Contents

Chapter 10

Monte Carlo

Monte Carlo simulations of photoproduction reactions and the detector response are an integral part of data analysis for HALL D. Monte Carlo data sets an order of magnitude larger than the real data for specific channels must be produced and analyzed within a unified analysis framework. The computer resources needed for this task were discussed in the previous chapter. This chapter describes how the simulation is to be carried out, the specific software components that exist at present, and some preliminary results regarding detector acceptance and resolution.

During the conceptual design phase of HALL D two parallel paths of Monte Carlo development have been followed. The first has been focused on simulating reconstructed events for acceptance and resolution studies, and for tests of partial-wave analysis. On this path the simulation of particle interactions in the detector followed by track/cluster reconstruction is replaced by a model which accounts for the smearing of the final particle momenta according to detector resolution. This so-called *fast* Monte Carlo approach is computationally very efficient and permits the exploration of large regions of detector parameter space during design. In fact, important parts of the design evaluation can only be accomplished by this approach, before a full event reconstruction package is available.

When the event reconstruction package arrives, a different sort of simulation code will be needed. This so-called *physics* simulation relies on a detailed geometrical description of the detector and a library of known particle-material interactions to estimate the detector response to a given event as accurately as possible. From this response it forms a simulated event record that is analyzed by the reconstruction package in a similar way as real events. The physics simulation package should come first in the order of software development because it provides useful test data for debugging the rest of the analysis chain. The

physics simulation is also useful at the design stage for estimating background rates in detector and trigger elements. This is the second path of Monte Carlo development being pursued by HALL D.

These Monte Carlo simulation programs are the first components in what will grow to be a large body of code for the HALL D experiment. It is useful to consider at the outset what pieces of these codes might be of broader use than strictly for simulation. For example, the reconstruction code will need access to the same alignment data as is used by the simulator. Some of the requirements for simulation can be met by incorporating existing software packages from other sources; however their use must be coordinated to avoid conflicts and unwanted dependencies in the future. Software developed at this early stage of the experiment must undergo numerous stages of evolution if it is to be of lasting usefulness. The incorporation of industry standards into the code wherever possible lays the groundwork for a smooth evolution in the foreseeable future. All of these things come together in the formulation of a software *framework* for the experiment.

In the sections which follow are discussed, first, the software framework, followed by a description of the individual components of the simulation package. The following three sections summarize the results from early design studies carried out with the fast Monte Carlo. The final section describes the general method how simulation results are incorporated into a partial-wave analysis.

10.1 Monte Carlo framework

In this context, a framework refers to a set of specified interfaces through which the different software components in a system interact and exchange information, together with a set of common tools that facilitate access to information through these interfaces by application programs . Use of a framework allows builders of individual components to have a relative degree of independence in their implementation choices, knowing what requirements they must satisfy in order to work successfully with the other pieces. Before proceeding to the specifics, it is worthwhile to note two general principles that have been adopted for HALL D code development.

1. All data within the framework must be viewable in a well-formed xml document format that expresses the structure and relationships within the data.

2. All major interfaces should be implemented as web services, in addition to the normal API.

Not specified in this list is any mandated set of languages, operating systems, or disk file formats. While prudence suggests a restricted set of choices for each of these for developing new code, it was decided that the benefits of the freedom to borrow existing programs from a variety of sources outweighs the cost in complexity. Where necessary, legacy code can be wrapped in such a way that it provides its functionality through a protected interface. In any case, software technology is changing too fast at present to allow a final deci-

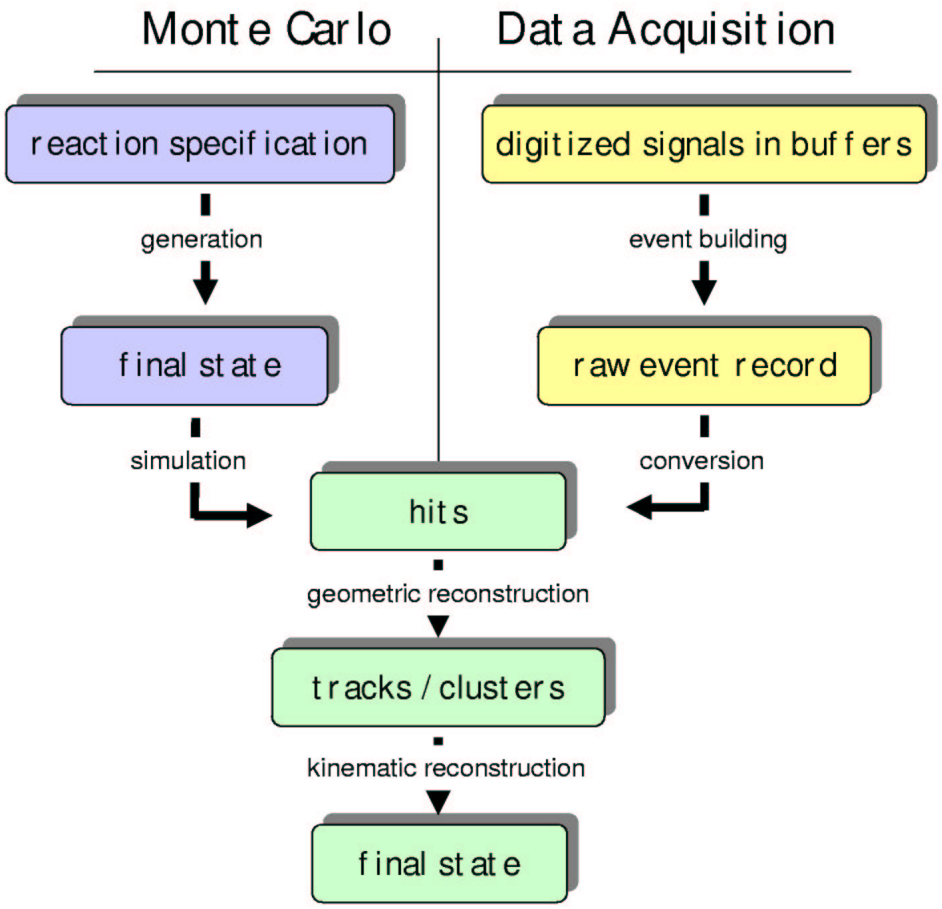

Figure 10.1: Data flow diagram showing the major software components responsible for data processing in HALL D.

The overall data-flow scheme for HALL D is shown in Fig. 10.1. Data flows from the top to the bottom of the figure. On the right-hand side, digitized events come from the detector, are converted to hits by applying corrections derived from the calibration database, and are passed to the reconstruction programs for further processing. On the left-hand column, events start off as lists of particles and their momenta coming from a physics event generator and are converted to hits in the simulator, after which they follow the same path as the real events.

The remainder of this chapter is concerned mainly with what happens to the left of the vertical line in Fig. 10.1. For clarity, we will distinguish between event *generation* (first step) and *simulation* (next step) in the Monte Carlo process. As far as the framework is concerned, the only thing of concern is how programs (or people) can access what is inside the boxes. The details of how data is stored inside the boxes, or what happens inside the processing steps is beyond the scope of the framework. The generator needs to be able to find out what kind and how many events to generate. The simulator needs to be able to get from the generator a sequence of event specifications, and it must be able to provide its event hits to the reconstruction code in a format that it understands. Not shown in the figure but also important are the detector geometry information and the simulation control data which the simulator also needs. Within the HALL D framework, all of these data have the common property that they are viewable in xml. When the software components are fully incorporated into the framework then each of the processing steps will be available on the HALL D grid as a web service.

The formal specification of all of these interfaces is incomplete at present. The most complete specification is that of the detector geometry information, which has been published [1] on the web. It is described in more detail in a later section. A draft specification for the event description has also been published. [2]. Depending on the location along the data-flow pipeline, different pieces of event information are available. However it is decided that access to all event data by application code within the HALL D framework is through a single interface. That interface must provide a mechanism for determining what kinds of information are available in an event and for providing what is available in a standard way to the client program.

