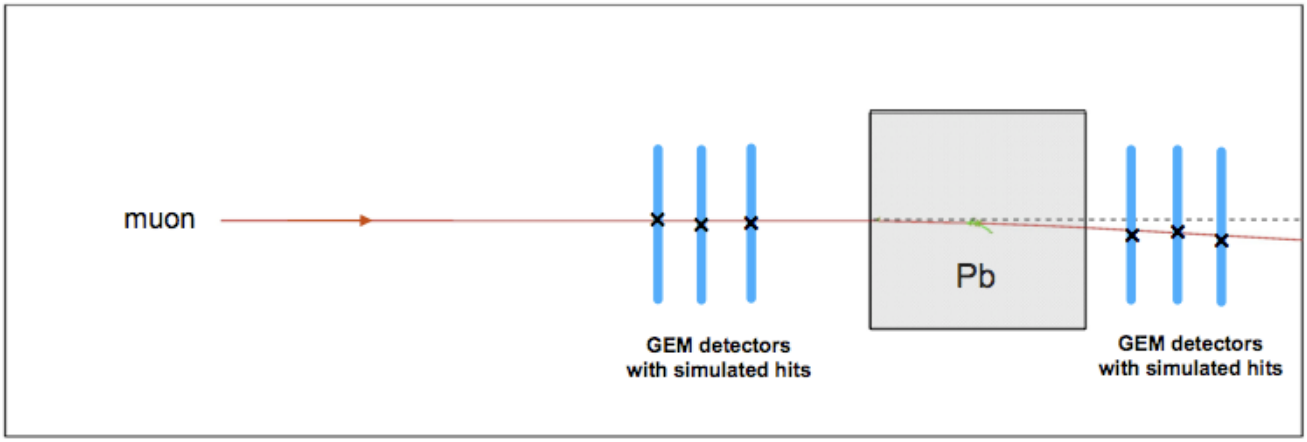
In February 2006 we managed to get a working installation of rocks with two nodes. However, in March 2006 our cluster was hacked and with a new set of installation hardware problems that did not allow us to reconfigure Rocks. After several attempts at this we decided to attempt OSCAR. OSCAR is an Open Source Cluster Application Resource which allows users to install the software on any operating system. We installed Scientific Linux on the computers with no problems and began the process of installing and configuring OSCAR. Although, the installation of the operating system was successful with little problems. OSCAR was not our preferred method for a cluster and it was very difficult to configure.

Before the fall 2006 semester started, Jennifer Helsby and I went to UF and met with Dr. Jorge Rodriguez where we spent two days learning how to manage and maintain a rocks cluster. By the end of the semester, with Dr. Rodriguez's help, we were able to get a 40 node cluster installed and working with condor. We bought another switch and mounted all the nodes into the rack.

