CLAS-NOTE-90-003

Fast Monte Carlo Program for the CLAS Detector

ELTON S. SMITH   FEBRUARY 16, 1990

Contents

1 Overview **1**

2 Detector Configuration **1**

2.1 Acceptance 4
2.2 Resolution 5
2.3 Detector Response 21

3 How to use FAST **24**

3.1 Input File 25
3.2 Examples 28
  3.2.1 Example 1: Resolution 28
  3.2.2 Example 2: Acceptance 28
  3.2.3 Example 3: Missing Mass 29
  3.2.4 Example 4: Particle Identification 29
  3.2.5 Example 5: Hardware Trigger Simulation 32
  3.2.6 Example 6: N* Analysis 33

4 Summary **33**

A LUND Variables **33**

B FAST Subprograms **34**

List of Figures

1 Side View of the CLAS detector. 2
2 Front View of the CLAS detector. 3
3 Azimuthal Acceptance. 6
4 Acceptance to Reach Region II Drift Chambers (Inbenders). . 7
5 Acceptance to Reach Region II Drift Chambers (Outbenders). 8
6 Acceptance for the trigger counters (Inbenders). 9
7 Acceptance for the trigger counters (Outbenders). 10
8 Acceptance for the shower counters (45°) (Inbenders). 11
9 Acceptance for the shower counters (45°) (Outbenders). 12
10 Acceptance for the shower counters (90°) (Inbenders). 13
11 Acceptance for the shower counters (90°) (Outbenders). 14
12 Additional Acceptance Requirements. 15
13 Fractional Momentum resolution (FWHM). 16
14 Angular resolution (FWHM). 17
15 Angular resolution (FWHM). 18
16 Vertex resolution (FWHM). 19
17 Timing resolution (r.m.s.). 20
18 Response of shower counter. 22
19 Missing Mass distribution of $ep \rightarrow e'N^*(1535)$. 30
20 Flight time vs momentum. 31

List of Tables

1 Command procedure for Example #1 26
2 Acceptance comparison between FAST and GEANT 29
3 LUND Particle Identification Codes. 35

1 Overview

The FAST Monte Carlo program has been developed for the purpose of rapidly and reliably estimating counting rates and expected resolution in the CLAS detector. FAST is intended for use in preparation of proposals and generating expected experimental distributions. However, FAST is a parametric Monte Carlo [1] which cannot replace detailed simulation of the detector with a program such as GEANT. FAST reproduces the general characteristics of CLAS and that is often all that is required.

FAST does not trace particle trajectories through the detector. Rather, the detector response to a particle originating in the target is estimated using parameterized functions for resolution, acceptance and detector response. These functions were determined by fitting the outputs of other programs [2, 3] that incorporate the details of geometry and physical response to particle interactions. Thus, a given version of the FAST Monte Carlo has a fixed geometry. Detector design is best accomplished by the programs that were used to compute the detector response used as input by the FAST Monte Carlo.

This document summarizes and supersedes previous descriptions of the FAST Monte Carlo [4, 5].

2 Detector Configuration

The detector geometry assumed by the FAST Monte Carlo is specified in the last Semi-annual Report to DOE [6]. This configuration uses the CIRCE coil shape and is illustrated in Figs 1 and 2. Briefly, the detector consists of the following detector elements beginning from the target and proceeding outwards. A small normal conducting magnet (magnetic shield) protects the drift chambers from low-momentum backgrounds spraying from the target and allows detectors to be placed in the region of the target (region I). Thus the drift chamber system consists of three packages corresponding to the target region (CH1 in region I), the region inside the magnetic field (CH2 in region II) and surrounding the large toroid (CH3 in region III).

An array of Cerenkov counters (60cm thick) occupy the space between the outer drift chambers CH3 and the array of scintillator counters used for triggering and particle identification by time-of-flight. The shower counters

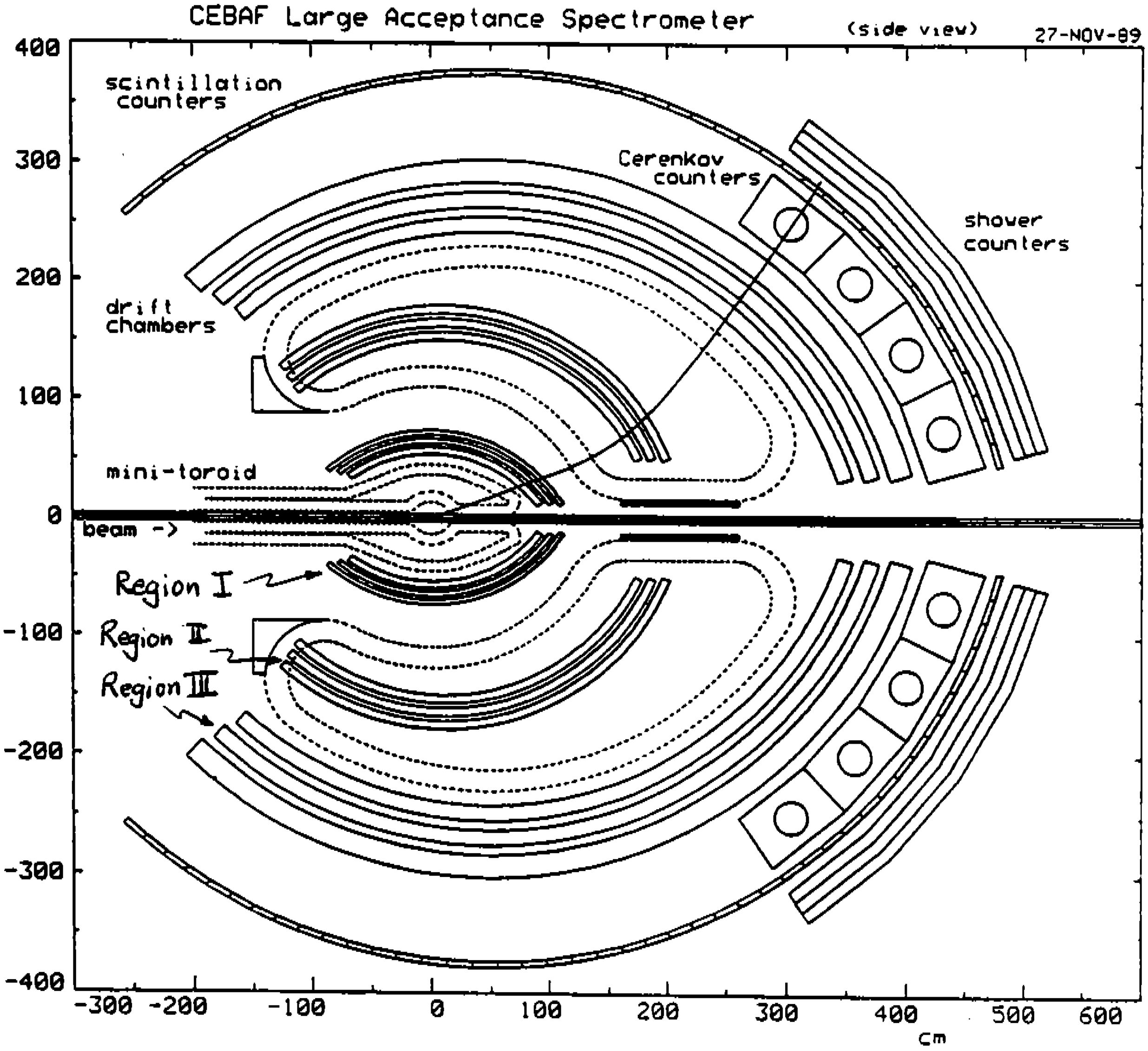

Figure 1: Side view of the CLAS detector, which includes the three drift chamber planes (regions I, II and III), the Cerenkov counters, time-of-flight scintillators and shower counters.

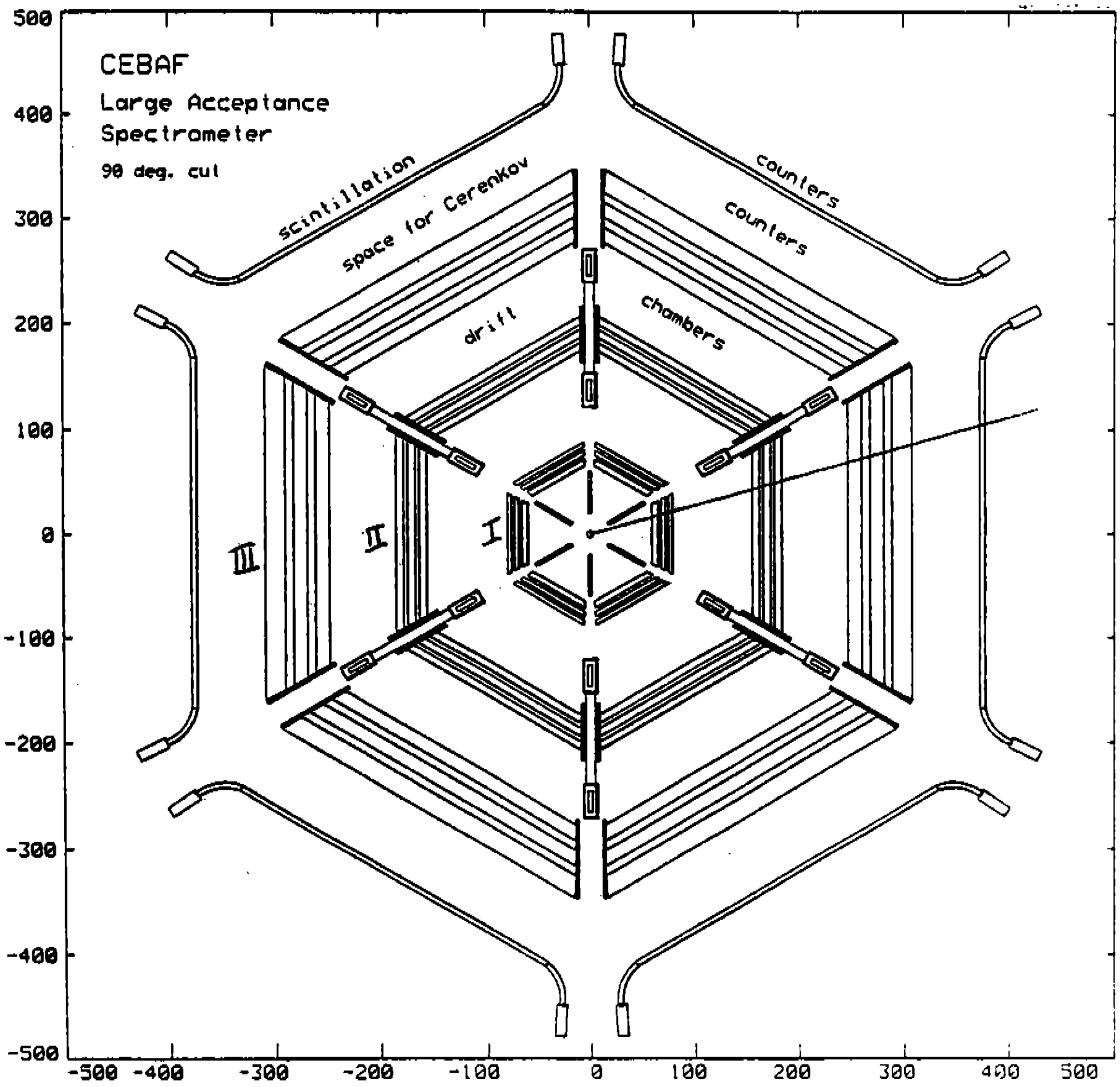

Figure 2: Front view of the CLAS detector, which includes the three drift chamber planes (regions I, II and III), the Cerenkov counters, time-of-flight scintillators and shower counters.

are placed outside the scintillation plane. The identification of electrons is accomplished with the combination of Cerenkov and shower counters. In five sectors they cover an angular region up to 45°, and in sector #1 (ϕ=0) the coverage extends to 90° for both detector systems. Thus there are six detector planes (CH1, CH2, CH3, CER, SCI, SHW) for which FAST computes various quantities such as acceptance, response, etc.