This is all quite easy to do by specifying the interface in terms of an xml schema. However doing event input/output through xml libraries is very expensive for large data sets, not only in terms of data volume but also cpu overhead. This is why the framework specifies that all data should be *viewable* in xml, not necessarily *stored* in xml. No restrictions are placed on what data formats are actually used internally by applications, or how events are

stored in disk files. In practice it proved convenient for the purposes of Monte Carlo to create a self-describing event data format that is very close to the underlying xml, called *hddm*. An hddm event stream (or file) begins with an event template in plain-text xml that describes the information that is available for each event, followed by the actual event data. The tags have been suppressed in the event data and the values written in binary format, so that the event record size is roughly equivalent to other binary formats. Framework tools exist which can automatically generate a miniature c or c++ library that contains the calls needed by an application program to access the event data, just by reading the first few lines in an hddm event file. Applications built with one of these libraries automatically verify that the data they require are present in the file before access is attempted. Finally, a single pair of translators called hddm-xml and xml-hddm exist which are capable of converting any hddm data stream to and from xml.

Thus the interface to the data in each of the boxes in Fig 10.1 is expressed in a xml specification that serves as an event template. The specification contains an inheritance mechanism that makes it easy to extend the event definition, so that producers and consumers of event data can decide to exchange additional information through the extended interface without interfering with the operation on the same data by older programs that rely on the base interface. All of this is verified automatically by the framework API library without any need for checks by application code. Writers of application code have the choice of accessing the data through the API (currently provided in c and c++ only) or by reading and parsing the xml. Use of the API is more efficient in that it eliminates the xml parsing step, but the choice of languages is restricted. On the other hand, standard tools are available in all major languages that make it easy to read and write xml. The advantage of this design is that anyone in any language that has the capability of reading ascii text has access to the event data in a standard way.

The hddm scheme is effectively an efficient mechanism for prototyping interfaces to event data. Eventually the information content of an event will stabilize to the point where the interface can be frozen, at least for the early stages of the pipeline. At that point the choice of the format for event data decouples from the interface. Different event formats at various stages along the data-flow path may be adopted based upon considerations of efficiency and prevailing technology. None of this has practical consequences for application code, provided that the interface remains everywhere the same.

10.2 Monte Carlo generators

There are two physics event generators available for use within the HALL D Monte Carlo framework, known as *genr8* [3] and *cwrap* [4]. Both programs are capable of describing a complex decay chain of intermediate states, where decays into two or three bodies are supported at each step. The invariant masses of each particle produced is sampled from a Breit Wigner distribution, whose mass and width is taken from the PDG. A general t-channel process is assumed, with the distribution in t drawn at random from the standard form for a peripheral reaction

$$\frac{d\sigma}{dt} \propto e^{-b|t|}$$

where the b parameter is specified by the user. Both meson and baryon decay chains are allowed. In the case of *genr8* the user may specify the t-distribution in the form of a histogram in place of specifying a value for b in the above formula. be specified by an input histogram.

The angular distributions at each decay vertex are generated according to phase space. This may appear to be a severe restriction in an experiment whose goal is partial-wave analysis, but in fact that is not the case. To see how the physical model for particle spins and decay asymmetries are applied to phase-space Monte Carlo data, see section 10.8.

Both *genr8* and *cwrap* were imported from other experiments, and so write their output events in different and somewhat esoteric formats. To incorporate them into the HALL D framework it was sufficient to provide translators from their private formats to a common hddm format that can be viewed as xml. The present draft specification for the standard xml interface to generated events is found in Ref. [2]. At present a second standard interface is also being supported known as *stdhep*. This somewhat archaic Fortran-based standard was in use by many HEP experiments over the last decade, and there are a number of useful Monte Carlo tools that rely on it, including *MCFast* (see section 10.5). Currently translators exist to supply generated events from either generator through either the xml or the stdhep interface.

Both generators use cryptic private formats for the input data that specify the reaction and desired number of events. At present there does not exist a single unified interface for specifying the reaction to be generated. The task of incorporating *genr8* and *cwrap* into the HALL D framework will not be complete until that interface has been specified, and translators have been written to convert that information from xml to a form understandable to the generators.

10.3 Detector Geometry

One of the most basic requirements for the simulation is access to a detailed description of the geometry of the experiment. Included in geometry is the shape and location of all relevant components, their properties in terms of material composition, density, etc., and the map of the magnetic field. Any objects with which beam particles may interact on their way to a detector are a part of the geometry, starting with the primary collimator and ending with the photon beam dump. Any application within the HALL D framework that needs access to detector geometry data obtains that information through one unified interface. This interface is specified in the form of a xml *document type definition* (DTD) which details what tags exist in the document, what are their arguments, and their structural relationships. The basic structure of the DTD was borrowed from the ATLAS experiment at CERN and adapted for the needs of HALL D. It describes the detector as a tree of volumes, each with specified shape, size, position and material properties. It allows elements to be grouped together and positioned as a unit, so that a survey datum can be expressed by a single element. More details on the interface can be found in Ref. [1].

Application code has access to geometry data through the standard xml libraries. Programs can scan the entire tree or ask for specific pieces of information, such as the position of the center of the target. At present the only consumers of geometry information are the simulation codes. The Geant simulator (see section 10.4) is capable of modeling any geometry, provided that the xml conforms to the DTD. The MCFast simulator (see section 10.5) supports a more limited geometrical description. A special set of tags in the geometry DTD have been created to describe the detector elements in simplified terms for MCFast, in places where the translation from the hierarchical description require some imagination. As more applications are created that depend upon access to specific pieces of geometry information, it will be necessary to extend the interface beyond the DTD to specify the presence and location of specific tags. Investigation is underway to determine if these more complex constraints might be better expressed using xml schema than the DTD.

At present the geometry description is implemented in a set of plain xml text files and organized under a sequential version system. In the future they will probably be stored in a database and indexed by date or run number.

10.4 Physics Simulation

The physics simulation for HALL D is provided by a program called *HDGeant*. The simulator requires four data interfaces: an event source, detector geometry data, simulation control information, and event output. HDGeant is capable of simulating events from any one of three sources.

1. events from a Monte Carlo generator
2. coherent bremsstrahlung source generator
3. automatic single-track generator (for testing)

The first of the three is an external event source described in section 10.2. Events from the generator are distributed uniformly along the length of the beam-target interaction volume and final-state particles followed out into the detector from there. The other two sources are internal to the simulator, and are used for special purposes. The coherent bremsstrahlung source generates uncollimated photons with the energy, angle and polarization characteristics of bremsstrahlung from an oriented diamond radiator. These photons enter the setup upstream of the primary collimator and are followed through the collimator region into the experimental hall, where interactions in the detector are allowed to take place. This simulation mode is useful for estimating detector backgrounds, and for studying the systematics of the collimated photon beam. The single-track generator is used for development of various parts of the simulation, and will be useful later in debugging the event reconstruction package.

The choice of the source for input events is specified in an input file known as the *control* file. Also in the control file are a number of switches that control the simulation mode, such as the number of events to simulate, cutoffs for a variety of physics processes, and debug options. HDGeant obtains the detector geometry directly from the standard geometry interface. Input events from the Monte Carlo generator are accessed through the the standard event interface implemented in the hddm library. Output events are likewise written out using the hddm library.

The output from the simulation is a list of *hits*, which are time and energy data from each detector element that received a signal during the propagation of the event through the detector. The hit data are stored in physical units appropriate to the signal (eg. ns, MeV) which is what the simulation directly produces. No provision is made in the simulator to convert these data back into ADC or TDC data in the form produced by the data acquisition hardware; that

would require couple the simulation to the the detector calibration database, and introduces an unnecessary complication to the simulation. If events in that form were desired at some point, a separate converter could be written to generate simulated raw events from the simulator output.

The major effort in the ongoing development of the simulation is to have a reasonably accurate model of the detector response in each of the detector elements. A basic model presently exists in the code for each of the detector components. These must be improved by the incorporation of intrinsic resolutions for each of the detectors. For example, the impact parameter of tracks in a straw tube of the central drift chamber is converted to a hit time value using a simple linear model for the time *vs* radius. For another, for the response of the lead-glass calorimeter, the total energy loss of charged particles is reported as the hit energy, without taking into account the difference in the Čerenkov response between different kinds of particles. Nevertheless, in its present form the simulator is useful for estimating many aspects of detector performance.

In addition to the detector hits, the simulation is also capable of writing out certain kinds of auxiliary information about the simulated event, for example the actual 3-d points of track impacts on the planes of the forward tracker or the true energy of a photon creating a cluster in the barrel calorimeter. Such information is called *cheat* data because it is not available for real events. However it is invaluable for Monte Carlo studies prior to the development of event reconstruction code, and will be useful in that development for checking the fidelity of the reconstruction.

