Simulating the Coordinate Detector in the Context of Jefferson Lab's Experimental Hall A

Angelo Rosso

Christopher Newport University
April 2023

Abstract

The objective of this project was to create an event generator for the simulation of the Coordinate Detector (CDet) that produces events that are consistent with experimental events from the GEp experiment in Hall A of Jefferson Laboratory (JLab). Using existing simulations of the GEp experiment provides realistic events, but is computationally taxing. This work found that using a theoretical model of elastic electron scattering can produce events much faster with approximately the same positions and momenta as the in-depth simulations.

1 INTRODUCTION

Experimental research in nuclear particle physics faces two major challenges to its progress. Particle accelerators use a large amount of energy and resources, and they take time to collect enough data to perform meaningful analysis. Somewhat surprisingly, theory is not one of these reasons; there is actually a surplus of theories that can not be tested because of the time and money required to perform accelerator experiments. Because of this, physicists have to go through a rigorous proposal process just to get an experiment approved to run on a particle accelerator. These proposals are expected to demonstrate that the experiment makes sense, is expected to collect meaningful data, and that the data can be analyzed to produce meaningful physics. To accomplish this, it has become increasingly common to simulate physics experiments using software before proposing the experiment be run on actual equipment.

My project deals with the simulation of experiments at the Thomas Jefferson National Accelerator Facility. Specifically, simulating the coordinate detector (CDet) in the Super Big Bite Spectrometer (SBS) program in experimental Hall A. Hall A specializes in the measurement of form-factors of nucleons. Form-factors are mathematical functions that describe the internal structure of particles. The coordinate detector is just one of the numerous detectors in the SBS detector package. It can provide supplemental vertical particle tracking as well as be used as a charged particle veto. A simulation for one bar of this detector was developed by Dr. Edward Brash (2023) using the GEANT4 physics simulation software. Recently, Lydia Lorenti completed her Master's thesis on the calibration and analysis of the performance of CDet. (Lorenti, 2019) Lorenti's project ensured that the simulation data agreed with real data from CDet tests with cosmic ray muons.

My project aims to build on this simulation of CDet. As the actual detector gets closer to being installed in Hall A to be used in experiments, it becomes more useful to have a realistic simulation of it in the context of the experimental hall. While previous efforts were focused on simulating one part of CDet in its test setup, my project augments the CDet simulation to be able to simulate the entire coordinate detector in an experimental context. This entails modeling the correct geometry of the entire detector in GEANT4, acquiring events that are consistent with events from Hall A experiments, and then making those events compatible with the input format of the CDet simulation.

To acquire realistic events, preexisting simulations of Hall A experiments are used. My project focuses on simulating the GEp experiment which has been simulated in the G4SBS simulation library. By enabling the thorough tracking functionality in G4SBS, the position and velocity of the particles that hit CDet one time step before they pass through the face of the detector can be extracted. These events are then converted into the form that the CDet simulation accepts and used as the input file. While this method works and produces realistic results, the G4SBS simulation of Hall A is very slow. My project also investigates simulating a simplified model of elastic electron scattering to determine if it can be used to quickly generate events instead of running the complex G4SBS simulation.

2 THEORY

As mentioned above the goal of this project is to create a way to generate events for the coordinate detector simulation that are consistent with GEp experimental events. The coordinate detector is a scintillation-based detector with two main purposes: to provide additional high-resolution, vertical position information or function as a charged particle veto. The purpose the detector will serve is dependent on if a proton or neutron experiment is being performed. In proton-target experiments like GEp, the coordinate detector will be used in the electron arm to provide tracking information on the scattered electron. In neutron-target experiments, CDet will be used in the hadron arm to determine if the scattered particle has an electric charge or not. The coordinate detector can do this because only charged particles interact with the scintillation material. This information can be used in experimental triggers to get rid of unwanted data. The actual geometry of the CDet is two layers of three modules each. Each module is cut in half vertically with a mirror. Each side is composed of 14 scintillating bars with 14 scintillating paddles in each. Because each paddle is so thin, CDet is able to provide high-resolution vertical position data. This geometry can be seen in the following figures.

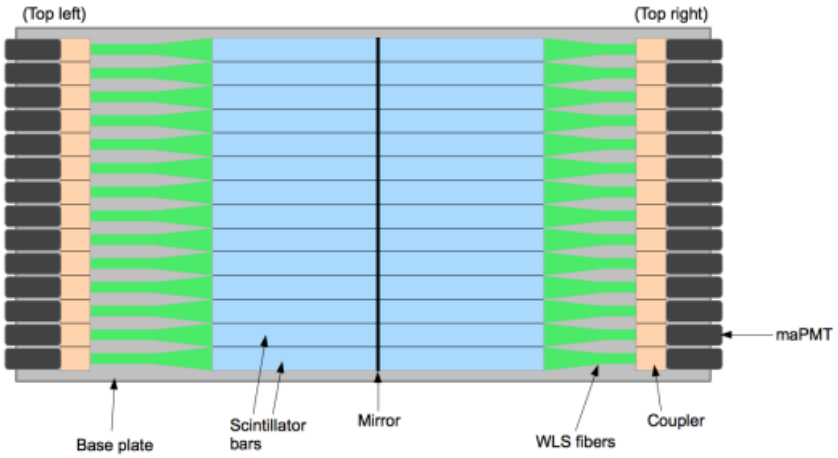

Figure 2.1. One module of the coordinate detector. Image from Lorenti (2019)