For every track read in from the Physics input data file the program computes the acceptance, resolution and response of each of the six systems mentioned above. The acceptance is stored in the common logical variable LACC(jpart,jdet) (jpart=track number, jdet=1-6 for each detector). If the energy deposition in a detector is above a threshold, the appropriate logical variable LEFF(jpart,jdet) is set true. The momentum variables are also smeared with the appropriate resolution. No action is taken until the analysis stage based on the results of these computations. Therefore an analysis may choose to use any or part of the information available to him. For example one may plot kinematical variables for all tracks or only those which satisfy certain acceptance criteria. Upcoming sections discuss how these quantities are estimated.

2.1 Acceptance

The CLAS detector has an acceptance for particles originating in the target which is a correlated function of (p/B) and scattering angle θ of the track (see Figures 3-6). B is the intensity of the magnetic field and p is the particle momentum. Generally, a minimum momentum ($\sim$ 200MeV) is required to overcome the magnetic field and reach a given detector plane. In addition, the coils restrict the azimuthal angle ϕ periodically every 60°(see Figure 3). Small Correlations between ϕ and momentum are ignored in this program.

The acceptance is also a function of the particle charge × magnet polarity. Thus each acceptance region is parameterized depending on whether a particle bends toward or away from the beam axis. An additional dependence must be included when one varies the position of the target, which is specified in the input file (see Section 3.1). [1] The acceptance functions are

---

[1] When one moves the target upstream, one may wish to move the drift chambers from region I as well. Which configuration is desired depends on the particular experiment. A data statement in the subroutine ACCEPT_CH1 allows positioning the region I drift chambers independently of the detector and target positions.

represented by parabolic B-splines to avoid the inherent problems caused by high-order polynomials and the dependence on the target position is specified by coefficients which depend only linearly on this variable. The use of splines constrains extrapolations of the acceptance functions to be well-behaved over a wide range of target positions (Figures 4-11).

The minimum momentum required to reach a given detector plane may be determined by factors other than the magnetic field. Heavy charged particles may range out before reaching a given detector and slow neutrons must reach the detector plane in a reasonable detection time (~ 100ns). Figure 12 shows the minimum momentum required for various particles to reach each of the detector planes by these two criteria which are included in the acceptance functions as additional requirements.

2.2 Resolution

The contributions to momentum and angular resolution due to position measurement (pos) uncertainties and multiple scattering (ms) were determined as a function of θ for 1GeV/c particles with the program MOMRES [3]. For simplicity, each contribution was then fit to a parabola which guaranteed proper behavior over the interval of interest (θ=0°–150°) and are shown in Figures 13-16. The resolution for arbitrary momenta can then be obtained by appropriate scaling:

$$\left(\frac{\sigma_p}{p}\right)^2 = \frac{1}{B}\left(p\left(\frac{\sigma_p}{p}\right)^2_{pos} + \frac{1}{\beta}\left(\frac{\sigma_p}{p}\right)^2_{ms}\right) \qquad (1)$$

$$\sigma_\phi{}^2 = \sigma_{\phi_{pos}}^2 + \frac{1}{p\beta}\sigma_{\phi_{ms}}^2 \qquad (2)$$

$$\sigma_\theta{}^2 = \sigma_{\theta_{pos}}^2 + \frac{1}{p\beta}\sigma_{\theta_{ms}}^2, \qquad (3)$$

where β has the usual meaning of velocity. Note that only the momentum resolution depends on the magnetic field intensity B. The contributions at p=1GeV/c were obtained for full field (B=1) and assuming that the particle trajectory traverses all drift chambers. For many experiments, the interaction vertex may be defined independently of the drift chamber measurements. In such cases, measurements of the momentum and angle of a track may be improved by using this information. The improvement due to such knowledge

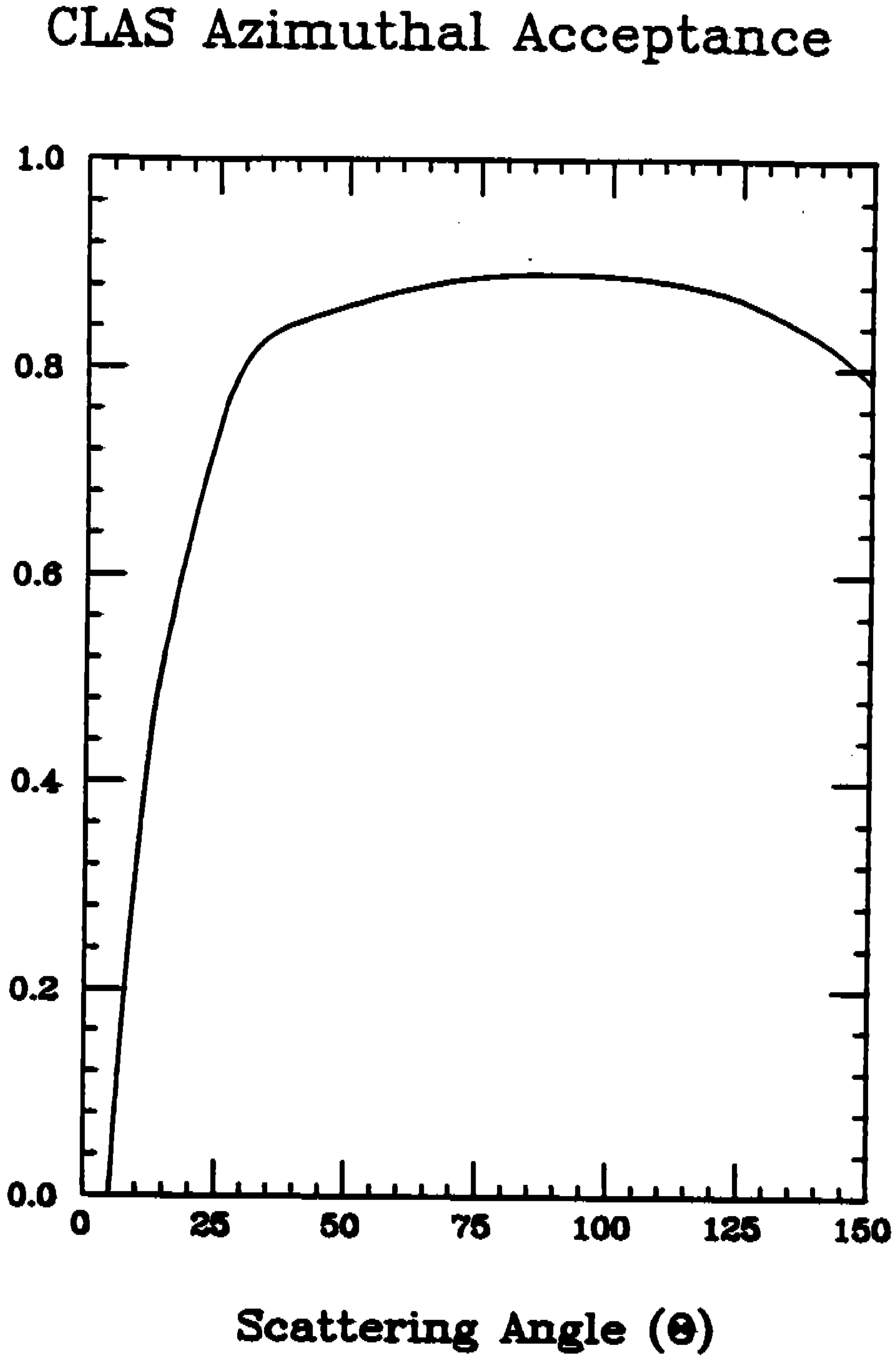

Figure 3: The azimuthal acceptance is determined by the size of the six coils.

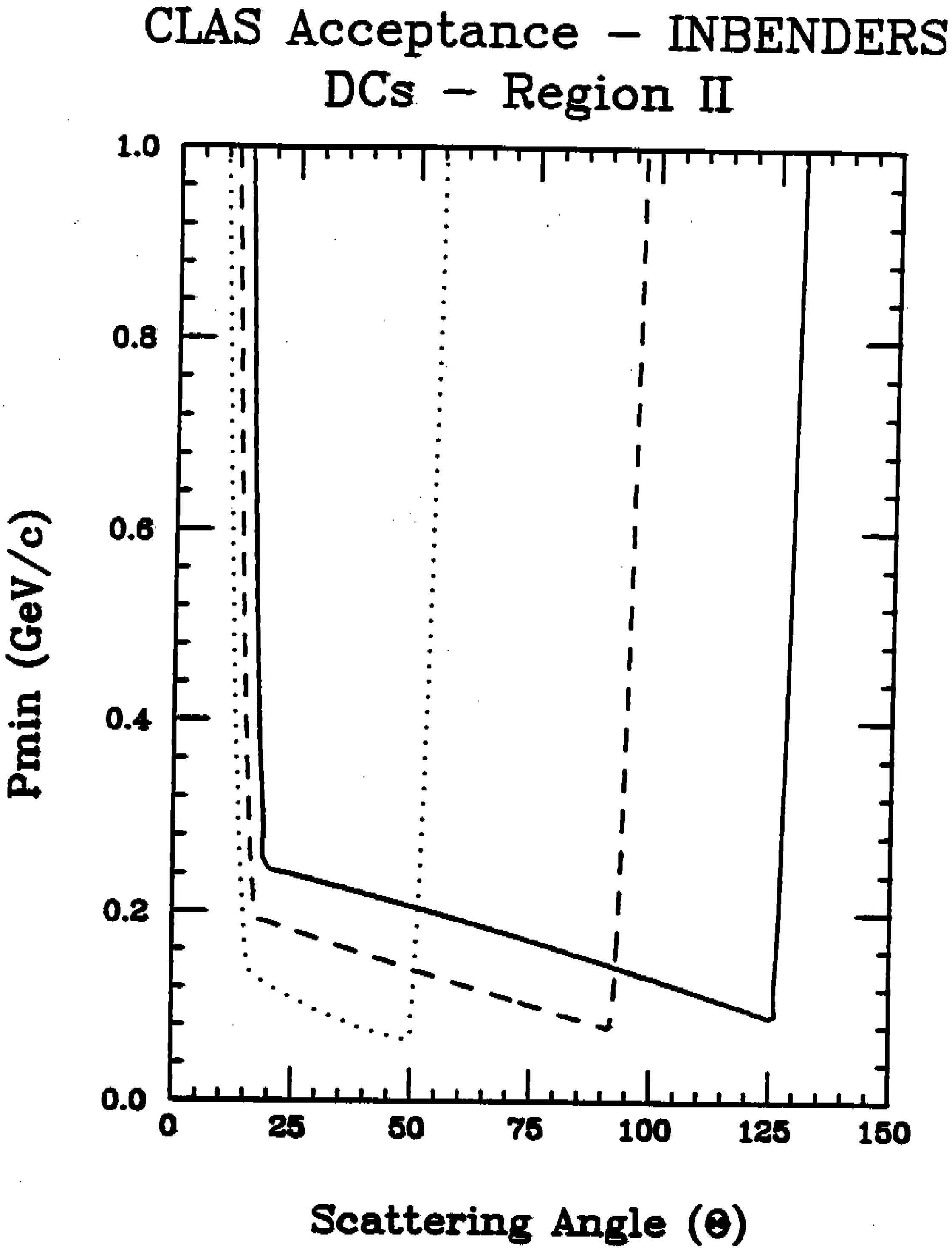

Figure 4: The parametrization of the minimum momentum required to reach the region II drift chambers for negative particles which bend toward the axis. The solid curve is for the target at its nominal position centered on the detector. The dashed and dotted curves show the acceptance for a target placed 100cm and 200cm upstream respectively.