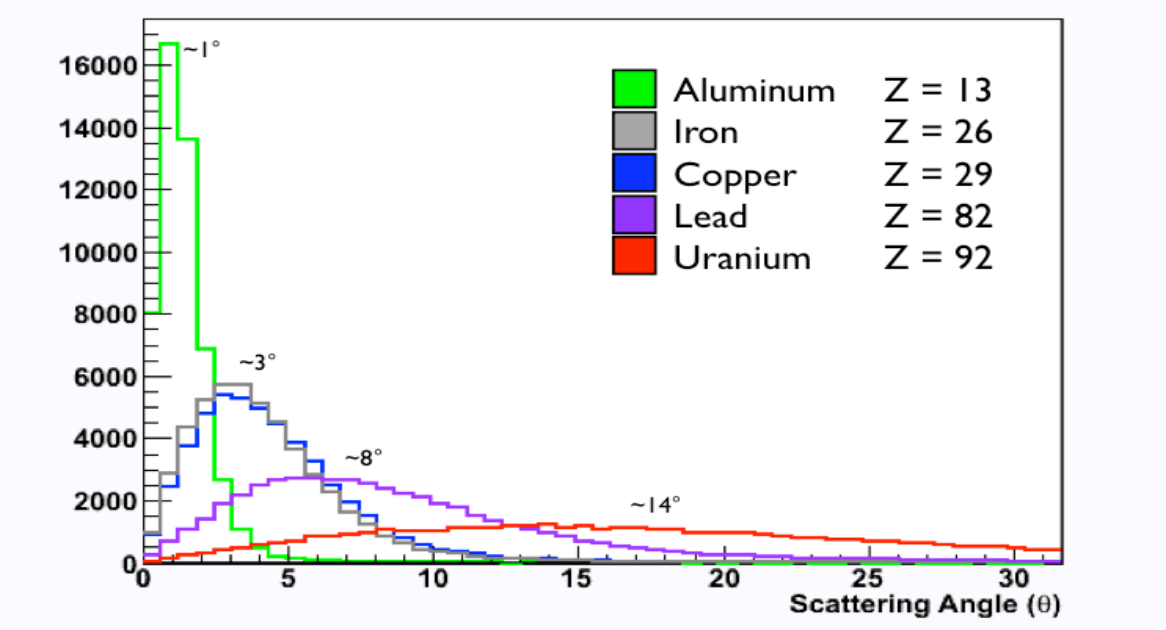
At the end of the semester we began the installation of OSG (Open Science Grid). However, while making a backup of the front end the hard drive crashed and the backup was unsuccessful. In the spring of 2007 we attempted to reinstall Rocks and condor. However, errors while creating the kickstart files for the nodes plagued us and restoring

the cluster to its former glory took until mid February. We installed the two Network Attached Storage (NAS) devices and immediately backed up the frontend onto nas-0-1.

With our cluster running in march 2007 we began installing OSG with the help of Patrick Ford and Richie Hoch. After numerous failed attempts and cryptic errors we spent the remainder of the semester attempting to connect to OSG. In May, I began the



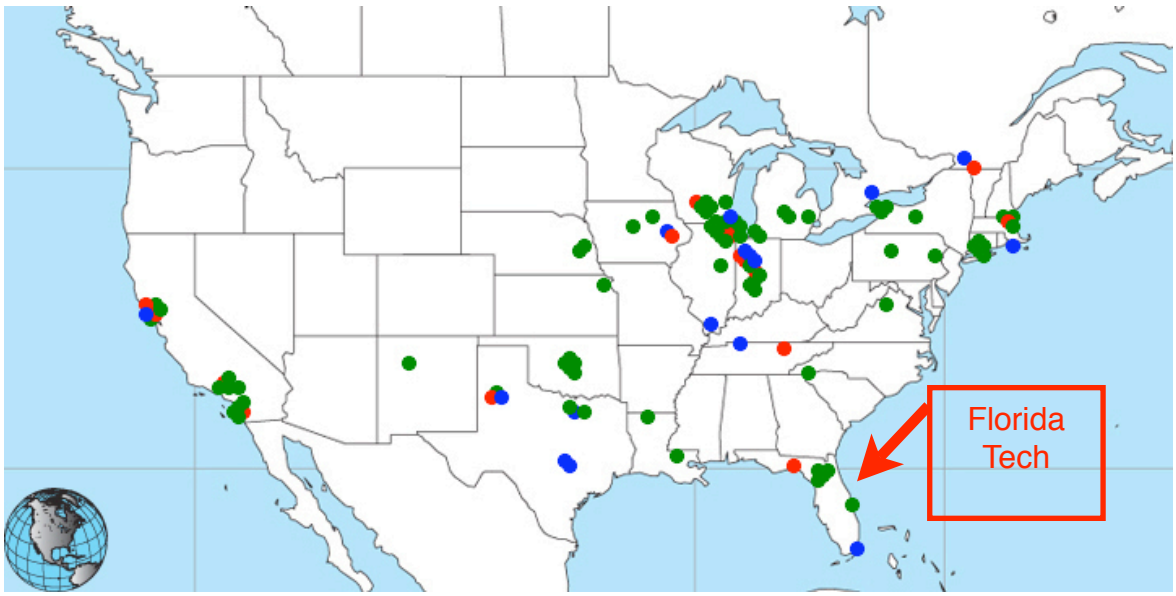
installation of Geant4 onto the frontend. After doing a week of research on the prerequisites for Geant4 I began by installing a basic Geant4 instance which consisted of only CLHEP. With the source code for both of these packages I managed to create RPM packages to install on each of the nodes. In the end we did not use these however they are available on our website. By the end of the summer I managed to create a simple muonApp that simulates the mono-energetic muon scenario below.