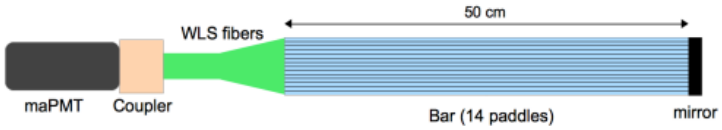

Figure 2.2. One bar of the coordinate detector. Image from Lorenti (2019)

The most accurate way to generate realistic events is by using the very detailed simulation library G4SBS. A simulation of the GEp experiment can be created and then the events which are incident on CDet can be extracted at a known distance back from the face of the detector. The positions and momenta of those particles can be saved to a root file which can be processed to be in the format that the CDet simulation accepts. To do this, a coordinate transformation must be done between the global and local coordinate frames in the hall.

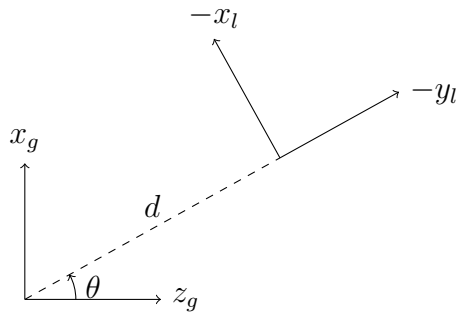

Figure 2.3. Global coordinate frame and CDet coordinate frame

In the image above, the global Hall A coordinate frame is labeled with a subscript g, and the local CDet coordinate frame is labeled with subscript l. In the GEp experiment, $d = 4m$ and $\theta = 29°$. Using the picture above and trigonometry, we can derive that:

$$x_l = z_g \sin\theta_0 - x_g \cos\theta_0 \tag{2.1}$$

$$y_l = -\, z_g \cos\theta_0 - x_g \sin\theta_0 \tag{2.2}$$

$$z_l = -\, y_g \tag{2.3}$$

Using these transformations, the output of the G4SBS simulation of the GEp experiment can be transformed and passed to the CDet simulation. The problem with this approach is that it requires the GEp simulation to be run which is very computationally intensive. For testing just CDet, it would be nice to have a quick way to generate events without running the entire GEp simulation.

My project investigates a very simplified model of the physics interactions to determine if it can produce accurate events. The main theory that is used in the simplified event generation model is elastic electron scattering. Most experiments at Jefferson Lab can be described as scattering experiments where electrons from the accelerator are accelerated to very high energies and then shot at a target with the aim of hitting either a proton or neutron depending on the experiment. In this project, I will focus on scattering off of a proton because I am simulating the GEp experiment which has a proton target. Assuming that the scattering is elastic means that we are making the simplification that kinetic energy is conserved in the collision. In actual high-energy scattering experiments, there is potential for the electron to actually ”break” the nucleon apart in which case the kinetic energy would not be conserved. Using the elastic simplification makes many calculations very simple because it implies that four-momentum is conserved in these collisions. Four-momentum is a four-dimensional vector that combines a particle’s energy

with its three-dimensional momentum as $P = (E, p)$ where P is the four-momentum, E is the energy and p is the three momentum. The components of the four-momentum vector are the energy of the particle and the momentum in each of the three spatial dimensions. The conservation of four-momentum can be expressed by the equation:

$$P_e + P_p = P_{e'} + P_{p'} \tag{2.4}$$

In this equation, P_e and P_p are the four-momenta of the incoming electron and stationary proton target, while $P_{e'}$ and $P_{p'}$ are the four-momenta of the scattered electron and scattered proton. This equation allows the calculation of momentum transfer as a function of only the scattering angle and incoming electron energy. Two important properties of four-vectors are how they multiply and how they square. Two four-vectors multiplied is defined as $P_1 \cdot P_2 = (E_1, p_1) \cdot (E_2, p_2) = E_1 E_2 - p_1 \cdot p_2$. The square of a four-vector is defined as $P^2 = (E, p)^2 = E^2 - p^2 = m^2$. Another simplifying assumption that will be made in the following calculations is that the electron mass is small so $m_e^2 \approx 0$. Now, we will introduce the concept of four-momentum transfer q as:

$$\begin{aligned}
q =& P_e - P_{e'} && (2.5)\\
q^2 =& (P_e - P_{e'})^2 && (2.6)\\
q^2 =& P_e^2 - P_{e'}^2 - 2P_e P_{e'} && (2.7)\\
q^2 =& m_e^2 - m_{e'}^2 - 2(E_e E_{e'} - p_e \cdot p_{e'}) && (2.8)\\
q^2 =& -2(E_e E_{e'} - p_e p_{e'} \cos\theta) && (2.9)\\
q^2 =& -2E_e E_{e'}(1 - cos\theta) && (2.10)\\
q^2 =& -4E_e E_{e'} \sin^2\frac{\theta}{2} && (2.11)\\
& && (2.12)
\end{aligned}$$

Conventionally, a new variable called Q^2 is defined as follows:

$$Q^2 = -q^2 \tag{2.13}$$

This allows us to write

$$Q^2 = 4E_e E_{e'} \sin^2\frac{\theta}{2} \tag{2.14}$$

This equation shows that the four-momentum transfer can be calculated with only the energy of the incoming electron (beam energy), the energy of the scattered electron, and the scattering angle.