In Table 10.1 is shown the average time required to simulate a single event on a cpu that is available today, for a few sample reactions. The beam simulation uses the simulator's internal coherent bremsstrahlung generator, and exercises mainly the electromagnetic shower simulation in the collimator region upstream of the detector. The single-track case is included to show the cost of tracking charged particles through the the magnetic field. The gammas show the corresponding cost for photons. The two are put together in the reactions which follow.

In order to obtain a reliable simulation of backgrounds from the collimator region, two enhancements to the standard Geant simulation library were incorporated into HDGeant. The first of these is the addition of hadronic interactions by photons in materials, and the second was Bethe-Heitler muon pair production. The standard Geant electromagnetic shower simulation does not include hadronic photoproduction processes or muon pair production because their cross sections are several orders of magnitude less than the dominant electromagnetic processes and their presence is generally not important to simulating calorimeter response. For the purposes of HALL D however, the

uncollimated beam	44 ms
1GeV π^+ at 15^o	55 ms
3GeV γ at 10^o	200 ms
1GeV γ at 45^o	90 ms
$\gamma p \to \pi^+\pi^- p$	210 ms
$\gamma p \to \pi^+\pi^-\pi^0 p$	430 ms
$\gamma p \to \eta\pi^0\pi^0 p$	670 ms

Table 10.1: Average time required by HDGeant to simulate a single event of various kinds. The tests were carried out on a single Pentium III 1GHz processor. The times are reduced by about a factor of 1.8 on the Athlon MP 1800+ cpu.

high intensities of showers in the collimator enclosure and the heavy shielding against electromagnetic backgrounds makes them important. In particular there are two kinds of penetrating radiation that must be considered: neutrons and high-energy muons.

The incorporation of muon Bethe-Heitler production into Geant was straightforward to do, simply by replicating the code for electron pair production with a changed mass, and the cross section reduced by the factor m_e^2/m_μ^2. The inclusion of photonuclear processes is more daunting. Rather than launch a development of our own, it was decided to incorporate a package that was developed earlier for use by the BaBar experiment known as *Gelhad* [5]. This package breaks provides four models of hadronic photoproduction that are applicable at different scales: single nucleon knockout, two-nucleon knockout via the quasi-deuteron process, single pion photoproduction in the delta-resonance region, and diffractive vector production in the diffractive region. From the point of view of photonuclear physics, this model is far from complete. It will not be used by HALL D to generate photoproduction events in the target. What it does provide is a starting point for estimating neutron fluxes in the hall from the collimator region.

The present HDGeant package is based on the widely-used version 3 of the CERN Geant library. Discussion has started regarding moving the development for HALL D over to the c++ simulation package known as Geant4 that is being used by some of the LHC experiments. Given that the Geant-3 library is written almost entirely in Fortran and is no longer being actively supported by the CERN computer division, its long-term viability depends upon support by the user community. The LHC Alice experiment has taken the major com-

ponents of Geant-3 and wrapped them for use in a c++ environment known as *AliRoot*. The choice of a long-term solution for a physics simulation for HALL D has not yet been finalized.

10.5 Fast simulation

A fast Monte Carlo package has been developed to understand the performance of key aspects of the HALL D detector systems. This package consists of a collection of modules, each serving some particular function. The modules consist of individual programs and library routines which use common event input/output formats. Figure 10.2 illustrates this modular structure.

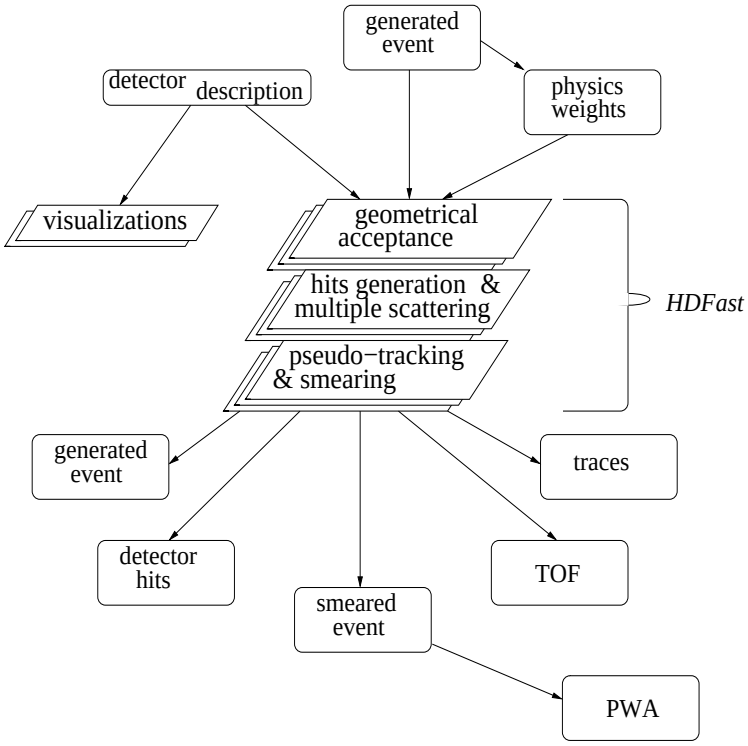

Figure 10.2: An illustration giving an overview of the HALL D Monte Carlo software which emphsizes its modular nature.

First, a Monte Carlo four-vector generator is used to create phase–space distributed events. Next is the detector simulation, *HDFast*, which is a fast and flexible simulation program based upon the MCFast package developed

by the simulation group at Fermilab. The Monte Carlo output includes (but is not limited to) the following data objects:

- generated event
- detector hits
- resolution modified (smeared) event
- time-of-flight information
- dE/dx information
- threshold Čerenkov information
- particle trajectory information

HDFast is a fast and flexible simulation program based upon the MCFast package developed by the simulation group at Fermilab[6]. MCFast consists of a set of modularized Monte Carlo library routines. It is designed to perform parameterized tracking by assembling a covariance matrix for each track that takes into account materials, efficiencies, and resolutions for all measurement planes, and use this matrix to smear the track parameters randomly. The covariance matrix is first diagonalized so as to properly account for effects due to correlations when parameters are smeared. In principle, the distribution of smeared tracks produced by this method would be similar to the distribution of real tracks that were measured by a real detector(with the same parameters) and analyzed with an idealized track fitting procedure[6].

HDFast is controlled via a set of user routines which act as an interface to the MCFast package. They control the tracking and smearing of the four-vectors, in addition to the booking and filling of monitoring ntuples and histograms. The detector geometry is controlled by an ascii file which is read in during program execution. This allows the user to quickly create or modify the detector geometry without the need to recompile the executable. In addition, ROOT[7] was used to develop an event display which reads in the ascii geometry file and displays a two-dimensional visualization (see Figure 10.8) of the detector configuration and event track projections.

10.6 Acceptance studies

In order to better understand the effects of finite acceptance of a proposed detector configuration, a simple study of the acceptance as a function of total

meson effective mass for various final states has been performed. In doing the Monte Carlo acceptance studies we considered the following reactions: schematically shown in Figure 10.8. This configuration is composed of the following:

- 2.24 Tesla solenoid magnet –LASS magnet,
- 5-layer Vertex Chamber (VTX),
- 22-layer Central Drift Chamber (CDC),
- 5 6-layer Forward Drift Chambers (FDC),
- Barrel Calorimeter which also acts as central TOF(BCAL),
- Cerenkov Detector,
- Forward time-of-flight (FTOF),
- Forward Lead Glass Detector (LGD) 172x172 cm with 8x8 cm beam hole,
- target-beam vertex distribution at $r = 0.0\,cm$, $z = 50\,cm$ with $\sigma_r = 0.3\,cm$, $\sigma_z = 15.0\,cm$ ($\hat{z}$ is along the magnet axis; the origin is located at the upstream face of the solenoid).

10.6.1 Acceptance performance

In the simulation, an event was accepted if the following minimum conditions were met:

- all charged tracks were found with a minimum of four hits per track, and
- all gammas were detected in either the BCAL and/or LGD.

The acceptance as a function of total effective meson mass is shown in Figure 10.3. It is important to note that at higher beam energies the forward boost results in more forward-going high-momentum tracks. And even though the mass acceptance seems good, the resolution of the forward-going higher-momentum tracks degrades. This issue has been studied in detail and is discussed in HALL D Note #7[8].