Using this scenario, we found the incident direction of the muon and the exiting direction with these two vectors we calculated the scattering angles of the muon going into and out of the material.

In the fall of 2007 we continued making simple simulations and creating our incident muons using Cosmic RaY (CRY). At the end of 2007 we were joined by Dr. Kondo Gnanvo, who has produced more elaborate simulations.

In the beginning of this semester I was tasked to ensure that we were on OSG and our new equipment was configured correctly. Patrick Ford, Richie Hoch, Kondo Gnanvo went to the FIU (Florida International University) Grid School in Miami to learn how to use and install OSG. Going to this course allowed Patrick and I to completely install OSG within two weeks.



After working on installing OSG I began working on updating Geant4 code to correct some of the errors we were having at run time including a CRY error which crashed the applications. The error can be seen below.

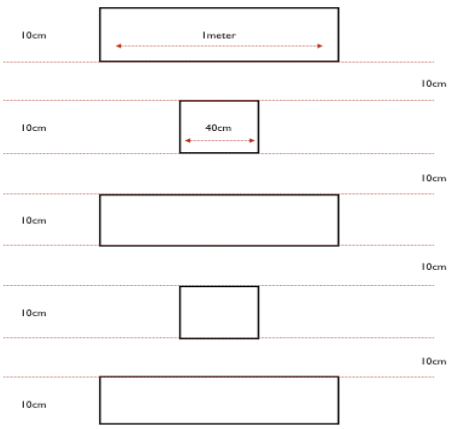
```
Start Run processing.
>>> Event 0
CRY::CRYPdf: Code has failed somehow (impossible...)
CRY::CRYPdf: Name muonChargeDist 40 nan
muonAppCoverageLateralGEM: CRYPdf.cc:282: double
CRYPdf::draw(CRYUtils*, int): Assertion `0' failed.
Aborted
```

The CRY code produced this error when an “Impossible” action occurred. Inside the source file called CRYPdf.cc there is a function which produces the above error. We managed to identify the cause and added a statement to correct for it. Below is the excerpt of the code I fixed.

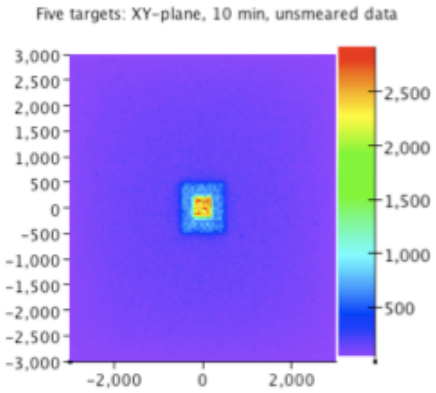
```
  for (int i=(i1-1)*divit; i<  std::min(1.0+i1*divit,1.0*cdfSize); i++ ) {
//******************** FIX IMPLEMENTED BY FLORIDA TECH ***************************
    if (isnan((*_cdfs)[bin][i])) {
     (*_cdfs)[bin][i] = rand+1;
     }
//********************************************************************************
    if ( (*_cdfs)[bin][i] > rand ) {
      if ( _type == CRYPdf::DISCRETE )
     return _min + i*( _max-_min)/(std::max(1.0,double(cdfSize)-1));
      if ( _type == CRYPdf::LINEAR )
     return _min + (i+utils->randomFlat())*( _max-_min)/(cdfSize);
      if ( _type == CRYPdf::LOG )
     return pow(10.,_min + (i+utils->randomFlat())*( _max-_min)/(cdfSize));
      std::cerr << "CRY::CRYPdf: Unknown pdf type? (impossible...)\n";
      assert(0);
    }
  }
std::cerr << "CRY::CRYPdf: Code has failed somehow (impossible...)\n";
  std::cerr << "CRY::CRYPdf: Name "<< name() << " " << bin << " "
         <<   (*_cdfs)[bin][cdfSize-1] << std::endl;
```