While this equation is very important, in order to simulate events from this model, we need a way to calculate the energy of the scattered electron given the beam energy (incoming electron energy) and the scattering angle. To do this, we will start from Equation (2.1), then solve for $P_{p'}$ and square the whole equation, which gives:

$$P_{p'} = P_e - P_{e'} + P_p \tag{2.15}$$
$$P_{p'}^2 = (P_e - P_{e'} + P_p)^2 \tag{2.16}$$
$$P_{p'}^2 = P_e^2 + P_{e'}^2 + P_p^2 - 2P_eP_{e'} - 2P_{e'}P_p + 2P_eP_p \tag{2.17}$$
$$m_{p'}^2 = m_e^2 + m_{e'}^2 + m_p^2 - 2P_eP_{e'} - 2P_{e'}P_p + 2P_eP_p \tag{2.18}$$

Using the fact that $P^2 = E^2 - p^2$ and $m_e^2 \approx 0$, we can substitute and simplify as:

$$m_{p'}^2 = m_e^2 + m_{e'}^2 + m_p^2 - 2(E_eE_{e'} - p_e \cdot p_{e'}) - 2(E_{e'}E_p - p_{e'} \cdot p_p) + 2(E_eE_p - p_e \cdot p_p) \tag{2.19}$$
$$0 = -(E_eE_{e'} - p_e \cdot p_{e'}) - (E_{e'}E_p - p_{e'} \cdot p_p) + (E_eE_p - p_e \cdot p_p) \tag{2.20}$$

Now we use the fact that the target proton is stationary meaning it has no momentum so its energy is just its rest mass energy thus $p_p = 0$ and $E_p = m_p$.

$$0 = -(E_eE_{e'} - p_e \cdot p_{e'}) - (E_{e'}E_p - p_{e'} \cdot p_p) + (E_eE_p - p_e \cdot p_p) \tag{2.21}$$
$$0 = -(E_eE_{e'} - p_e \cdot p_{e'}) - (E_{e'}m_p) + (E_em_p) \tag{2.22}$$
$$0 = -E_eE_{e'} + p_e \cdot p_{e'} + m_p(E_e - E_{e'}) \tag{2.23}$$

The last simplification we need to make is that because we are using high-energy electrons, the mass is very small compared to the momentum so $E_e \approx p_e$ and $E_{e'} \approx p_{e'}$. We will also simplify the dot product using the definition $a \cdot b = ab\cos\theta$ where θ is the angle between a and b.

$$0 = -E_eE_{e'} + p_e \cdot p_{e'} + m_p(E_e - E_{e'}) \tag{2.24}$$
$$0 = -E_eE_{e'} + E_eE_{e'}\cos\theta + m_p(E_e - E_{e'}) \tag{2.25}$$
$$0 = -E_eE_{e'}(1 - \cos\theta) + m_pE_e - m_pE_{e'} \tag{2.26}$$
$$m_pE_e = E_eE_{e'}(1 - \cos\theta) + m_pE_{e'} \tag{2.27}$$
$$m_pE_e = E_{e'}(E_e(1 - \cos\theta) + m_p) \tag{2.28}$$
$$E_{e'} = \frac{m_pE_e}{E_e(1 - \cos\theta) + m_p} \tag{2.29}$$

With a known beam energy, the only unknown in this equation is the scattering angle. This can be calculated using some trigonometry by using the position of the detector and where the particle hit on the detector.

The scattering angle is defined as the angle that the outgoing path makes with the path of the incoming particle if it had not collided with anything. The only complication in this calculation is that there are differences between the coordinate frames in which the particles collide and are detected. The electrons collide with the target in the global Hall A coordinate frame which

defines the positive z direction to point along the direction of the electron beam and the positive y direction vertically up. The particles are detected in the local CDet frame which defines the negative y direction to go into the face of the detector and the positive z direction to be vertically down as seen in Figure 2.3. Using this information, we can calculate the scattering angle from the position (x, y, z) in the detector frame, the distance the detector is away from the target (d), and the angle the detector is from the beam direction (θ_0) as follows:

$$\theta = \arccos\left(\frac{-x\sin(\theta_0) + z\cos(\theta_0)}{\sqrt{x^2+y^2+z^2}}\right) \tag{2.30}$$

Using these two equations, we can calculate the scattering angle and energy of a scattered electron given where it was detected on the detector. In the GEp simulation, it was observed that CDet was hit uniformly all over. Because of this, the event generator can generate random hits all over the acceptance of the detector and then use these equations to calculate the momentum components.