In Figure 10.4 through Figure 10.7, we show the acceptance for the Gottfried-Jackson decay angles (the particle decay angles often used in the partial wave analysis). It is clear that the Gottfried-Jackson angular acceptance is quite good. The acceptance for gammas is also rather high, but it suffers more from

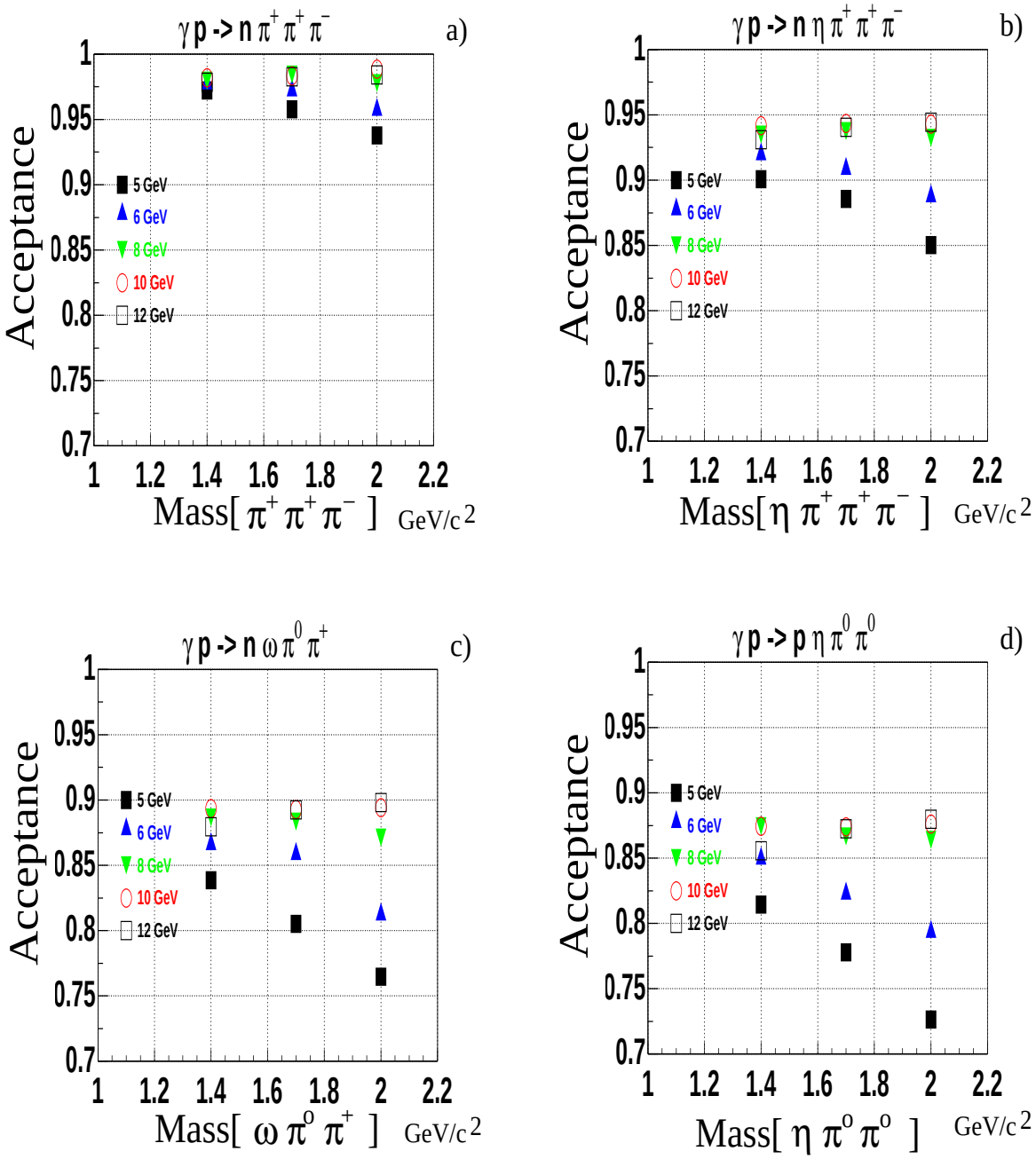

Figure 10.3: The acceptance as a function of total effective meson mass: a) $X^+ \to \pi^+\pi^+\pi^-$, b) $X^+ \to \eta\pi^+\pi^-\pi^+$, c) $X^+ \to \omega\pi^0\pi^+$, d) $X^0 \to \eta\pi^0\pi^0$. The acceptance studies were performed for effective meson masses of 1.4, 1.7, and 2.0 GeV/c^2, and at each mass point the photon beam energy was varied from 5 to 12 GeV.

holes in the forward and backward regions. The hole in the backward region results from backward-going gammas, which is the dominant factor at lower beam energies. The forward hole, due to gammas passing through the beam hole in the LGD, becomes important for higher beam energies. Figure 10.8a displays an event for reaction $\gamma p \to p\eta\pi^0\pi^0$ at $Mass(X) = 2.0\ GeV/c^2$ and beam of 5 GeVthat was lost due to the upstream hole. For this channel 75% of the lost events were of this type. On the other hand, for a 12 GeV beam and the same final state about 50% of the lost events are due to the beam hole(See Figure 10.8b). While the beam hole is unavoidable, the hole in the backward region suggests the need at the lower beam energies for a backward gamma veto. Regardless of this, the acceptance for the Gottfried-Jackson decay angles is flat and not strongly dependent on $Mass(X)$ or the beam energy. This is important for partial wave analysis because, although the effects of acceptance distortions are accounted for in the method, large acceptance corrections can lead to large systematic errors in the results.

10.7 Monte Carlo Study of Photon Energy Resolution

In this study the GENR8 program was used to generate the events. Four different exclusive reactions were studied, two with photons produced at the baryon vertex:

$$\gamma p \to N^*(1500)\pi^+ \to (n\eta)\pi^+ \to n\pi^+\gamma\gamma \tag{10.1}$$

$$\gamma p \to X^+(1600)\Delta^0 \to (\pi^+\pi^+\pi^-)(n\pi^0) \to \pi^+\pi^+\pi^- n\gamma\gamma \tag{10.2}$$

The Δ^0 reaction (reaction 10.2) has a 3π-meson mass of $1.600 GeV/c^2$, and a width of 300 MeV/c^2. The two meson vertex reactions are:

$$\gamma p \to X^+(1600)n \to (\eta\pi^+)n \to n\pi^+\gamma\gamma \tag{10.3}$$

$$\gamma p \to X(1600)p \to (\pi^+\pi^-\pi^o)p \to p\pi^+\pi^-\gamma\gamma \tag{10.4}$$

In both reactions 10.3 and 10.4, the meson systems were generated with a Breit–Wigner distribution of mass $1.6 GeV/c^2$and a width of $0.3 GeV/c^2$.

Each of the above reactions were simulated using a beam energy of $8 GeV$, and a t-channel slope of $5 GeV/c^2$. The production and decay vertex was assumed to be at the center of the target. For each system, 10,000 events were generated. The direction and energy of the photons were recorded and analyzed.