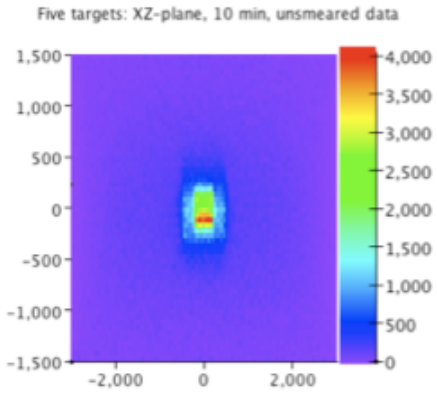
With this if-statement the error is avoided by replacing the value “`(*_cdfs)[bin][i]`” to the value of `rand` plus one.

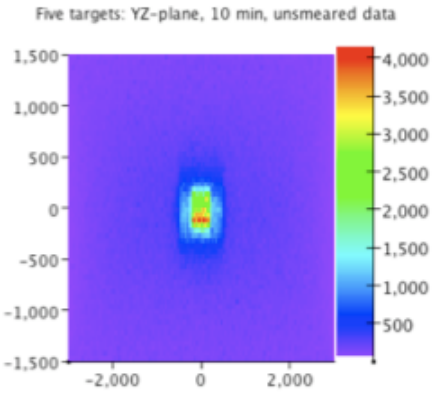
The next thing I completed during this semester dealt with the muon tomography simulations. When we ran our analysis we would run a new simulation for every exposure time. So to run an identical simulation for a ten minute, four minute and one minute exposures each one would have a new configuration file that produced the desired output. With the changes I made to the muonApp, we can now run a ten minute exposure simulation which in turn produces the raw data for the four and one minute simulation.



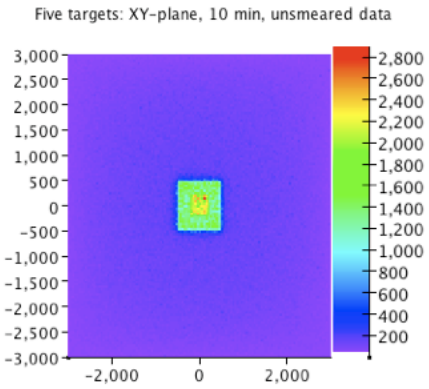
I have also ran several vertical simulations as seen below:
Each scenario includes two materials either Aluminum and Uranium or Iron and Uranium where the wider material is either Iron or Aluminum. Below are some of our

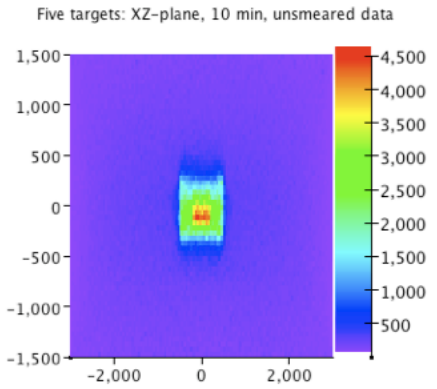


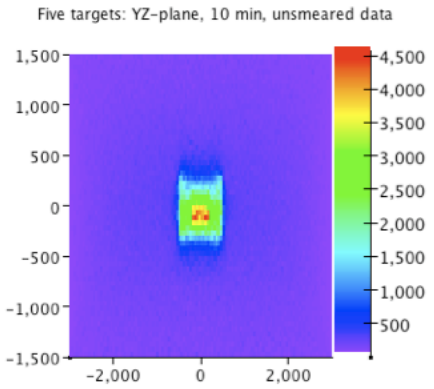




results,







This is a simulation is of aluminum in top and side views from left to right.
This is a simulation is of Iron in top and side views from left to right.

My next task, which is currently underway, is to ensure that our new hardware is working in the coming weeks before the summer is over. This should have good enough documentation that anyone can configure and run from scratch. I will also be making changes to the muonApp to accept the creation of multiple detector materials within the configuration files.