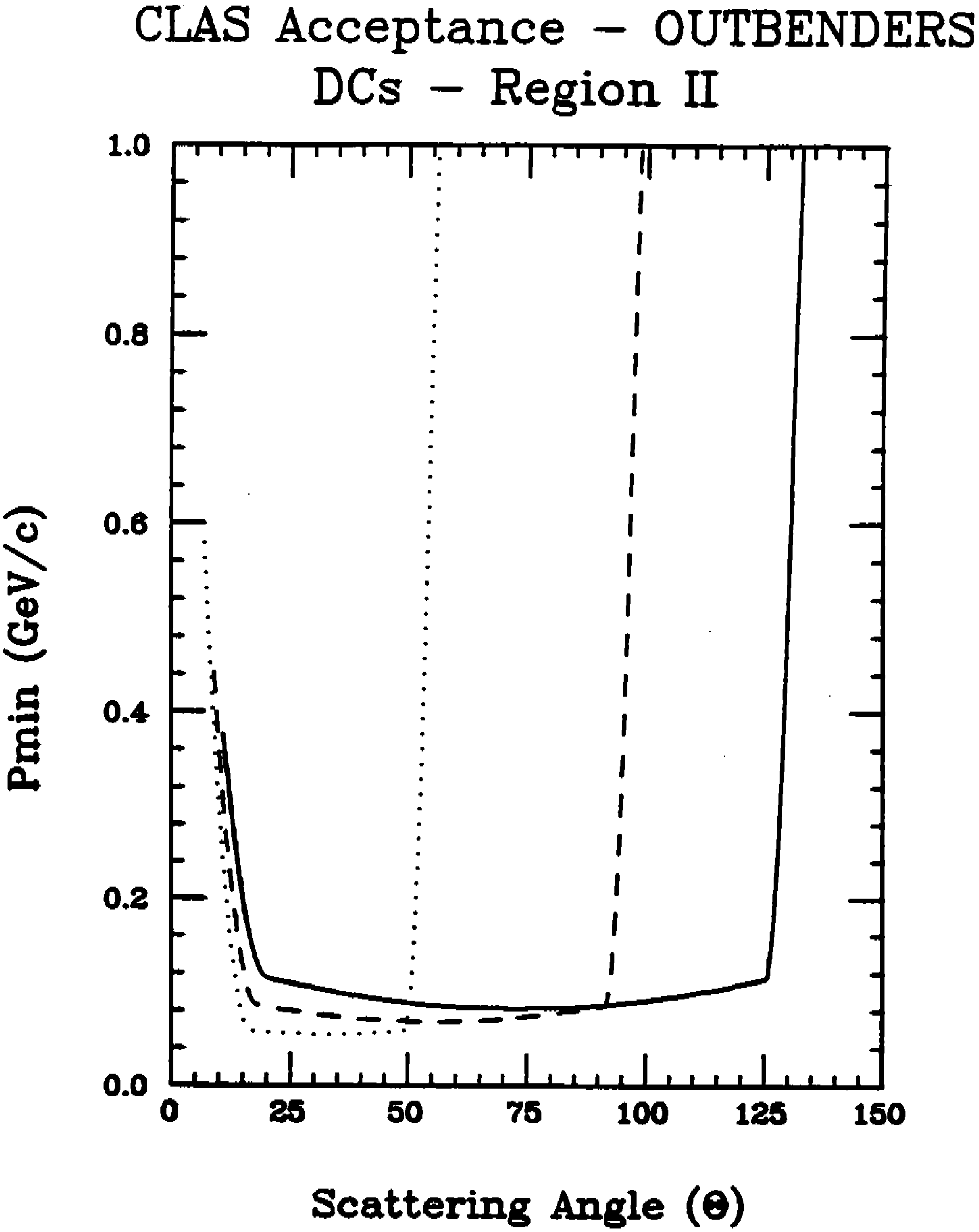

Figure 5: The parametrization of the minimum momentum required to reach the region II drift chambers for negative particles which bend away from the axis. The solid curve is for the target at its nominal position centered on the detector. The dashed and dotted curves show the acceptance for a target placed 100cm and 200cm upstream respectively.

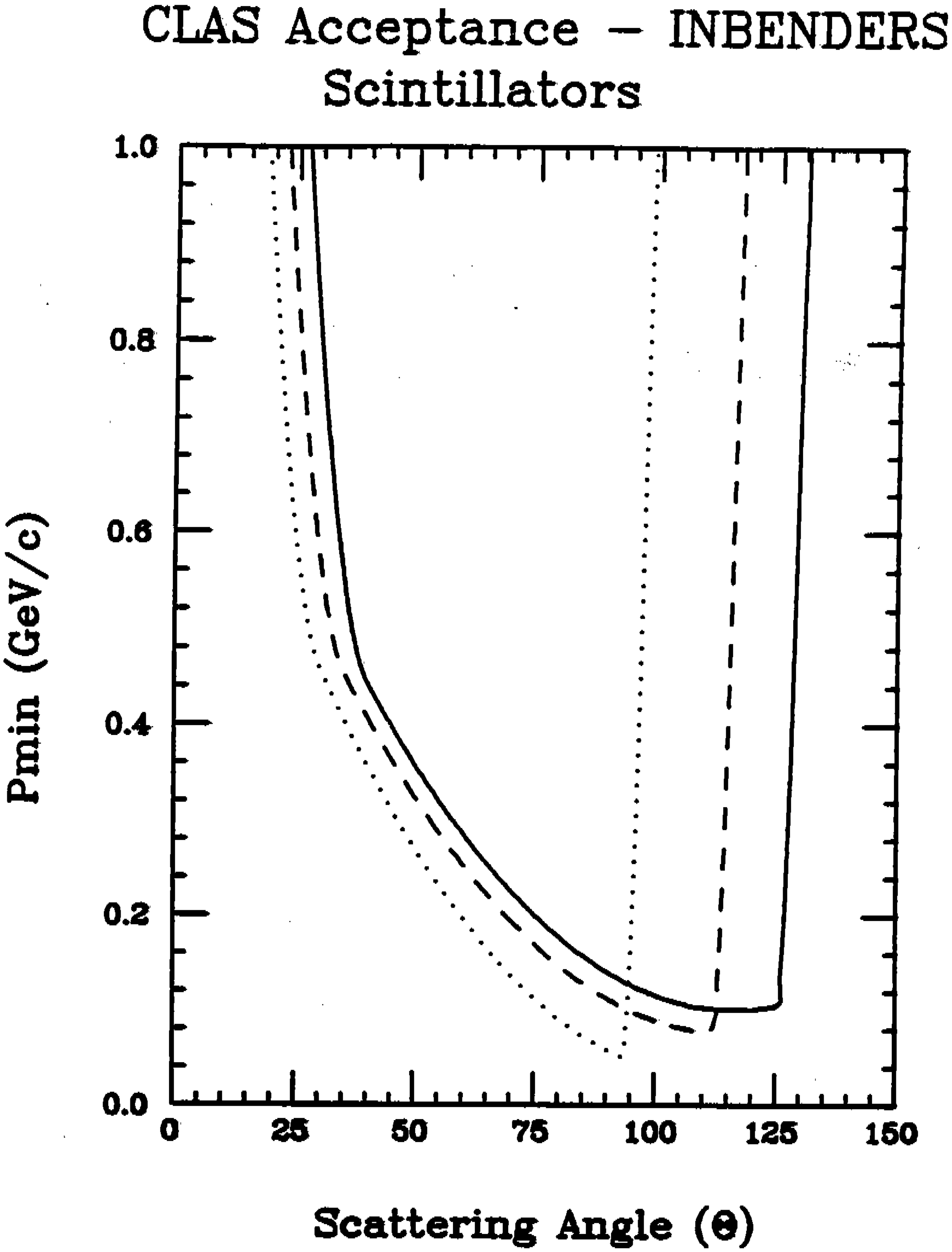

Figure 6: The parametrization of the minimum momentum required to reach the trigger counters for negative particles which bend toward the axis. The solid curve is for the target at its nominal position centered on the detector. The dashed and dotted curves show the acceptance for a target placed 100cm and 200cm upstream respectively.

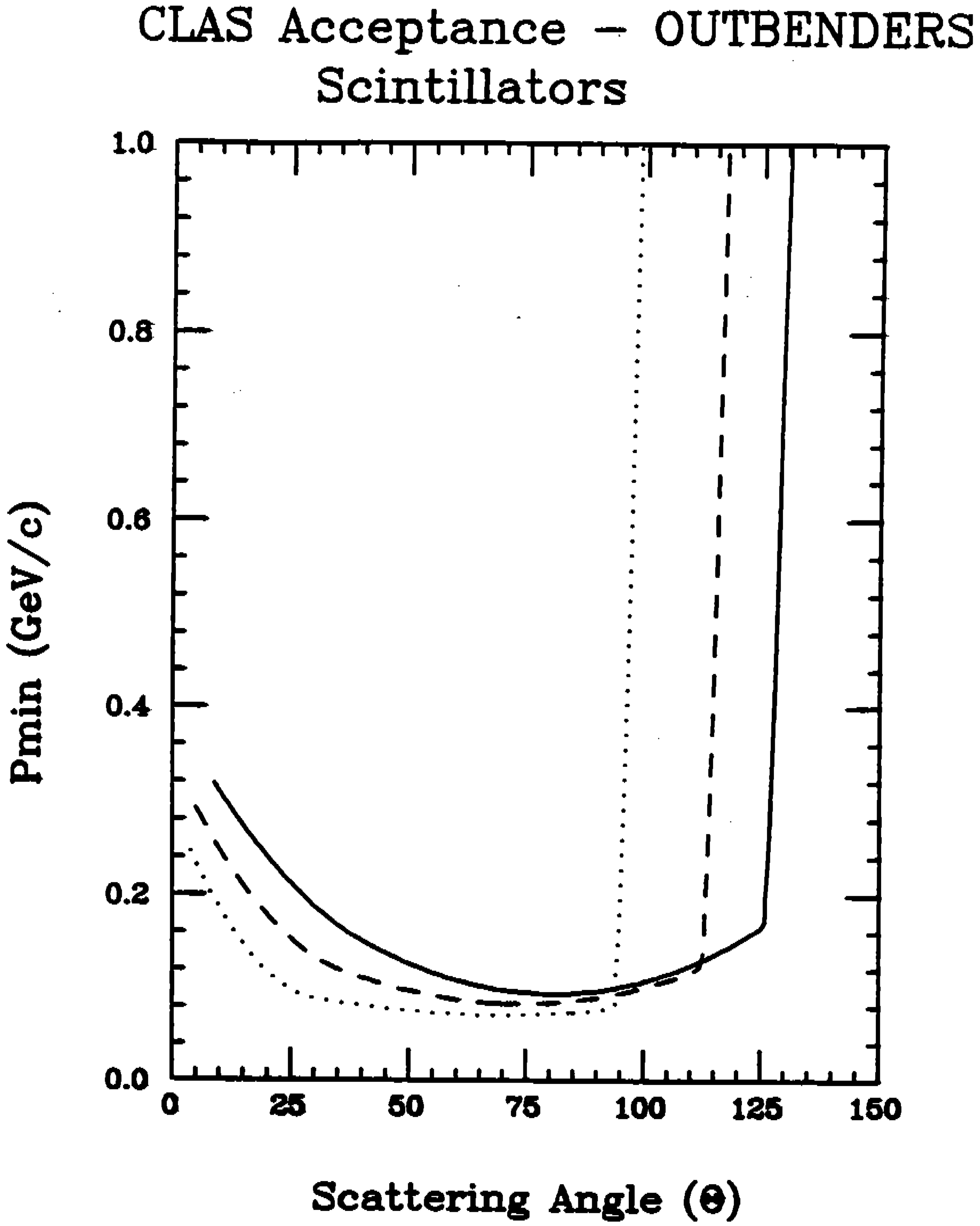

Figure 7: The parametrization of the minimum momentum required to reach the trigger counters for negative particles which bend away from the axis. The solid curve is for the target at its nominal position centered on the detector. The dashed and dotted curves show the acceptance for a target placed 100cm and 200cm upstream respectively.