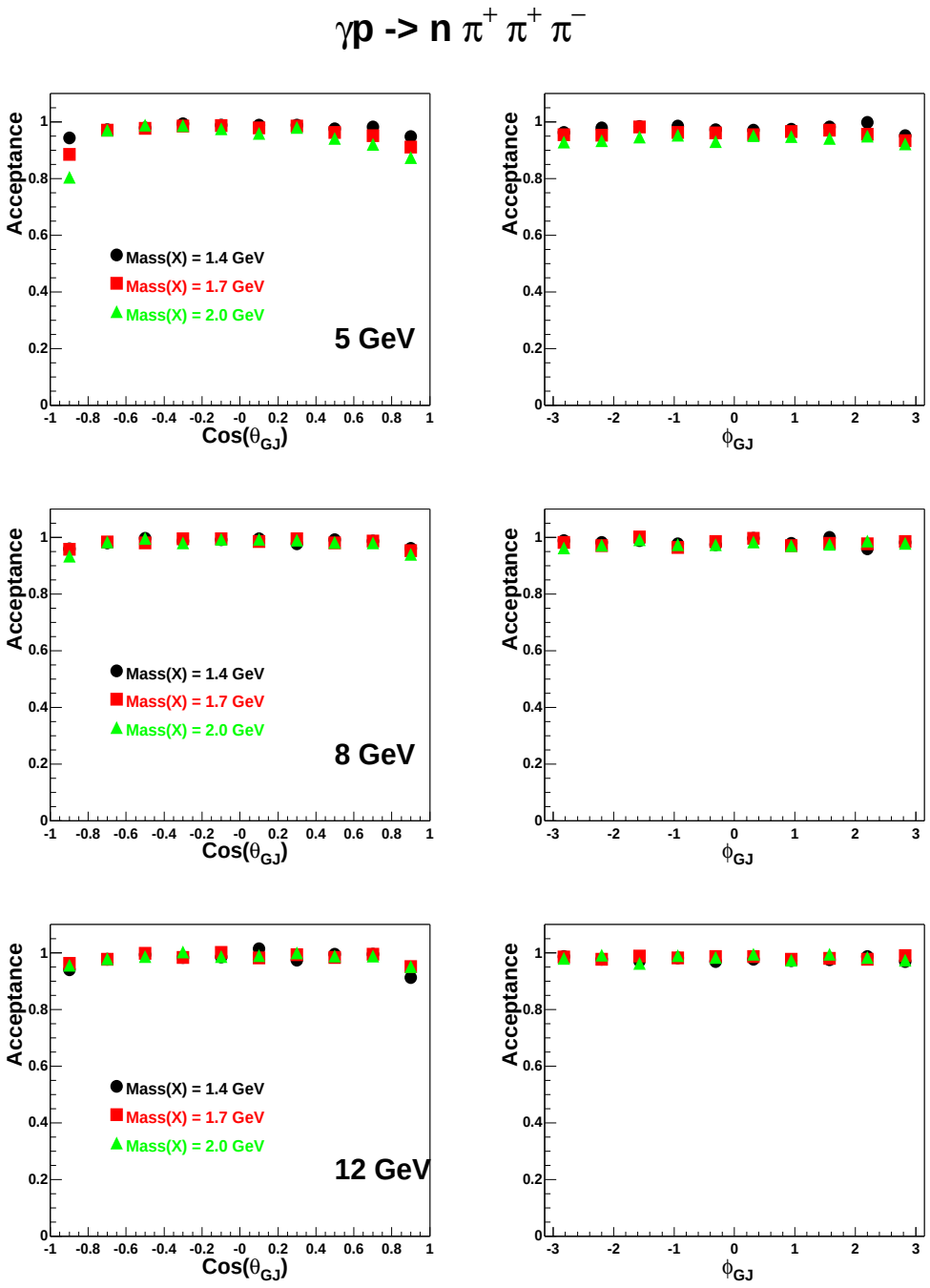

Figure 10.4: The acceptance in $cos(\theta_{GJ})$ and ϕ_{GJ} for $X^+ \to \pi^+\pi^+\pi^-$. The acceptance was studied for X^+ effective masses of 1.4, 1.7, and 2.0 GeV/c^2, and for different photon beam energies of $5GeV$ (top), $8GeV$ (middle), and $12GeV$ (bottom).

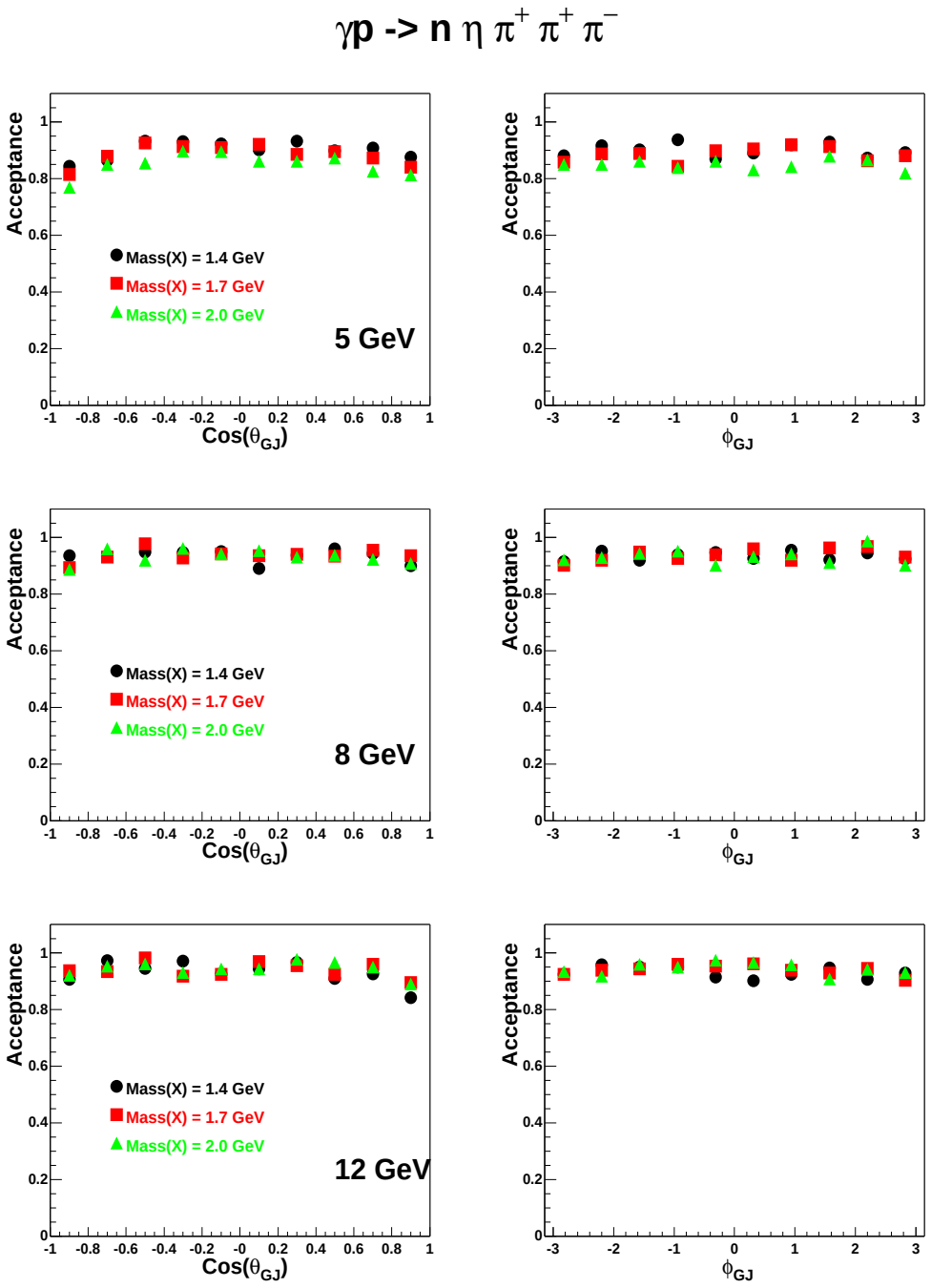

Figure 10.5: The acceptance in $cos(\theta_{GJ})$ and ϕ_{GJ} for $X^+ \to \eta\pi^+\pi^+\pi^-$. The acceptance was studied for X^+ effective masses of 1.4, 1.7, and 2.0 GeV/c^2, and for different photon beam energies of $5GeV$ (top),$8GeV$ (middle), and $12GeV$ (bottom).