In the data analysis, the data created from this simplified model will be compared to the data generated by the G4SBS simulation to determine if the model can produce realistic events. In particular, the position and momentum of the hits from each method will be compared. In general, I expect the data from the two generators to be almost the same because the simplified model provides an accurate representation of electron scattering in terms of where the electrons go and how much momentum they have. What this model does not give is accurate information about what happens to the proton and the polarity of the scattered particles. While these details are very important to the experiment, they may not be as important to the simulation of the coordinate detector.

3 METHODS

The creation of this project generally entailed making a G4sbs simulation of the GEp experiment, converting the information from this simulation into a form compatible with the CDet simulation, creating the simplified generator using the elastic scattering model, and then comparing the events generated by the two different methods. There were, however, substantial complications with the completion of these steps.

The first task of this project was to create the GEp simulation using the G4SBS simulation software. This would be used to generate events to pass into the CDet simulation as shown below.

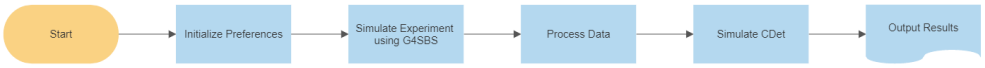

Figure 3.1. General Program Architecture

JeffersonLab (2021) is a simulation library created using the Geant4 simulation software. Geant4 was created by CERN, which is the European Organization for Nuclear Research. They created Geant4 to provide an easy way to simulate very complicated physical interactions. To go along with Geant4, they created a data analysis framework called ROOT. ROOT provides a structured way to store data from Geant4 simulations in a tree structure and many built-in analysis tools. Luckily, because simulations are commonplace in nuclear physics, the GEp experiment has been extensively simulated in G4SBS already. I was able to take an existing simulation and modify it to my needs by choosing the kinematic setting I wanted and enabling additional tracking functionality. The additional tracking branches I enabled gave me access to information such as the position and momentum of particles one time step before they contact the coordinate detector. At this point, I noticed a potential issue that arose in the difference between what the CDet simulation was simulating and what the G4SBS simulation was simulating. The CDet simulation was only simulating one bar of one module of the coordinate detector. This is a problem because if a ran a G4SBS simulation in which $100,000$ particles hit CDet, only around 600 will have hit this one out of the total 168 bars in the full detector. When I brought up this issue to Dr. Brash, he modified the CDet simulation to include the entire detector geometry.

Originally, I tried to use Python and the UPROOT library to open the output file from the G4SBS simulation, extract the information I need, and create a new ROOT file to pass to the CDet simulation. I was able to do everything that I tried to do, but when I passed the ROOT file I created to the CDet simulation, there were errors with the data types. While I ensured the types of each value were the same, the CDet simulation was expecting the data to be in C++ vectors. This makes sense because Geant4 and ROOT are written in C++ and use it as their scripting language. Because of this, I had to convert my Python code to C++ code. This fixed the type issue and allowed the

CDet simulation to run with my ROOT file as the input. Unfortunately, when the CDet simulation completed, none of the particles hit the detector. This was clearly an error because I had taken only the particles that hit CDet in the G4SBS simulation and passed those to the CDet simulation. I realized that the error occurred in the process of converting between the coordinate frames of each simulation.

As described in Figure 2.3, the tracking data from the G4SBS simulation is stored in the global Hall A coordinate frame while the CDet simulation expects data in its local coordinate frame. I knew these two simulations used different coordinate frames, but I thought at first that the tracking data would be in the local spectrometer coordinate frame. This is a coordinate frame used in G4SBS for all of the detectors in a certain arm of the experiment which is defined as having the positive z axis in the direction of travel of incoming particles (perpendicular to the face of the detectors). If this were the case, converting between G4SBS output data and CDet input data would be simple because the directions of the axes are the same, just with different labels. However, because the G4SBS tracking data uses the global coordinate frame, trigonometry was needed. Once I figured out which coordinate frames all of my data was in, I was able to program the transformation and the particles were hitting the detector in the CDet simulation as expected.

After performing some analysis to confirm that everything was working as expected, I tried to come up with a way to generate realistic events without running the entire G4SBS simulation. The method that was decided on is to use the concept of elastic electron scattering to program a very simple event generator that would produce data with the same position and momentum distributions as the data from the G4SBS simulation. The necessary equations were derived as explained in the theory section from basic four-momentum conservation and trigonometry. This generator was first prototyped in Python and then implemented in the C++ event generator. This generator was added as an option to be chosen by the user as shown below.