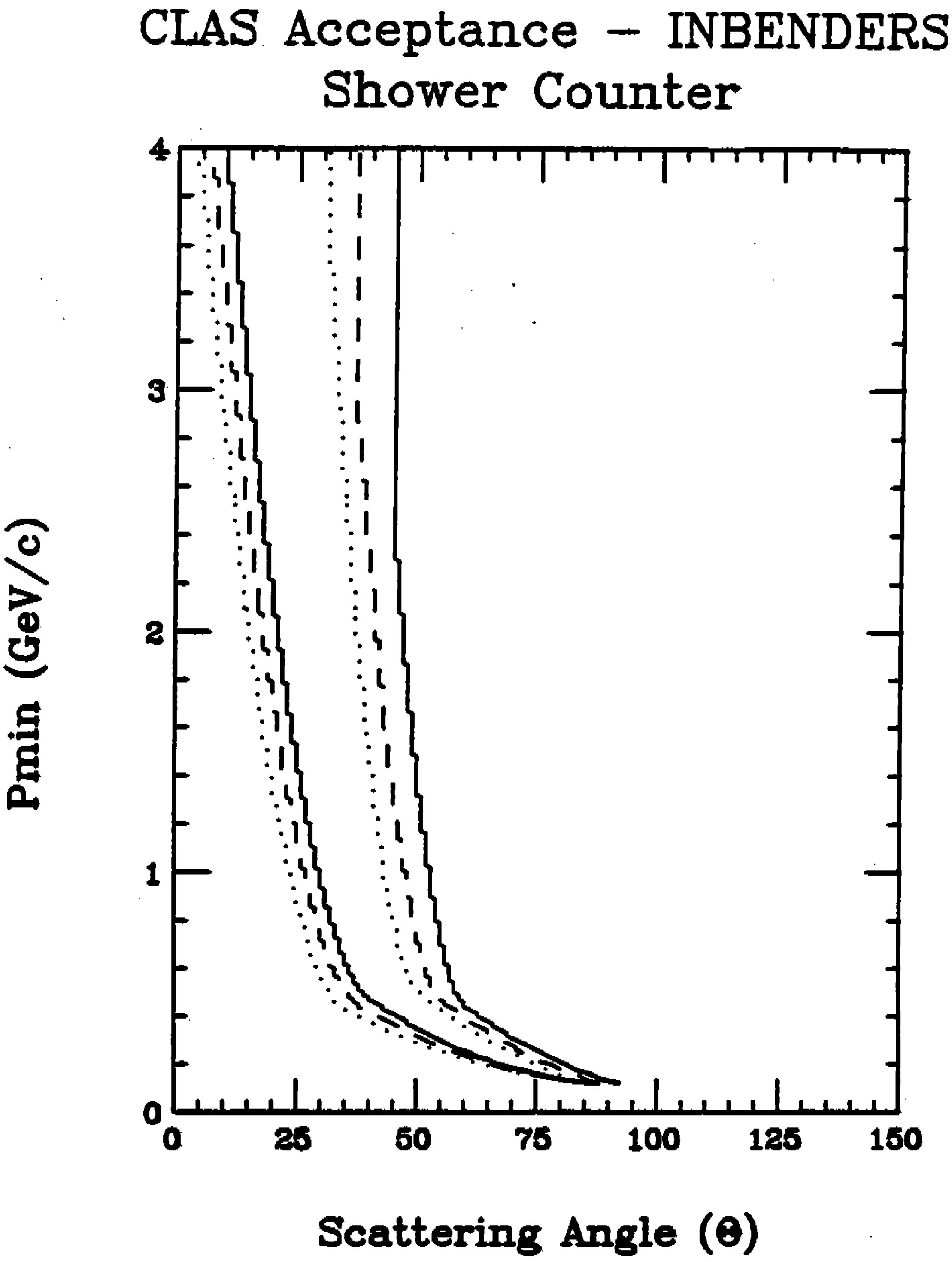

Figure 8: The parametrization of the minimum and maximum momentum required to reach the shower counters (45°coverage) for negative particles which bend toward the axis. The solid curve is for the target at its nominal position centered on the detector. The dashed and dotted curves show the acceptance for a target placed 100cm and 200cm upstream respectively. The discontinuities of the curves are an artifact of the plotting program.

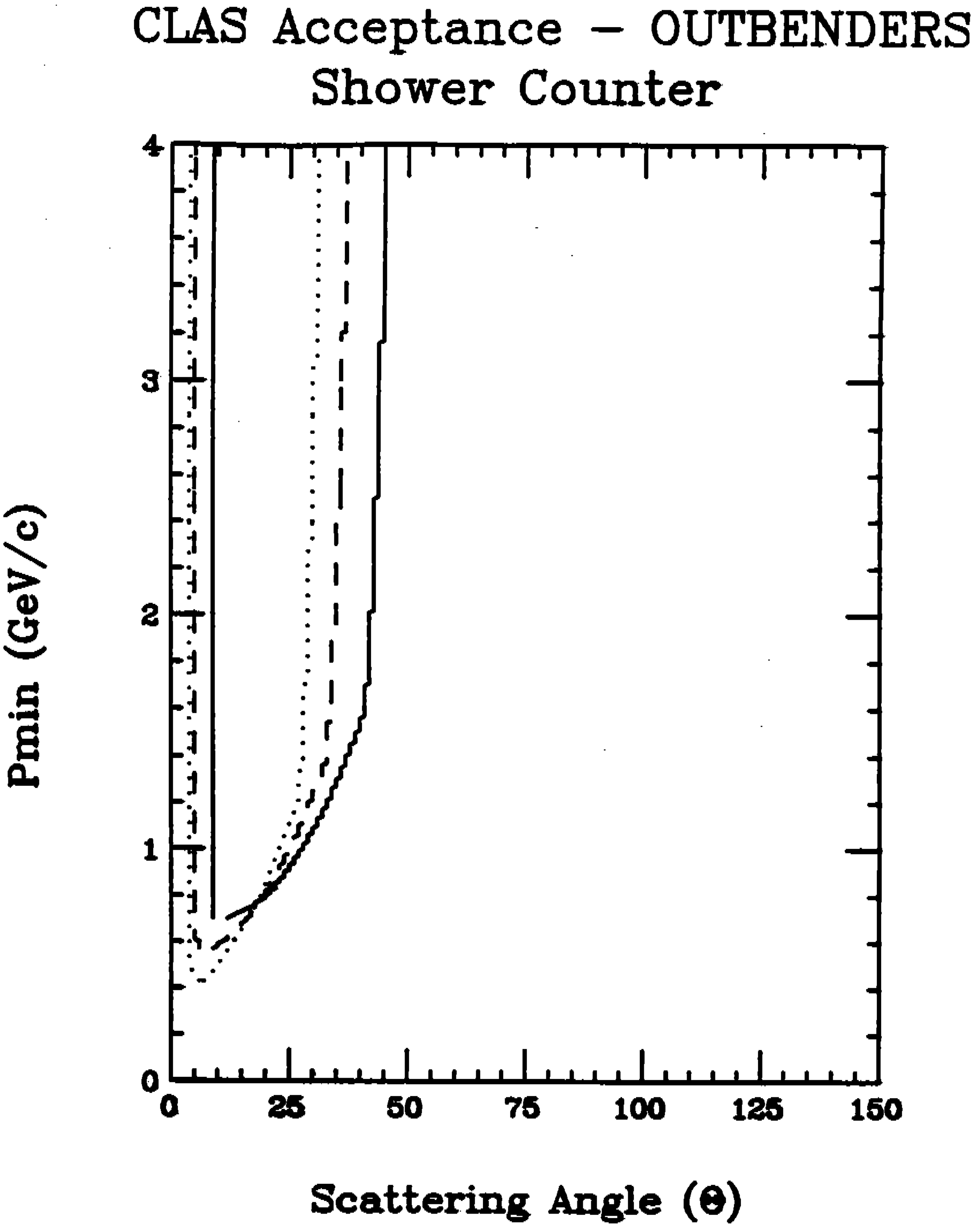

Figure 9: The parametrization of the minimum and maximum momentum required to reach the shower counters (45°coverage) for negative particles which bend away from the axis. The solid curve is for the target at its nominal position centered on the detector. The dashed and dotted curves show the acceptance for a target placed 100cm and 200cm upstream respectively. The discontinuities of the curves are an artifact of the plotting program.

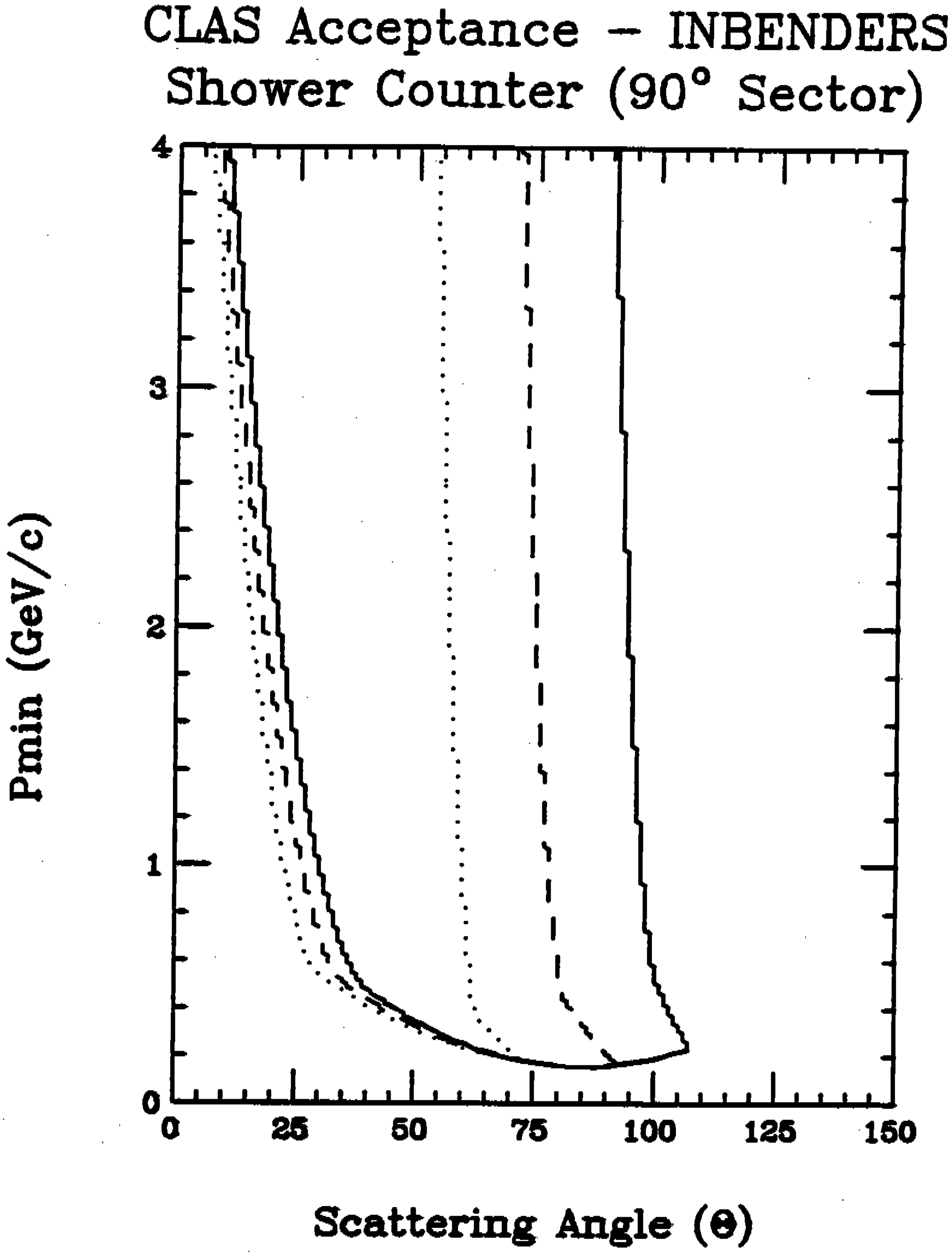

Figure 10: The parametrization of the minimum and maximum momentum required to reach the shower counters (90°coverage) for negative particles which bend toward the axis. The solid curve is for the target at its nominal position centered on the detector. The dashed and dotted curves show the acceptance for a target placed 100cm and 200cm upstream respectively. The discontinuities of the curves are an artifact of the plotting program.