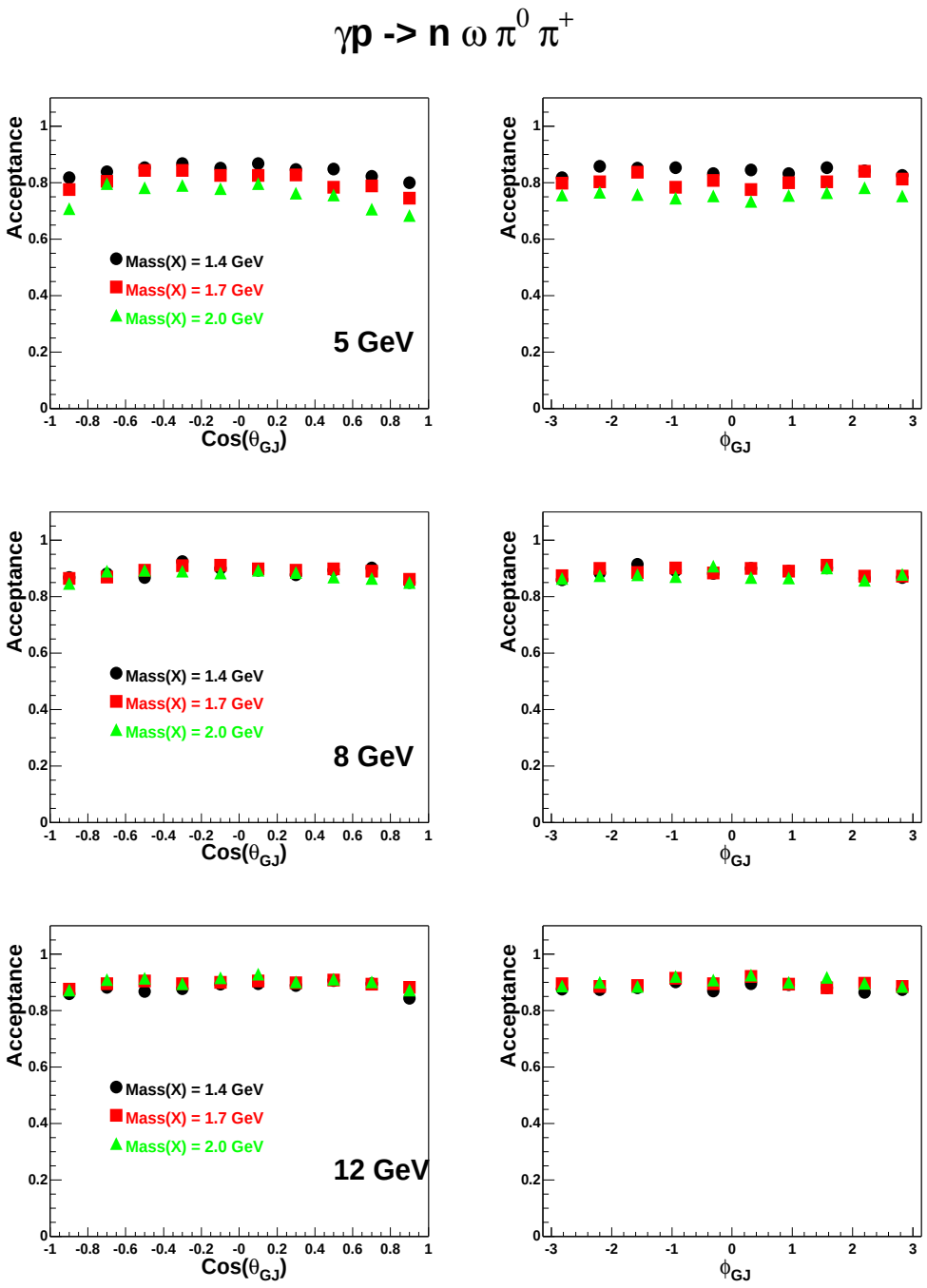

Figure 10.6: The acceptance in $cos(\theta_{GJ})$ and ϕ_{GJ} for $X^0 \rightarrow \omega\pi^0\pi^+$. The acceptance was studied for X^+ effective masses of 1.4, 1.7, and 2.0 GeV/c^2, and for different photon beam energies of $5GeV$ (top), $8GeV$ (middle), and $12GeV$ (bottom).

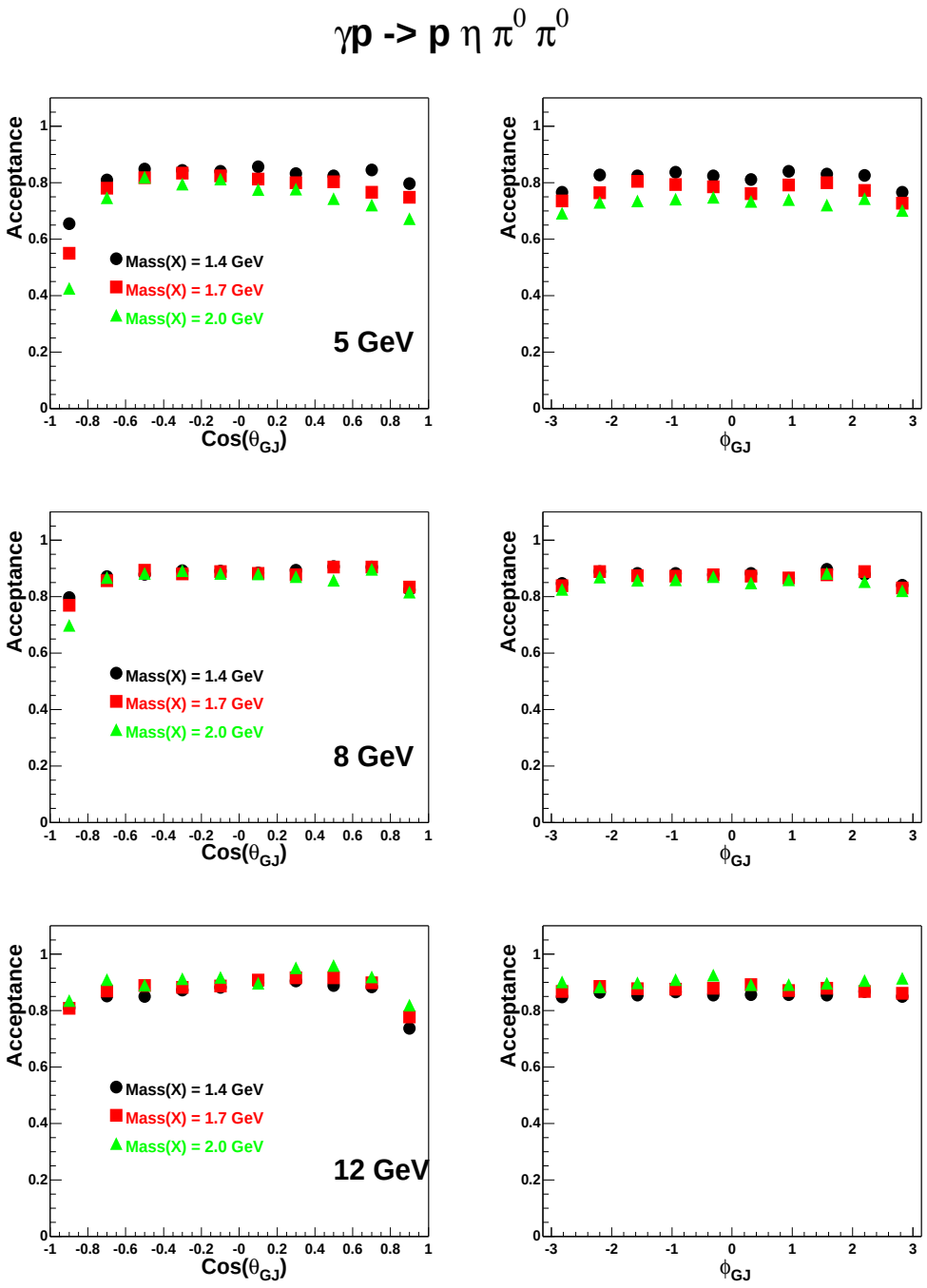

Figure 10.7: The acceptance in $cos(\theta_{GJ})$ and ϕ_{GJ} for $X^0 \rightarrow \eta\pi^0\pi^0$. The acceptance was studied for X^+ effective masses of 1.4, 1.7, and 2.0 GeV/c^2, and for different photon beam energies of $5GeV$ (top), $8GeV$ (middle), and $12GeV$ (bottom).