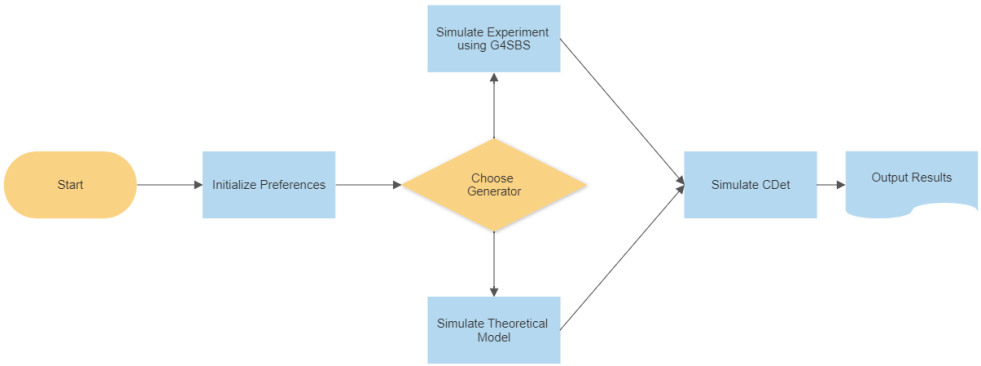

Figure 3.2. Program Architecture with Two Generators

The way it works is a random position on the detector is generated as a hit. This is allowed because, from the G4SBS data, it was observed that the entire face of the detector was hit uniformly. From the position, I can calculate the scattering angle and from the scattering angle, I can calculate what

the momentum of that electron must have been to hit the detector in that location.

After I had the two methods of generating events, I needed to determine if the simplified model generated events that are actually consistent with the GEp simulation. To do this, I used a Jupyter Notebook and Python to create graphs of position vs momentum for each axis for each of the two methods and compared them. I also made histograms of position and momentum for each axis for each generation method and compared them. These results are discussed more in the data and results sections.

4 DATA

The same data was collected using the event generator based on the G4SBS simulation of the GEp experiment and the generator based on the simplified electron scattering model. The events from these generators will be compared and analyzed in the Discussion section. For both generators, the data was collected by running a simulation with $40,000$ events. Data is used from the generated input files and simulated output files.

4.1 G4SBS GENERATOR

The data in this section is data produced by the event generator that runs the G4SBS simulation of the GEp experiment to acquire events.

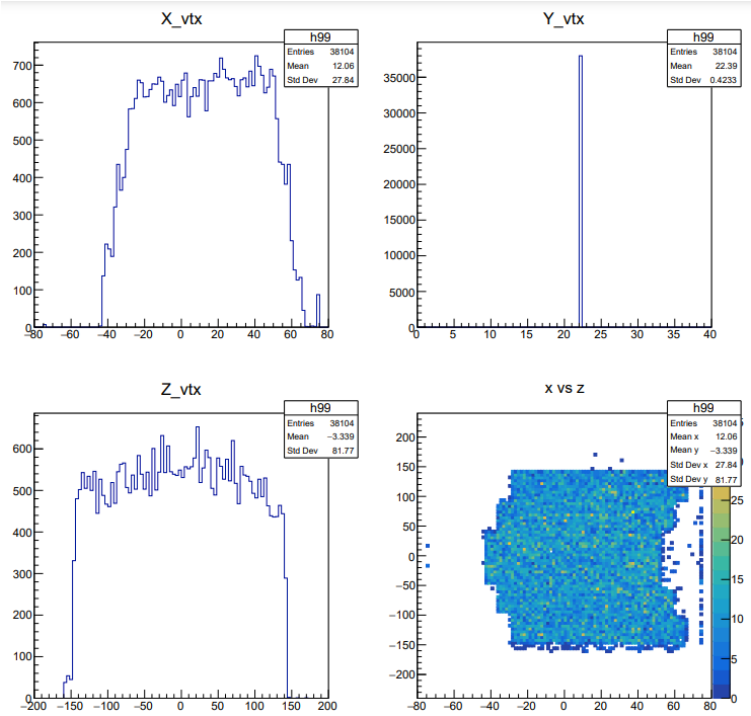

Figure 4.1. Position distributions for G4SBS-generated data.

These position distributions show the positions of the particles one simulation time step before they make contact with the detector. The data is shown in the local detector coordinate frame where the $-y$ direction is into the detector and the x and z directions are horizontal and

vertical to the face of the detector respectively.

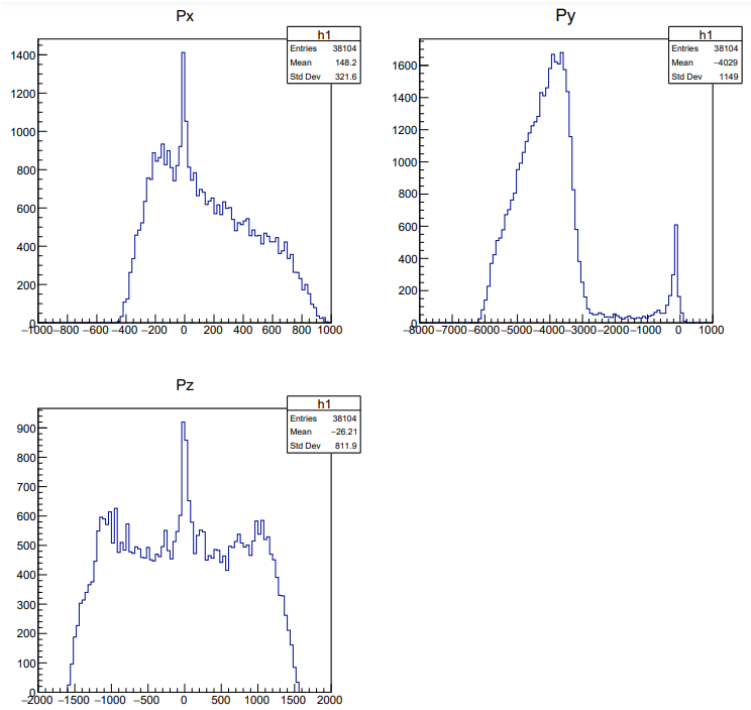

Figure 4.2. Momentum distributions for G4SBS-generated data.