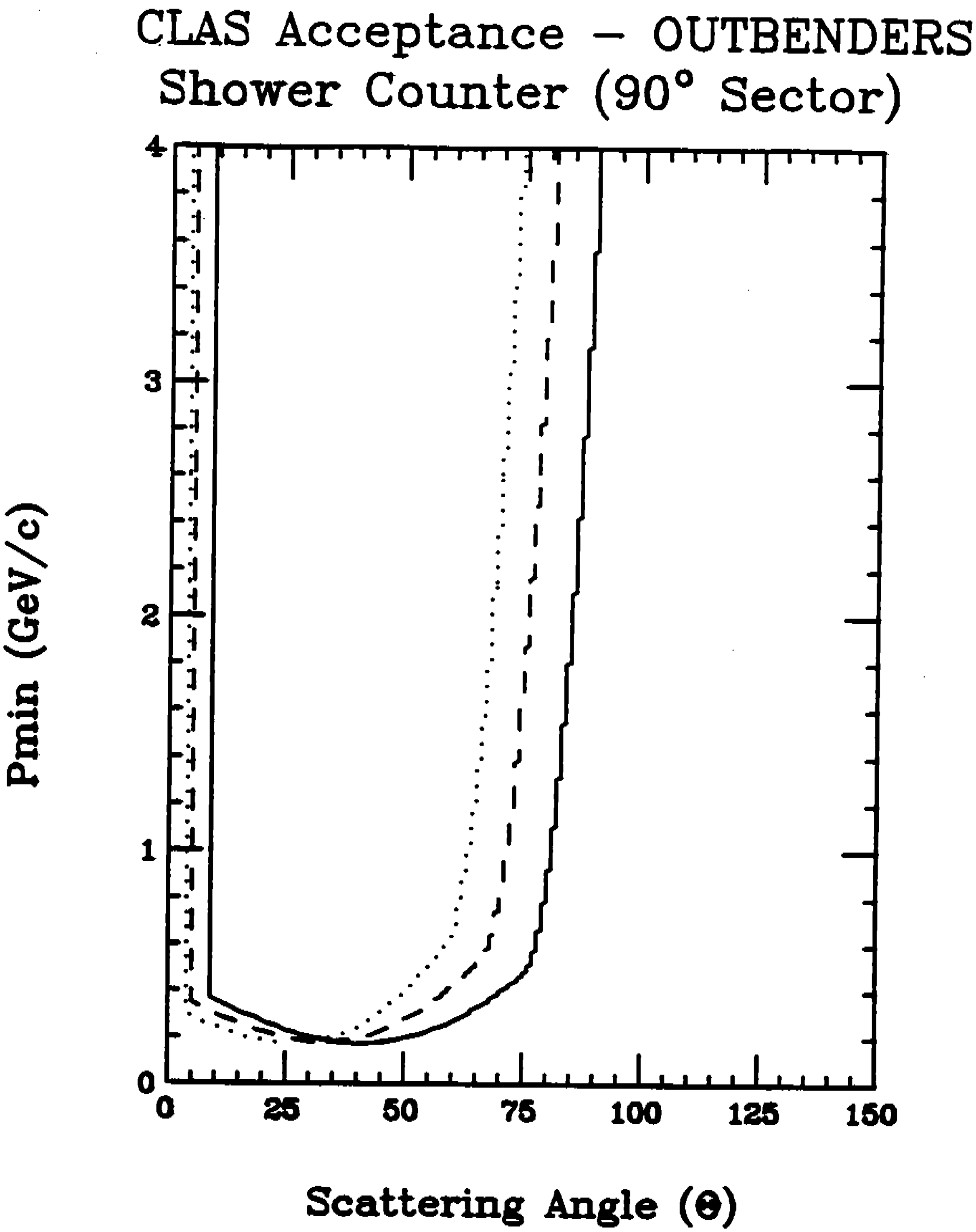

Figure 11: The parametrization of the minimum and maximum momentum required to reach the shower counters (90°coverage) for negative particles which bend away from the axis. The solid curve is for the target at its nominal position centered on the detector. The dashed and dotted curves show the acceptance for a target placed 100cm and 200cm upstream respectively. The discontinuities of the curves are an artifact of the plotting program.

CLAS Range/Time Requirements

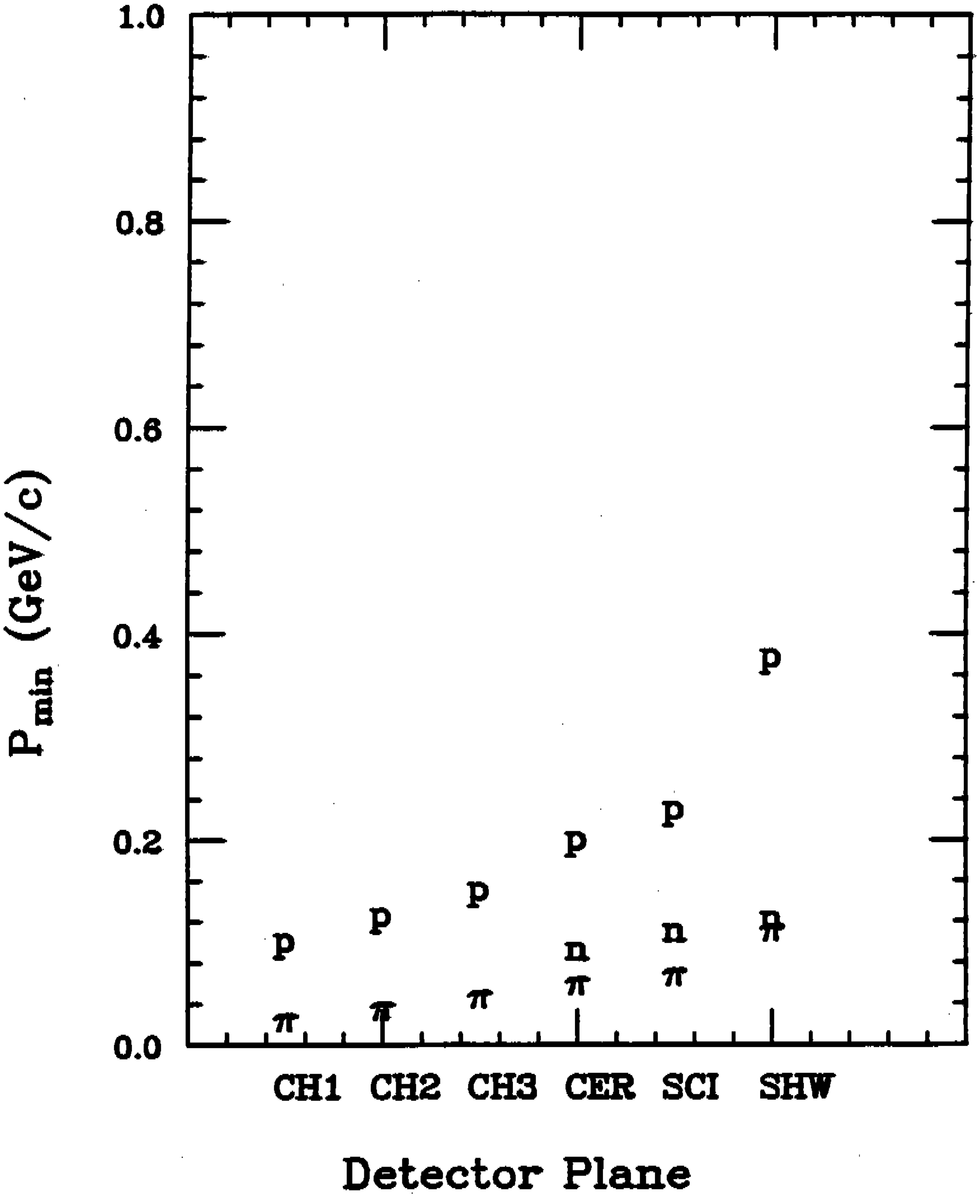

Figure 12: Particle range in terms of detector element reached versus momentum for protons, charged pions and neutrons.

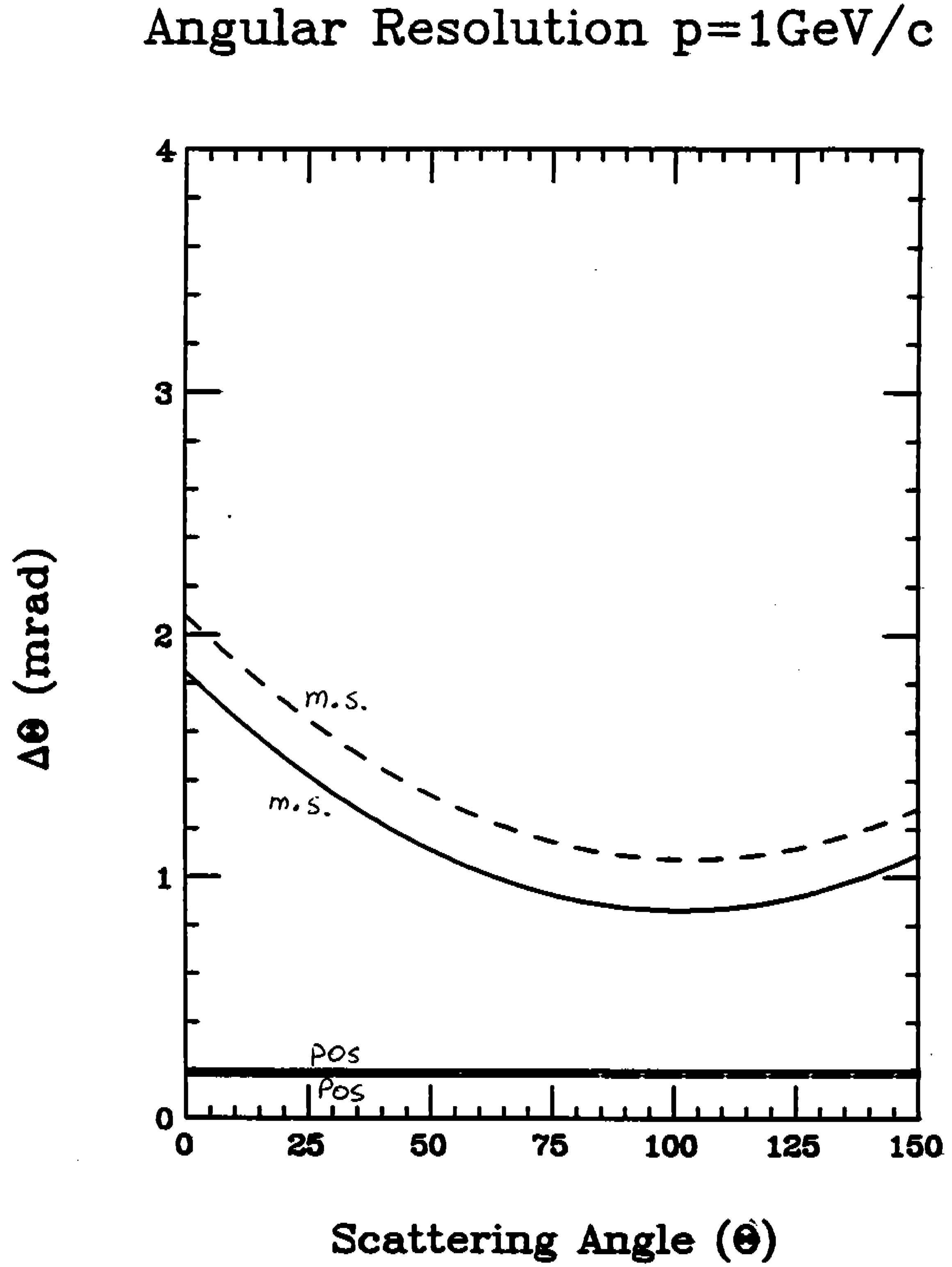

Figure 14: The parametrization of the angular resolution θ (FWHM) is shown for the case when the vertex is specified (solid) and for long targets (dashes). The contributions due to multiple scattering and position resolution are shown separately.