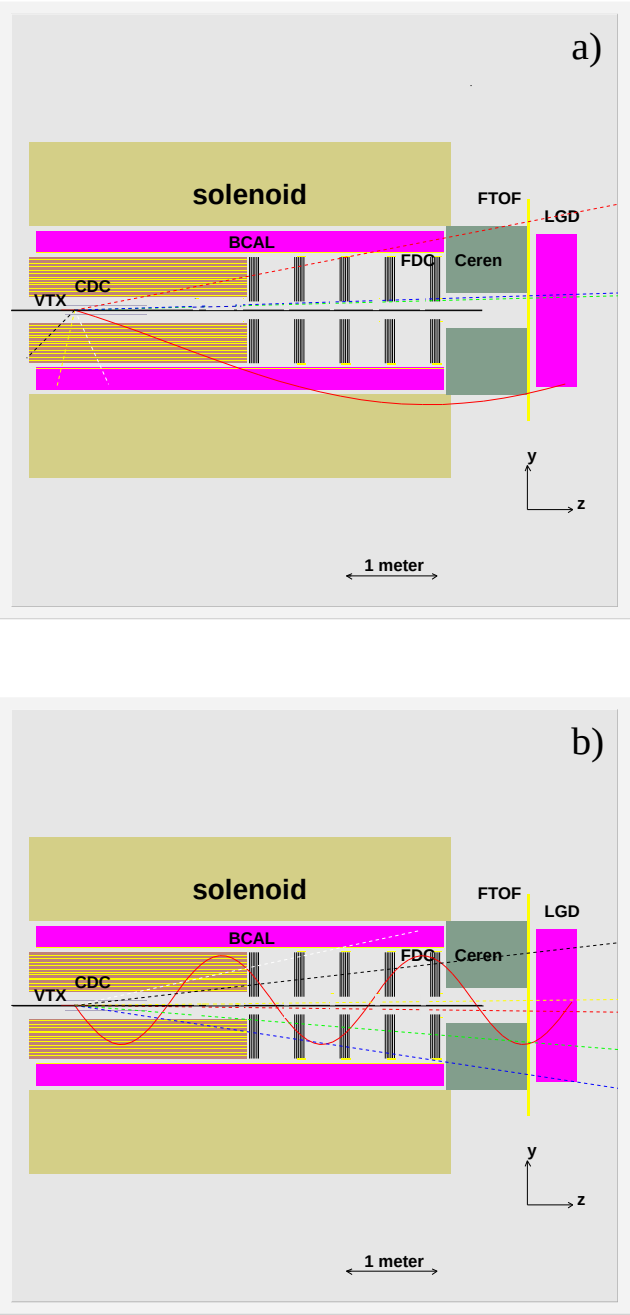

Figure 10.8: Event displays of lost events for $\gamma p \to p\eta\pi^0\pi^0$ for $Mass(X) = 2.0$ GeV/c^2: (a) backward missed gamma at a beam energy of 5 GeV, and (b) forward missed beam hole gamma at a beam energy of 12 GeV. The events shown contain both charged particles (solid lines) and photons (dashed lines).

10.7.1 Photon Detector Energy Resolution

The photons produced in the above decays were traced into the Barrel Calorimeter and the Lead Glass Detector. Figure 10.9 and 10.10 show the percentage of photons that would enter, but not be detected by the Barrel Calorimeter due to the minimum energy thresholds.

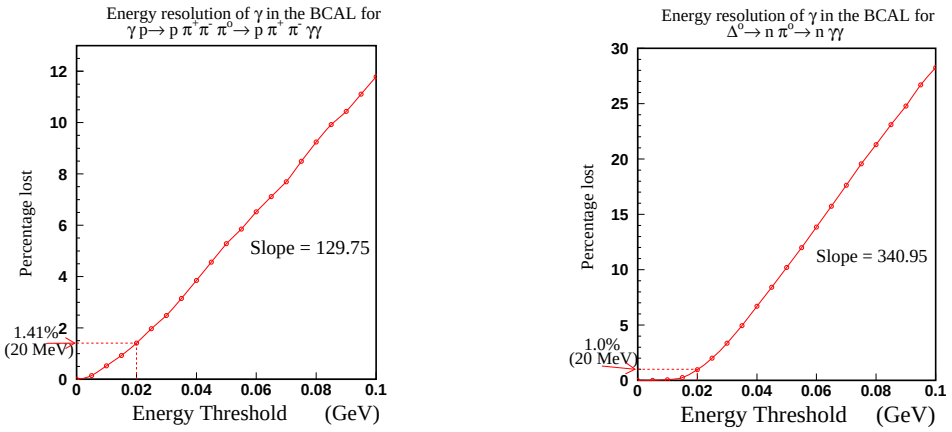

Figure 10.9: The percentage loss due to the energy threshold of the BCAL. Left is for reaction 10.4, while the right figure is for the Δ^0 decay from reaction 10.2. The percent of the total photons entering the Barrel Calorimeter for reaction 10.4 is 57% and reaction 10.2 is 87%.

Currently, the design calls for the energy sensitivity of 20 MeV for the Barrel Calorimeter. One can see that this results in around a 1% loss of photons which is quite acceptable. However, if this energy can not be met, the percentage of photons lost rises rapidly with the increased energy threshold, especially for the Δ^0 (reaction 10.2) decay. For example, if the threshold is 50 MeV, then 5% of the 3π reaction is lost, and 10% of the Δ^0 reaction is lost. The situation for the η reactions is not so severe, as would be expected from the higher energy photons in the η decay (figure 10.10).

The results for the Lead Glass Detector are similar, but the percentage rise is not so significant at higher energy thresholds. The only system with significant loss in the lead glass array is the 3π (reaction 10.4) decay. At the sensitivity threshold of 100 MeV, the lead glass detector will not see 0.718% of the photons. The design calls for a $150MeV$ detection minimum in the LGD. At this energy, the detector will miss 1.86% of the photons (figure 10.11).

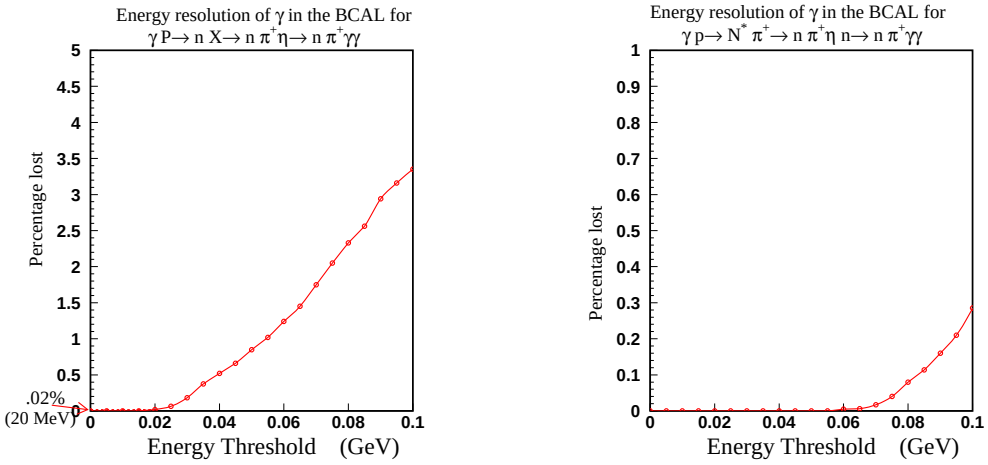

Figure 10.10: The percentage loss due to the energy threshold of the BCAL. The left plot is for reaction 10.3, and right is for reaction 10.1. The percent of all the photons entering the Barrel Calorimeter for the η X (reaction 10.3) and the η N^* (reaction 10.1) are 55% and 88% respectively.

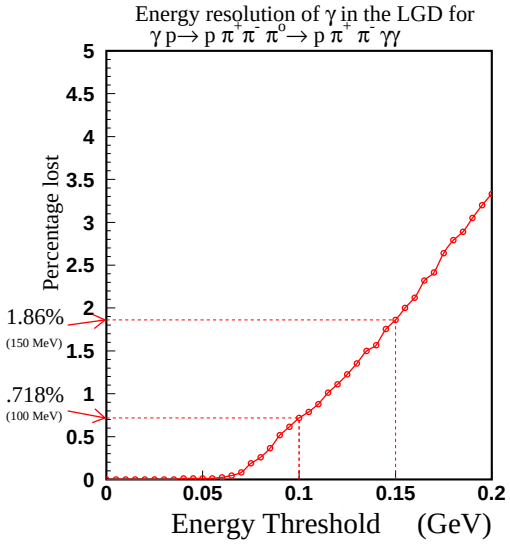

Figure 10.11: The percentage of undetected photons for a given energy threshold of the lead glass detector. From reaction 10.4.

10.8 Physics Event Weighters

Conceptually, what one would like to do in the analysis of any given reaction is to write down as complete as possible a set of diagrams leading to the final state and sum their amplitudes as a function of a minimal set of unknown parameters. This model would then be fed to the event generator to produce a Monte Carlo sample which could be reconstructed and compared to the data. By repeating this procedure for different values of the parameters through a fitting procedure, the best values of the parameters and an overall evaluation of the model could be derived.

Practically, this is not what is done because it is too expensive to recompute the entire Monte Carlo sample at every step in the fit. Instead a single Monte Carlo sample is produced using an initial crude approximation to the physics model distribution, and then corrections are applied using a weighting procedure after the sample has been simulated and reconstructed. The initial approximation is defined by the following three simple assumptions; (a) particles from high-energy photoproduction are produced independently from meson and baryon vertices; (b) the momentum separation between the two vertices is described by an exponential distribution in the Mandelstam variable t; (c) within each vertex the particles are produced through a cascade of two- and three-particle decays which are each distributed according to a phase-space density function. If this approximation were an adequate model of the physics then there would be no need for the HALL D experiment. Nevertheless it is a useful starting point because it can be used to produce a Monte Carlo sample of events with adequate coverage over the full kinematic range of interest.