The above momentum distributions each have spikes at values of zero due to low-energy ”noise” particles. The x and z distributions have a mean value relatively close to zero which makes sense because these are the directions on the plane of the detector. The z direction is vertical and has no bias. The x direction is horizontal in the plane of the detector face so as the scattering angle increases, the energy decreases according to Equation (2.29). Ignoring the noise in the y direction, the momenta are large and negative meaning they are going toward the face of the detector.

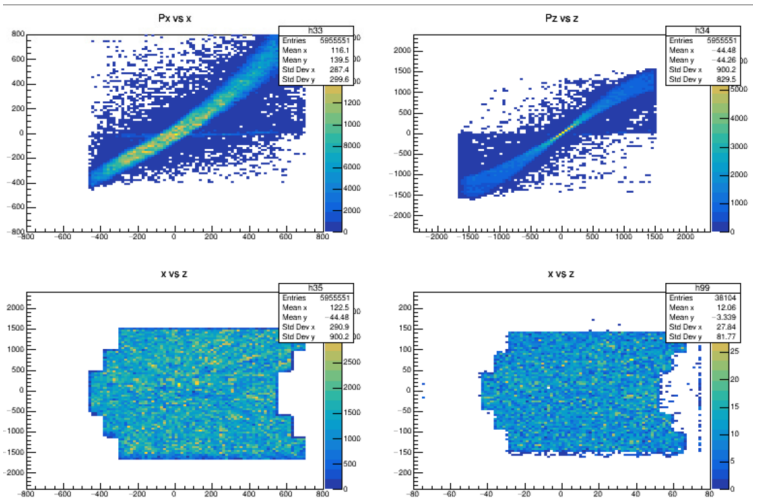

Figure 4.3. Data from G4SBS event generator. Top: momentum vs position plots. Bottom: x vs z position plots for photons and electrons.

The lower two histograms in the above figure both show data for x and z positions, but the left plot is for the simulated photon positions in the detector while the right plot is the generated electron positions before making contact with the detector.

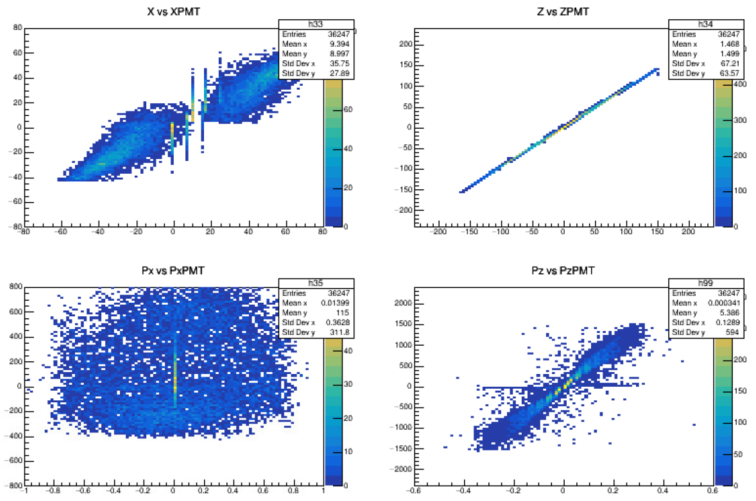

Figure 4.4. Reconstruction vs Reality from G4SBS data

The plots above show the quality of information we can actually get from the output of the detector. The plots compare the calculated positions and momenta of each hit from the output of the PMT's vs the actual positions and momenta from the simulation.

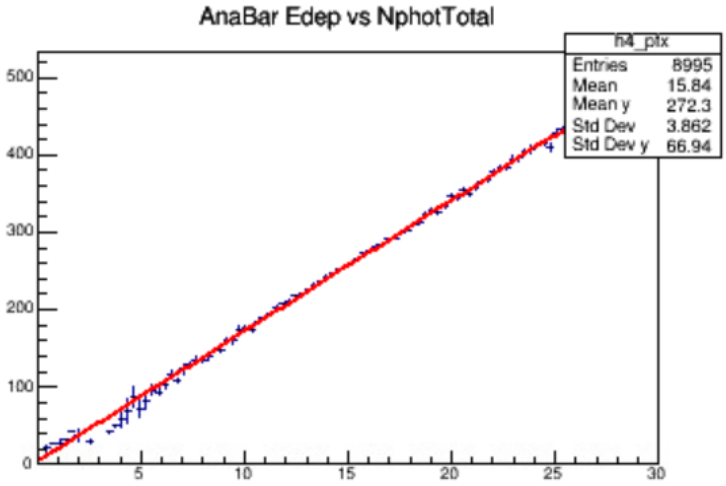

Figure 4.5. Fitted energy deposition vs number of photons graph from G4SBS data.