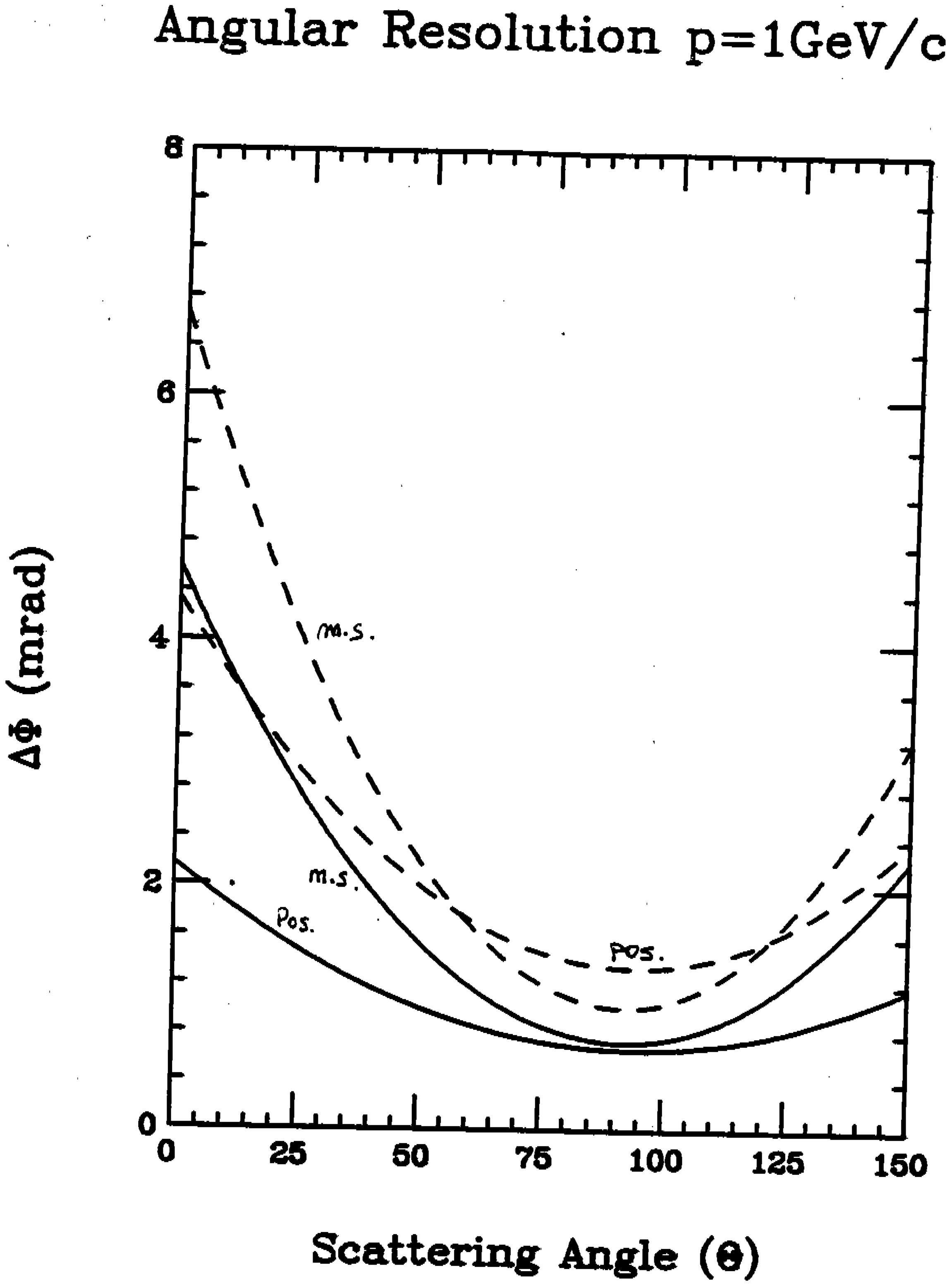

Figure 15: The parametrization of the resolution of the azimuthal angle ϕ (FWHM) is shown for the case when the vertex is specified (solid) and for long targets (dashes). The contributions due to multiple scattering and position resolution are shown separately.

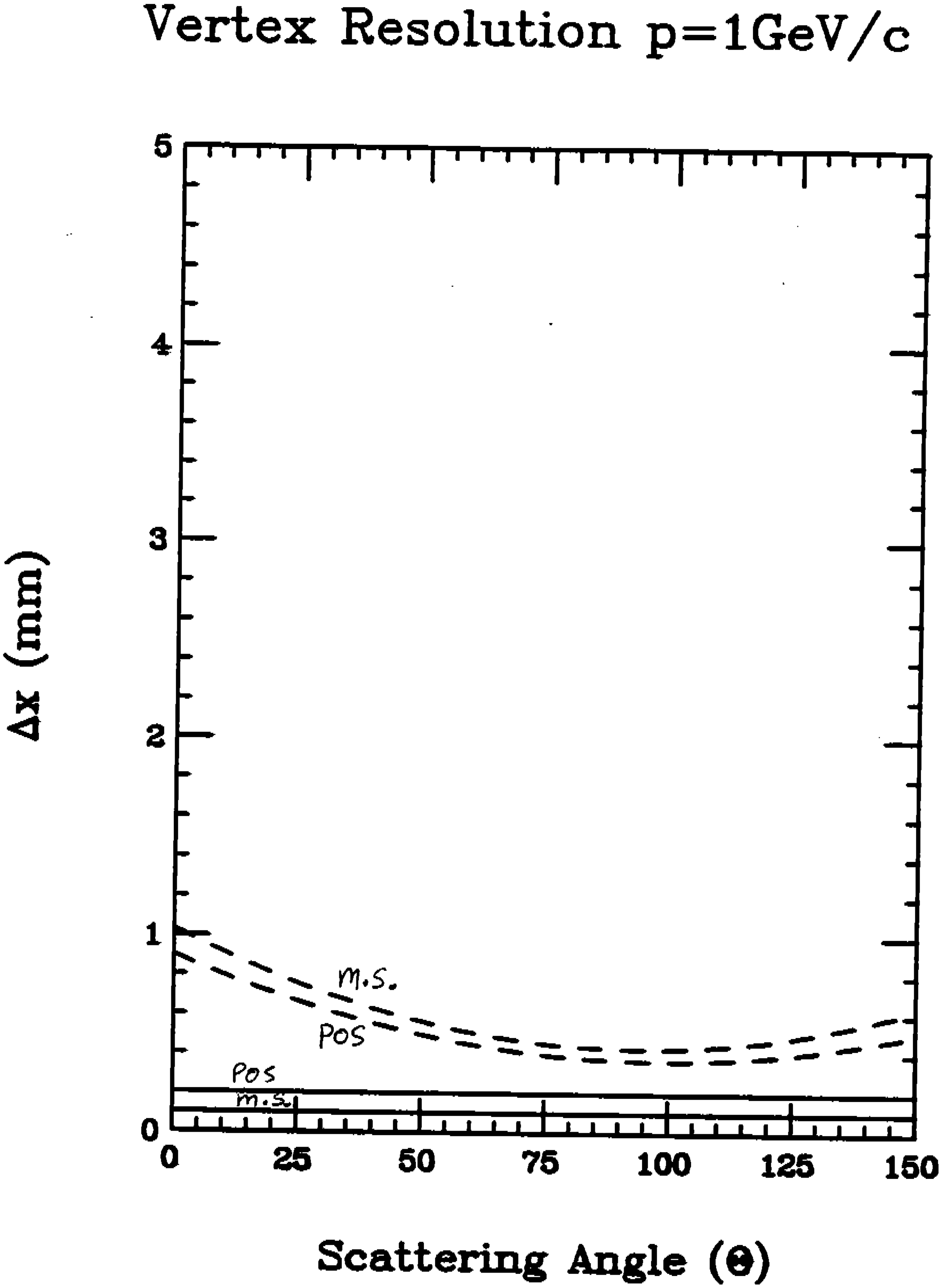

Figure 16: The parametrization of the vertex resolution (FWHM) is shown for long targets (dashes). The contributions due to multiple scattering and position resolution are shown separately.

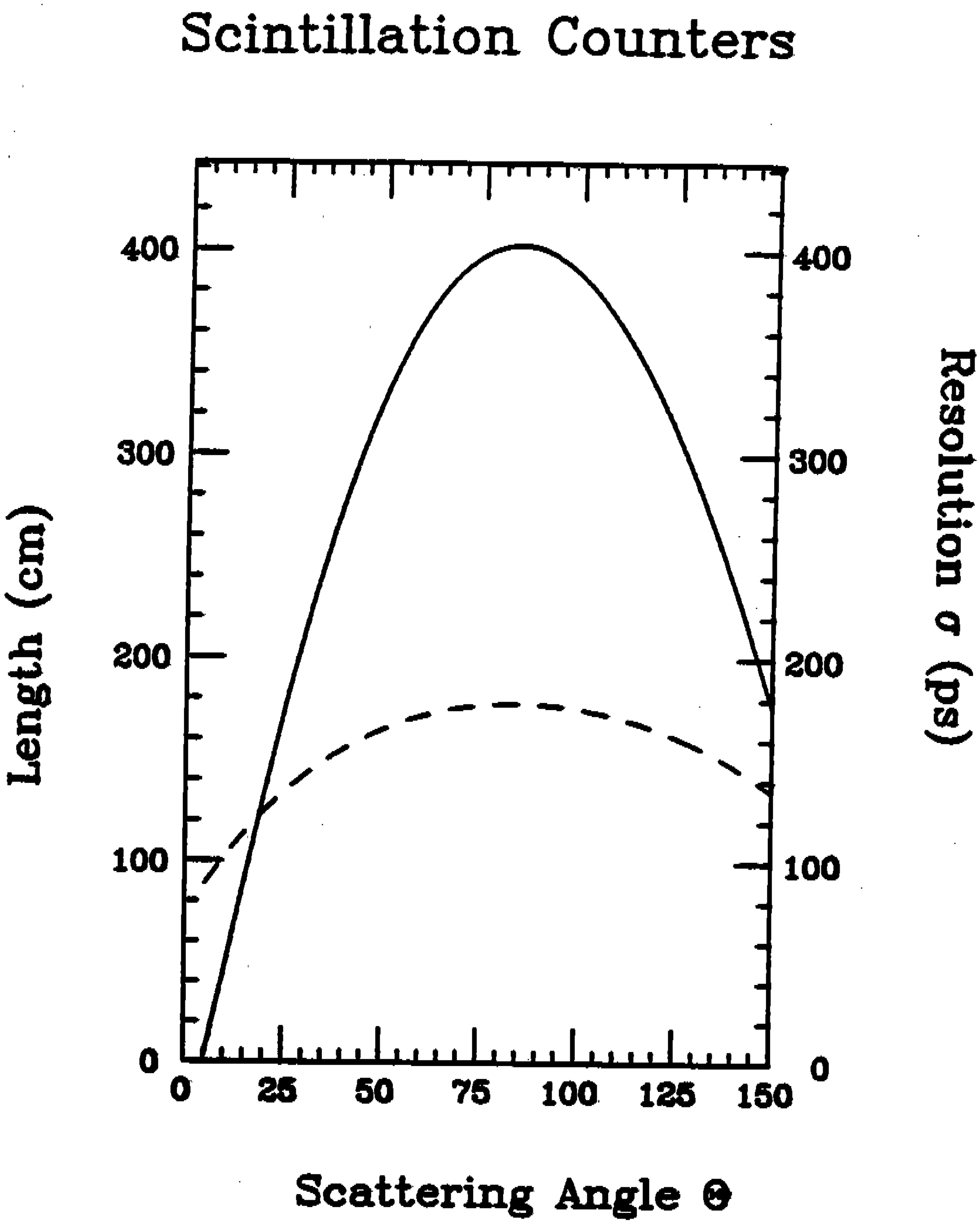

Figure 17: The parametrization of the timing resolution (r.m.s.) is shown on the right scale (dotted curve). The length of the scintillator counters is given by the scale on the left (solid curve).

has been parameterized and may be selected using the 'vertex' option in the input file (see Section 3.1).

In cases when low-energy particles are used in the analysis which do not have hits in all drift chambers, the momentum and angular resolution will be degraded. Thus, the estimated resolution for each of these quantities is computed depending on whether a track will reach region I, II or III drift chambers. The lack of hits in region III will generally worsen momentum resolution, but leave angular resolution unchanged. However, pattern recognition will be more difficult. Thus our estimated resolutions for cases with limited information are on the conservative side. Even so, this information should be used with care and consideration of the particular experiment under study.

Energy loss and rescattering in the target contribute to the momentum and angular resolution. For many experiments, which use thin targets, these effects may be neglected. However, tagged-photon beam experiments are often limited by the photon flux and therefore require relatively thick targets. In addition, the photon beam has a extended profile. We have therefore added input parameters (see Section 3.1) to specify the characteristics of the target and surrounding material. The multiple scattering in the target is added in quadrature to the angular distributions obtained by fitting tracks in the magnetic field. The energy loss in the target varies depending on the actual vertex location. The momentum resolution is thereby decreased when a thick target is used with a relatively wide beam. Thin targets may also contribute to the resolution at 90^o when particles exit transverse to the beam direction.