Assuming the independence property of the Monte Carlo sampling technique, every event in the Monte Carlo sample is independent of the others. Each reconstructed Monte Carlo event carries with it the information about the original generated kinematics, from which the physics amplitudes can be calculated. For a given set of model parameters these amplitudes can be summed to form a probability for each event, which is called a *weight*. If all sums over the Monte Carlo sample during partial wave analysis are carried out including these weight factors then the foregoing conceptual procedure is recovered. Although the statistical errors in the weighted Monte Carlo sample are no longer simple Poisson factors, they are straightforward to calculate. In general these errors are larger for the weighted technique than for an unweighted procedure, but that is readily offset by generating a somewhat larger sample. Exactly how much larger depends on how different the weighted distribution is from the initial, but usually this factor is not larger than two.

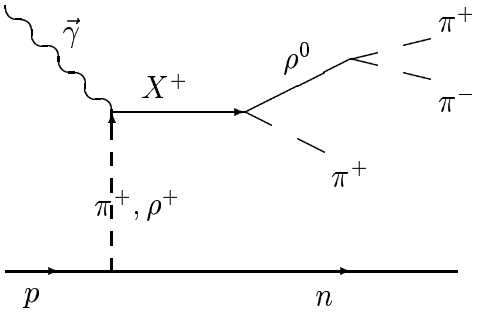

Figure 10.12: A schematic diagram of the photoproduction amplitude via one–π or ρ exchange. The state X^+ then decays via $\rho^0\pi^+$, and the ρ subsequently decays into $\pi^+\pi^-$.

Ultimately it is not known until the final stages of the analysis how large a Monte Carlo sample is adequate for any given channel, but for the purposes of the design a conservative factor of 10 more Monte Carlo than real events has been adopted as a benchmark.

The above method is well-established for partial wave analyses in high-energy physics. To gain experience within the context of HALL D it was decided to apply the procedure to a photoproduction reaction. To this end, an event generator for the 3π final state has been written using the one–pion charge–exchange mechanism as discussed in reference [9] for reaction 10.5.

$$\vec{\gamma}p \to X^+n \quad \left(X^+ \to \left[\rho^0 \to \pi^+\pi^-\right]\pi^+\right) \tag{10.5}$$

A schematic of this process is shown in Figure 10.12. One–π charge exchange requires both a spin–flip at the nucleon vertex, and that the X^+ particle carry the helicity of the incoming γ, ($M_X = 1$). Any number of resonances X^+ with different masses, widths and production strengths can be included in the generator. In addition, the photon beam can have any polarization desired. An extension to this program allows for ρ–exchange under the same conditions as the π exchange. These two amplitudes represent *unnatural* and *natural* parity exchanges resepctively. Events produced using one of the phase space generators can then be weighted according to the physics weighter, and then passed through the HALL D Monte Carlo program. These can then be used as input to the partial wave analysis as described in the next chapter.

A sample of the output of this generator is shown in Figure 10.13. These events have been generated with four resonances: $a_1(1260)$, $a_2(1320)$, $\pi_2(1670)$ and an exotic $\pi_1(1600)$. The masses and widths are all consistent with current accepted values. In addition, one can see the ρ° in the $\pi^+\pi^-$ invariant mass spectra. A full list of known resonances [10] that could be put in this generator is given in table 10.2.

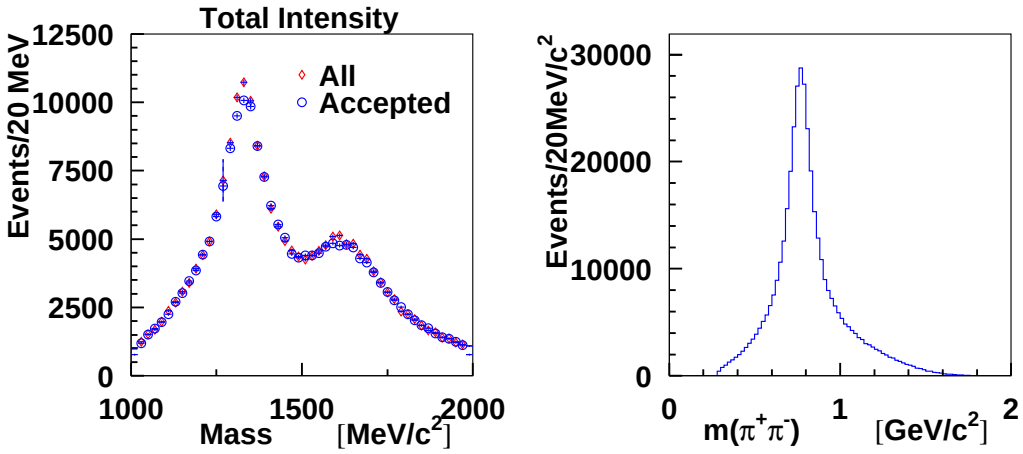

Figure 10.13: (Left) The genrated 3π mass spectrum using four intermediate resonances, X^+. The diamonds correspond to all generated events, while the circles correspond to events which have been run through the HALL D Monte Carlo program. (Right) The $\pi^+\pi^-$ invariant mass from the 3π events. The peak corresponds to the $\rho^0(770)$.

Resonance	Mass	Width	$L_{\rho\pi}$
$a_1^+(1260)$	$1.230 GeV/c^2$	.250 to $.600 GeV/c^2$	$L = 0, 2$
$a_2^+(1320)$	$1.318 GeV/c^2$	$.105 GeV/c^2$	$L = 2$
$\pi_1^+(1600)$	$1.593 GeV/c^2$	$.168 GeV/c^2$	$L = 1$
$a_1^+(1640)$	$1.640 GeV/c^2$	$.300 GeV/c^2$	$L = 0, 2$
$a_2^+(1660)$	$1.660 GeV/c^2$	$.280 GeV/c^2$	$L = 2$
$\pi_2^+(1670)$	$1.670 GeV/c^2$	$.259 GeV/c^2$	$L = 1, 3$
$a_2^+(1750)$	$1.752 GeV/c^2$	$.150 GeV/c^2$	$L = 2$
$a_4^+(2040)$	$2.014 GeV/c^2$	$.361 GeV/c^2$	$L = 4$
$\pi_2^+(2100)$	$2.090 GeV/c^2$	$.625 GeV/c^2$	$L = 1, 3$
$a_6^+(2450)$	$2.450 GeV/c^2$	$.400 GeV/c^2$	$L = 6$

Table 10.2: A list of known charged 3π resonances that could be produced in photoproduction and decay via $\rho\pi$. The column $L_{\rho\pi}$ are the allowed orbital angular monetum between the ρ and the π when the resonance decays. Because we require non–zero isospin, many states can not be produced.

List of Figures

List of Tables

Bibliography

[1] R. Jones, 2001. HDDS - Hall D Detector Specification (http://zeus.phys.uconn.edu/halld/geometry/).

[2] R. Jones, 2001. HDDM - Hall D Data Model (http://zeus.phys.uconn.edu/halld/datamodel/doc/).

[3] P. Eugenio. *Genr8*: A general monte carlo event generator. Technical report, Carnegie Mellon University, 1998.

[4] S. Teige. A monte-carlo user guide. Technical report, Indiana University, 1998.

[5] Art Snyder, 1995. GELHAD in BBSIM (http://www.slac.stanford.edu/BFROOT/www/Computing/Offline/Simulation/gelhad.html)

[6] Information on MCFast may be optained via the WWW at http://fnpspa.fnal.gov.

[7] Information on ROOT may be optained via the WWW at http://root.cern.ch.

[8] C. A. Meyer. Tracking Resolution Requirements in the Meson Spectroscopy Facility at Jefferson Lab. Technical Report HallD Note **7**, Carnegie Mellon University, 1998. `http://www.phys.cmu.edu/halld/notes_main.html`.

[9] A. V. Afanasev and A. P. Szczepaniak. Charge exchange $\rho^0\pi^+$ photoproduction and implications for searches for exotic mesons. *Phys. Rev. D*, **61**:114008, 2000.

[10] D. E. Groom, *et al.* (Particle Data Group). Review of particle physics. *Eur. Phys. Jour. C*, **15**:1, 2000.