This plot shows the linear relationship between the energy deposited and the number of photons produced. The legend also shows that the average energy deposited is 15.84 MeV and the average number of photons produced is 272.3.

4.2 SIMPLIFIED MODEL GENERATOR

The data in this section is data produced by the event generator that uses the simplified theoretical model of elastic electron scattering to acquire events. Because these graphs are the same as the ones for the G4SBS generator, only notable differences will be noted.

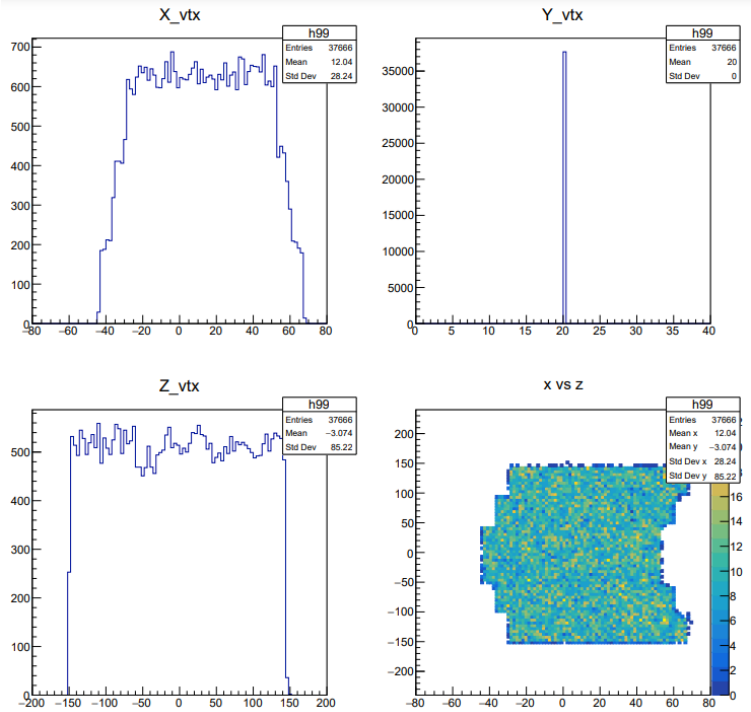

Figure 4.6. Position distributions for theoretical model-generated data.

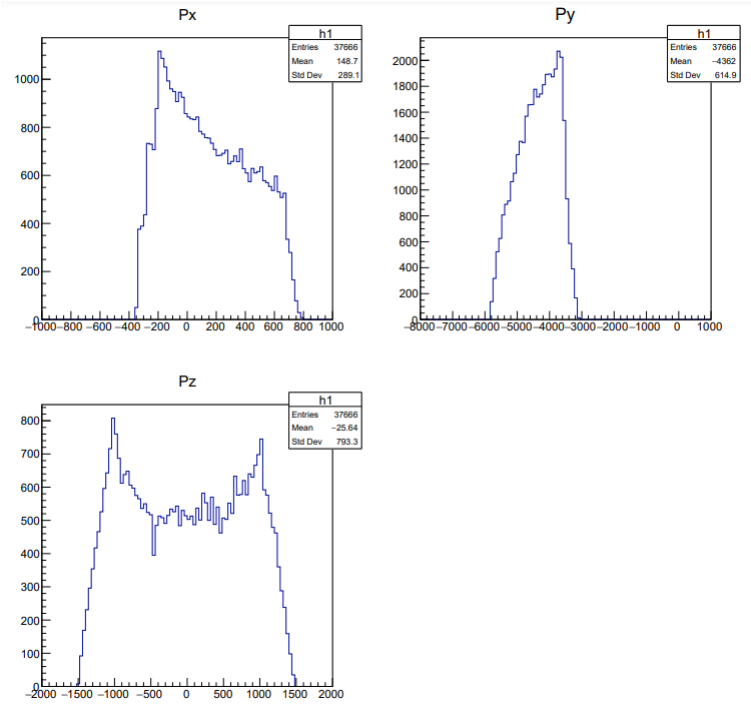

Figure 4.7. Momentum distributions for theoretical model-generated data.

Note that the above plots do not have spikes at zero as the momenta plots from the G4SBS data do. These spikes are caused by low-energy noise particles which are not included in the simplified model.

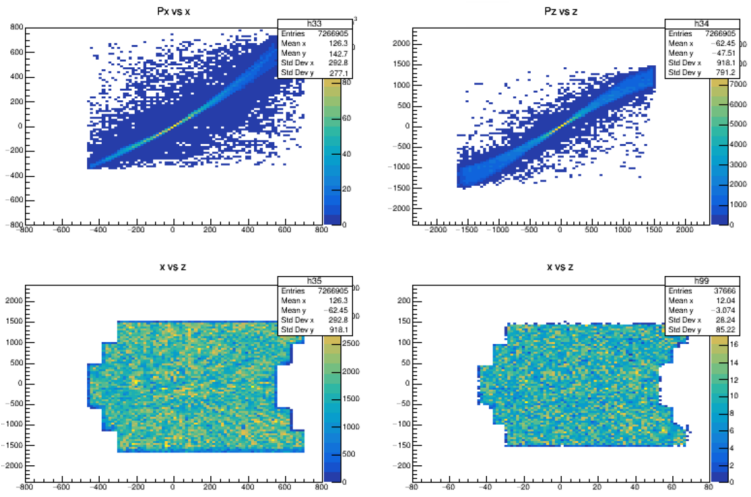

Figure 4.8. Data from theoretical model event generator. Top: momentum vs position plots. Bottom: x vs z position plots for photons and electrons.