The timing resolution obtained from the time-of-flight counters is parameterized as a function of the length of the counter as shown in Figure 17.

2.3 Detector Response

For some applications, one needs to know not only whether a track hit a given detector (e.g. shower counter), but also where the counter was hit and what the pulse height was. This information is required, for example, in simulating the trigger. For example, a cut on visible energy in the shower counter will determine the efficiency for tagging electrons compared to accepting pions. The shower counter response to electrons, pions, protons and neutrons is shown in Figure 18. The response in the Cerenkov counter is 1 if a charged particle has velocity which is greater than .9989212c, which corre-

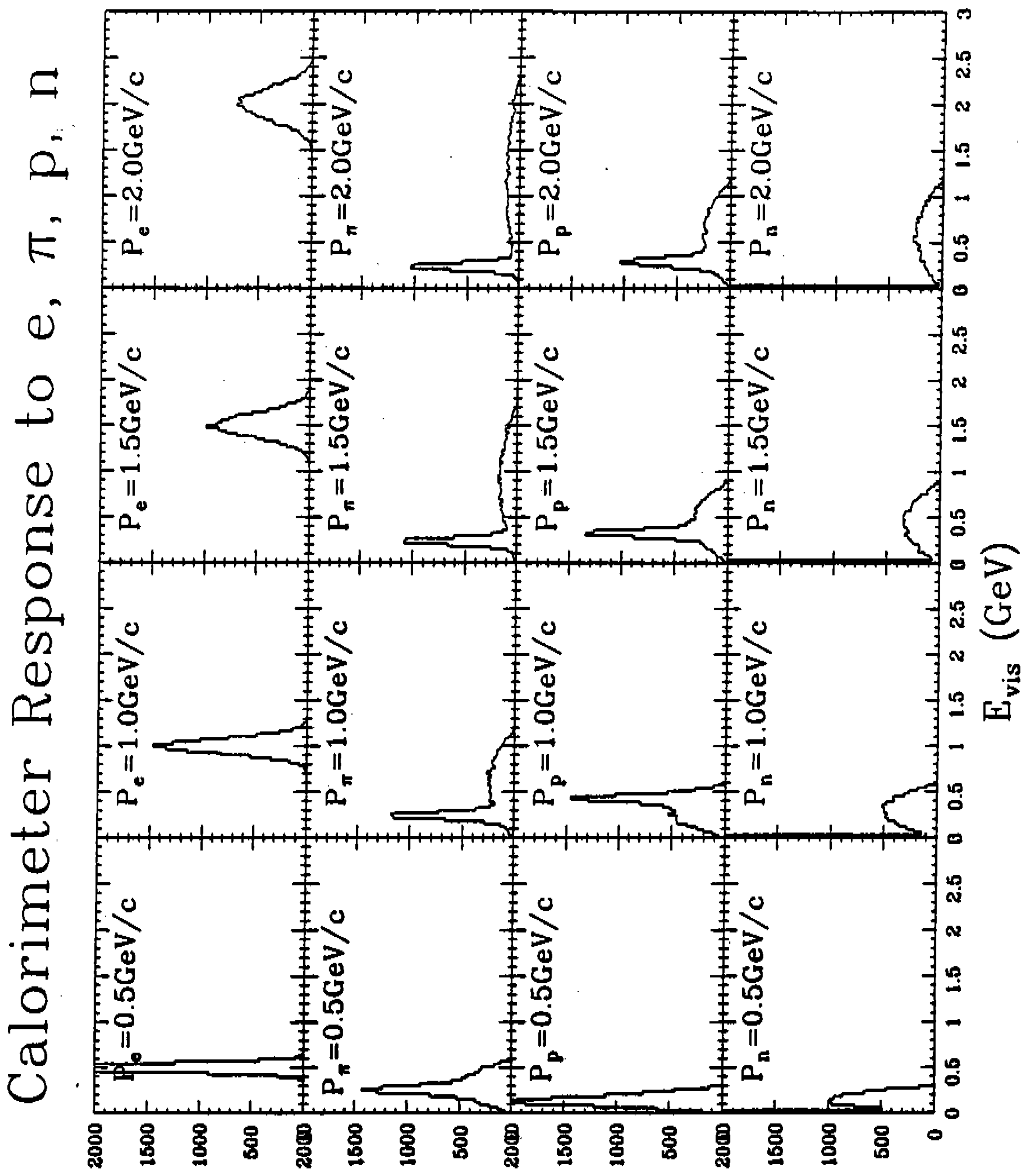

Figure 18: Visible energy in the shower counter deposited by electrons, pions, protons and neutrons of fixed momenta.

sponds to the critical velocity for freon, and 0 otherwise. However, pions will cause a "hit" in the Cerenkov counters .25% of the time, representing a typical conversion probability in the counter walls. All charged tracks produce hits in the scintillators. Neutrons produce a hit 5% of the time and photons produce a hit 10% of the time. We presently assume that charged particles are detected in the drift chambers with 100% efficiency. However, for particles which might decay, the returned efficiency is equal to their survival probability.

To determine where a counter was hit by a charged track, the bending in the magnetic field is approximated in first order by

$$\Delta \ = \ .075 \int B dl / p. \tag{4}$$

Δ is the total the bending angle due to the magnetic field p is the particle momentum. $\int Bdl$ is the total integral magnetic field at a given scattering angle. This approximation is quite adequate at large angles. In the forward direction the derivative of the field is so large that precise estimates of the bending require a much more sophisticated treatment of the problem.

For the Cerenkov, scintillators and shower counters, three variables are filled which specify the detector response and which counter was hit (the sector and module numbers). The variables are

hit_cer - hit in Cerenkov counter (0 or 1)
hit_sci - hit in scintillator counter (0 or 1)
hit_shw - hit in shower counter (visible energy in GeV)
isec_cer - Cerenkov counter sector (1-6)
isec_sci - scintillator counter sector (1-6)
isec_shw - shower counter sector (1-6)
imod_cer - Cerenkov counter module (1-5)
imod_sci - scintillator counter module (1-48)
imod_shw - shower counter module (1)

The sector and module numbers are zero if there is no acceptance for the track in question. The hit values represent the response of a particular detector to a given particle. Thus these variables are filled without regard to actual acceptance.

3 How to use FAST

This a brief guide on how to use the program for one's own application. We first mention a few conventions which are used throughout. The file extension .COM is used for the command procedure to LINK a program and .BAT is used for a command procedure to execute that same program. For example, FAST.COM links the Fast Monte Carlo and FAST.BAT will run the program. The FORTRAN code resides in the subdirectory FASTMC$SRC, one file for each subprogram. Files with common block specifications are given the extension .INC and reside in the subdirectory FASTMC$INC. The program is coded in VAX FORTRAN, where exceptions to standard FORTRAN were made when generality imposed severe constraints on the application. FAST uses default variable types so that portions of the program may be examined in isolation and correctly interpreted. The directory FASTMC$DEMO contains command procedures to run all the examples, detailed below, as well as some introductory notes contained in the file README.DOC.

The first step is to specify the physical process(es) one wishes to study. FAST is based on the general philosophy that the physics Monte Carlo programs should be separate from detector simulation. The physics input to FAST is taken from a file which specifies all the tracks which are associated with a given event. For compatibility, the format of the file is the same as that output by CELEG [7].

The procedure for using the FAST program is easy. The two routines which are most likely to be changed are FAST_HSTDEF and FAST_ANAL, which are used to make histogram definitions and analyze the event respectively. In the distributed version these are dummy subroutines. The examples provided in FAST_ANAL1–6 and the corresponding subroutines FAST_HSTDEF1–6 show how to access information. Once FAST_ANAL has been written, the new analysis program may be linked with FAST.COM and executed with FAST.BAT. This program uses the HBOOK histograming package, [2] a specialized library SPLINELIB and the CERN libraries GENLIB, PACKLIB and KERNLIB. Kinematical information is stored in LUND [11] common blocks (see Appendix A).

---

[2] The version of FAST as of February 1990 uses the new CERN package of routines from December 1989. To use the previous HBOOK format simply edit the files FAST_INIT.FOR and FAST_END.FOR by searching for lines which refer to "OLD HBOOK."

3.1 Input File

The file with the logical name Fast_list, which controls the processing within the program, is usually read directly from the batch command procedure, i.e. SYS$INPUT. See Table 1 for an example of the input listed within the file FAST1.BAT. Each of the parameters defined below correspond to one line of the input file:

- **Title.** Title used to identify the particular run of interest. It is also used for the HBOOK global title together with the date.
- **Event Limit.** Maximum number of events to process if less than the number of events on the input file.
- **Magnet polarity.** +1 bends negative particles toward the axis, −1 bends negative particles away from the axis.
- **Vertex Definition.** The determination of momentum and angles may be improved with precise knowledge of the position of the interaction vertex. If this character variable is 'vertex' the improved resolution functions are used. If this variable is 'novertex' momentum and angle measurements are obtained from the information in the drift chambers alone.
- **Resolution.** To study the effect of the contributions to momentum resolution, this character variable may be set to 'pos', 'mul' or 'both' to include the effects due to position uncertainties only, multiple scattering only or the quadratic sum of both. This option only has an effect for particles which traverse all three drift chamber regions.
- **List.** For debugging purposes, this character string will output the momenta of each track read from the input file if it is set to 'list'.
- **Analysis Routine.** This option allows any of the six example analysis routines to be run ('example1',...,'example6') without relinking the program. Set to any other string, the user written FAST_ANAL analysis routine is called.
- **Data Input and Output Files.** We have added the capability of writing an output file with the FAST Monte Carlo. The format of the

Table 1: DCL Command procedure FAST1.BAT which tests resolution. The actual example runs three times to illustrate the effects of reducing the magnetic field and contribution of the target to overall resolution.

```
$!
$!              Procedure to run CLAS Fast Monte Carlo Program
$!
$!                This procedure is setup to execute Example #1.
$!
 $ define       Fast_input   SYS$INPUT
 $ define       Fast_lst     FAST1.LIS
 $ define       Fast_hbk     FAST1.HBK
 $ define       LUND         GEN_E1FIXED.DAT
 $ define       Trigspec     Trigger_delta_1prong.dat
$!
$ run fastmc:Fast
’Example 1: Resolution - Full Magnetic Field’ /Title
10000             /event limit
-1                /magnet polarity (+1 = neg bend toward axis, -1 neg away)
’vertex’          /’vertex’ defined, ’novertex’ for long targets
’both’            /’pos’,’mul’,’both’ - pos or mult scat terms only, or both
’nolist’          /’list’ or ’notlist’ print events read from file
’example1’        /’example1’,’example2’,’example3’,’other’ analysis options
F                 /T = Input File is in ”Fast Data” format (F=LUND Format)
F                 /T = Write Output file in ”Fast Data” format (F=no output File)
1.                /Magnetic field strength relative to nominal
0.                /Target position (cm) (positive is downstream)
1.,.012,2.3,19.   /Target radius(cm), length(cm), density(g/cm3), rad length(cm)
’other’           /’hydrogen’ or ’other’ treats hydrogen special in dE/dx
.05,2.,21.1       /Container thickness(cm), density(g/cm3), rad length(cm)
.01               /RMS beam spread (cm)
$ eod
$ exit
```

file is referred to as the "Fast Data" format and includes all the information contained in the CELEG input file as well as trigger information and calculated variables stored in the Common /fsmear/ (specified in FAST_COMMON.INC). To write an output file one must define the logical variable FAST_DATOUT and set the logical switch in the ninth line of SYS$INPUT to T (see Table 1). A file in this same format may be read by the program by defining the logical variable FAST_DATIN and setting the switch in the eight line of SYS$INPUT to T (see Table 1). When FAST reads a file in this new format, preprocessing of the event is bypassed as all the relevant information is read from the file. Note that in this case the input specifications of polarity, vertex definition and resolution terms are ignored and taken from FAST_DATIN.

- **Target Position.** The position (cm) of the target along the beam axis may be specified with this option. A negative value moves the target position upstream and may be used to increase the acceptance at small angles. All acceptance functions and geometric quantities are computed as a function of this parameter. However, the dependence of resolution on target position is expected to be small and is presently ignored.

- **Target Parameters.** This line specifies the parameters of the target which is assumed to have a cylindrical shape. The parameters specified are radius (cm), length (cm), density (g/cm^3) and radiation length (cm).