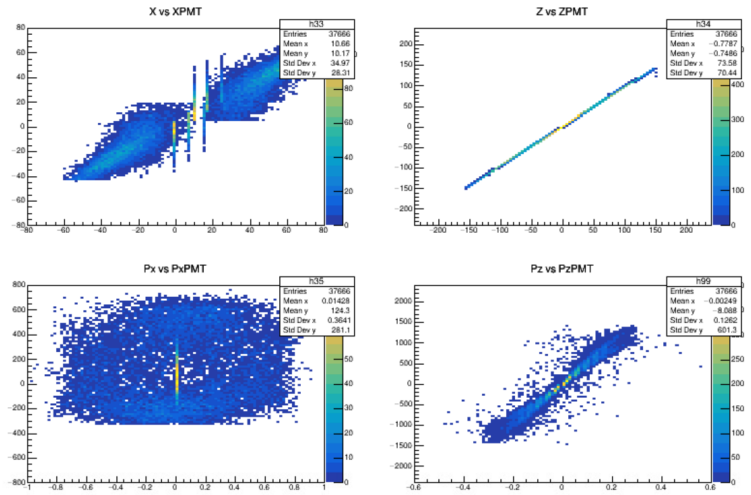

Figure 4.9. Reconstruction vs reality from Model data.

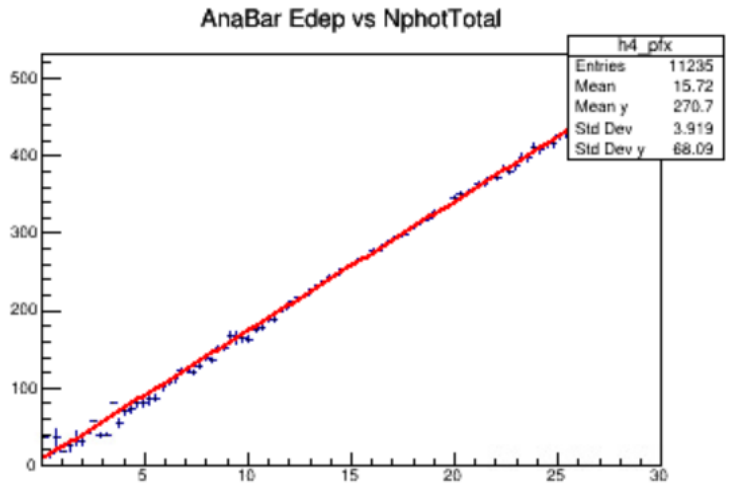

Figure 4.10. Fitted energy deposition vs number of photons graph from model data.

The legend in the above plot shows that the average energy deposited is 15.72 MeV and the average number of photons produced is 270.7.

5 DISCUSSION AND CONCLUSIONS

Overall the project was successful in creating a realistic event generator for the simulation of the coordinate detector. The G4SBS-based generator produces very accurate events by simulating every physics interaction that occurs, however, it is very slow. The simplified model-based generator can produce events very quickly but is only an approximation of the actual events CDet will encounter.

The results clearly show that the generator based on the simplified theoretical model produces events that are consistent with events from the G4SBS-based generator. For most purposes, the simplified generator produces accurate enough events. Cases where this model is not sufficient arise if you need more information than just the position and momentum of the electron. This model does not give information about the scattered hadron or any polarity information.

Figures 4.4 and 4.9 show that the position information that is actually given by the detector is much better for the z direction than for the x direction. This makes sense when considering the geometry of the detector; the scintillating paddles are about 500 millimeters in the x direction while they are about 5 millimeters in the z direction. This disparity is why the tracking functionality of this detector is frequently stated as giving "additional vertical" position information.

This project was successful in creating a realistic event generator for the CDet simulation as well as motivating a complete simulation of the entire geometry. There is, however, additional work that can be done. The current implementation of the event generator supports a way to simulate any G4SBS experiment that CDet is involved in but has only been fully implemented for the GEp experiment. Future work could create simulations of other experiments that CDet is in and then add the correct coordinate transformation for those experimental setups into the data processing script. Additionally, if more experiments are added, future work could look into how effective CDet is at being a charged particle veto in neutron experiments.

REFERENCES

Brash, E. (2023). "Anabarmc." *https://github.com/brash99/AnaBarMC*. [Accessed: 3/20/2023].

JeffersonLab (2021). "Super bigbite simulation." *https://github.com/JeffersonLab/g4sbs*. [Accessed: 9/20/2022].

Lorenti, L. (2019). "Calibration and performance studies of the coordinate detector for the super bigbite spectrometer in jefferson lab's hall a via geant4 simulation and root analysis." M.S. thesis, Christopher Newport University, Christopher Newport University.