- **Hydrogen Option.** If the target is hydrogen this character string should be set to 'hydrogen'. This will account for the unusually large energy loss in a hydrogen target relative to other materials.

- **Target Container.** The material surrounding the target is to be included in these effective parameters. The geometry assumed for this material is a cylindrical shell enclosing the target. The materials in the beam pipe required for shielding, given in the input file in Table 1, usually dominate. The parameters are thickness (cm) density (g/cm^3) and radiation length (cm).

- **Beam Spread.** The energy loss and multiple scattering in the target are averaged over a Gaussian beam profile with an rms width given by this parameter in cm.

3.2 Examples

The generation of events must be handled by a separate physics Monte Carlo such as CELEG. By way of example, a driver routine for the FAST Monte Carlo has been written which reads a file in the format specified by CELEG. Since one may quickly learn from example, six analysis subroutines have been written and are distributed along with the FAST Monte Carlo for the purpose of testing and demonstration. Each example may be run several times with different inputs to illustrate or test a particular aspect of the program. The command procedure which is used to run example 1 is given in Table 1. Options to the program are controlled by the input file which in the example is SYS$INPUT and is listed directly in the command procedure. The options are generally self-explanatory and define various running conditions as detailed above.

3.2.1 Example 1: Resolution

This example is used to test the parameterizations of the resolution functions. As presently configured, the program reads a file of generated events. The file GEN_E1FIXED.DAT contains 1000 events with p=1GeV, $\theta=20^\circ$and $\phi=0^\circ$and 1000 events with p=2GeV, $\theta=100^\circ$and $\phi=20^\circ$. The program counts the number of events which hit two super layers (2000) and the number that hit the trigger counters (1000). Using the histogramming package HBOOK, the momentum and angles of each of the smeared tracks is plotted and the r.m.s. resolution can be read directly from the plot statistics. The list file FAST1.LIS can be sent to the printer. The histograms are also stored for convenience on the disk file FAST1.HBK. This file may be inspected with the Physics Analysis Workstation PAW [8].

3.2.2 Example 2: Acceptance

The second example is a test of the acceptance functions used by the program. Plots of momentum versus angle of the accepted events are generated, which directly show the region of the detector acceptance. The program is run

Table 2: A comparison of the acceptance obtained with the FAST Monte Carlo and a GEANT simulation of the reaction $ep \rightarrow e'N^*(1535)$. The event sample was generated for 4GeV/c incident electrons. The scattering angles of the electron and proton were required to be greater than 20°.

Coincidence	*FAST acceptance*	*GEANT acceptance*
e only	.607	.605
e and p	.311	.315

several times, for full and half magnetic field intensity and a target position of 2m upstream of detector center.

3.2.3 Example 3: Missing Mass

The third example uses the same file of the reaction $ep \rightarrow e'N^*(1535)$ which was used with GEANT to study the missing mass reconstruction of CLAS (TAP2 [9] report pp. M39-M42). After imposing roughly the same cuts as the previous analysis we histogramed the same variables with good agreement (see Figure 19). No attempt was made to tune any of the FAST Monte Carlo parameters. The comparison of the acceptance of the two procedures is shown in Table 2. Please note that this comparison was made for an outdated detector configuration. The missing mass resolution has been improved due to more precise measurements transverse to the bend plane.

3.2.4 Example 4: Particle Identification

This example takes as input a file containing 4000 pions, 4000 kaons and 4000 protons all generated with uniform momentum between 0 and 4 GeV/c. Half of the particles were generated at θ=20°, the rest at 90° and all in the mid plane between the coils (ϕ=0°). The program then histograms (see Figure 20) the flight time between the target and the scintillators as a function of momentum for various acceptance criteria. This indicates the level of particle identification possible by means of the time-of-flight technique. We note that we have only included first order corrections to the path length due to

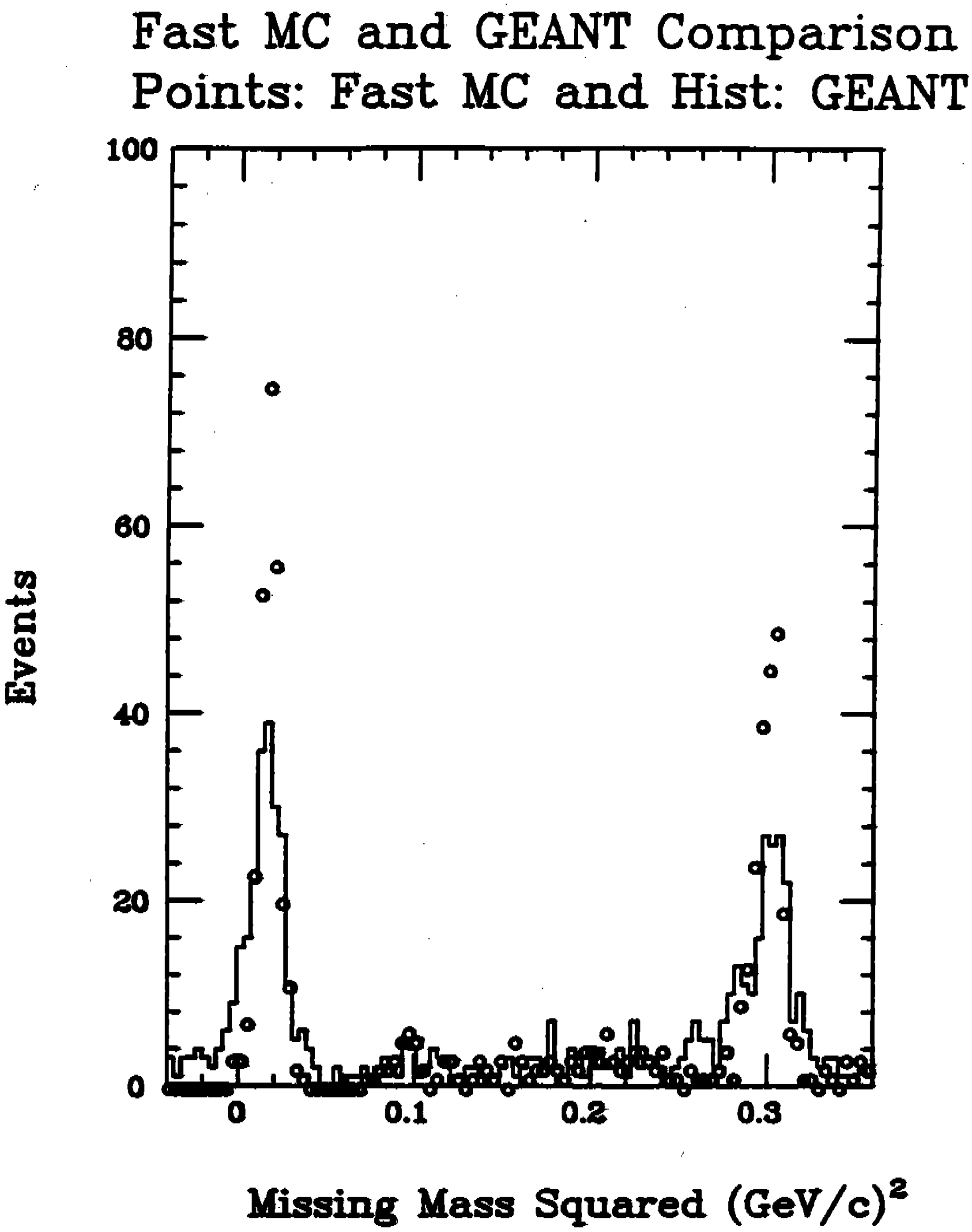

Figure 19: Comparison of the FAST Monte Carlo with GEANT simulation for the missing mass reconstruction for the reaction $ep \rightarrow e'N^*(1535)$. Please note that this comparison was made for an outdated detector configuration. The missing mass resolution has been improved due to more precise measurements transverse to the bend plane.

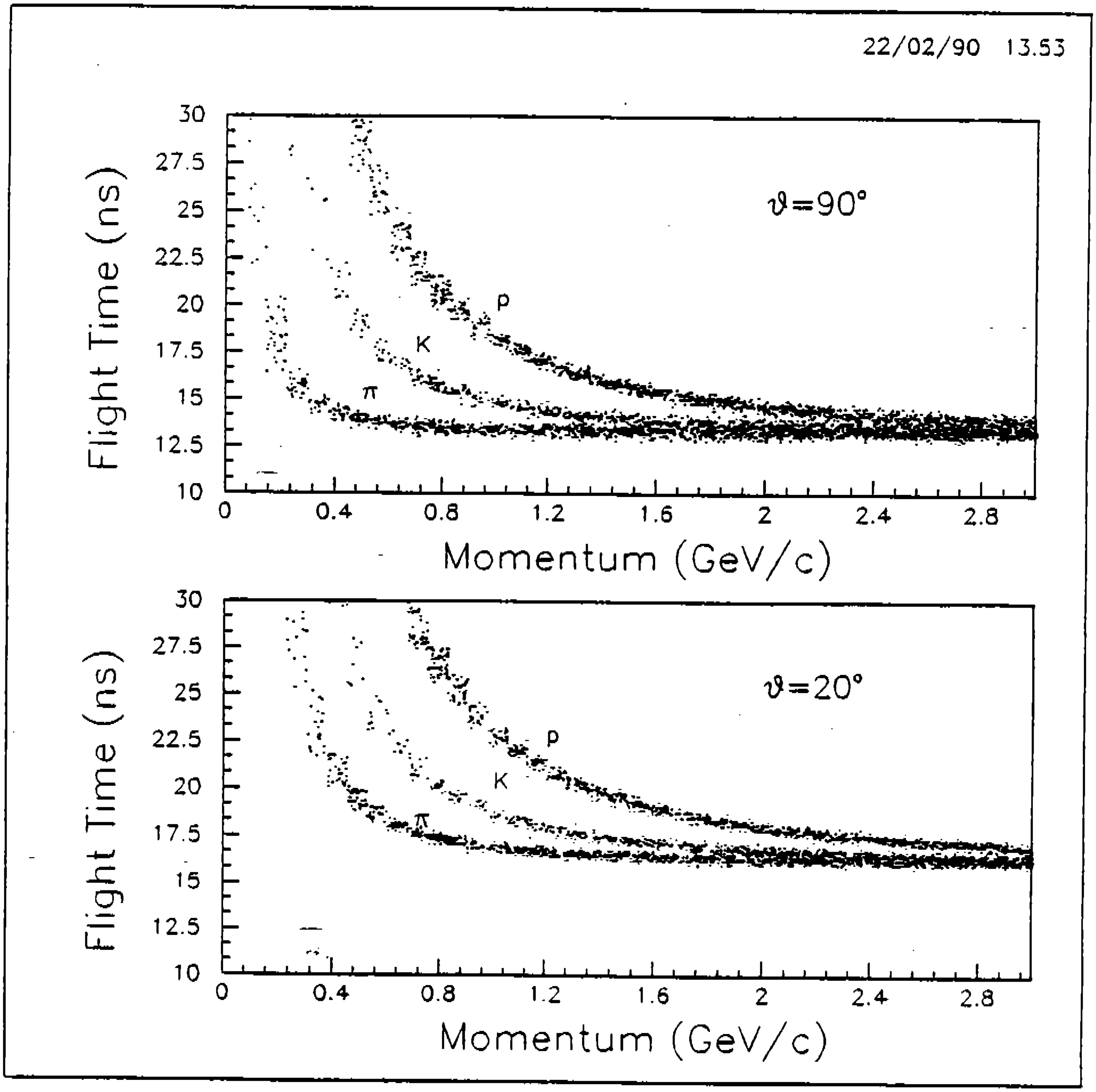

Figure 20: The flight time is plotted versus momentum for pions, kaons and protons at θ=20° and θ=90°.

bending in the magnetic field. However, at forward angles the variation of the magnetic field with angle is so large that further corrections need to be made to accurately determine the path length. Nevertheless, these approximations are adequate to study particle identification since the corrections are negligible at high energies where particle identification is difficult. Finally, we also histogram the kaon momentum with various criteria which shows the loss of kaons due to acceptance and decay.

3.2.5 Example 5: Hardware Trigger Simulation

The simulation of trigger levels 1,2 and 3 within the FAST Monte Carlo has been implemented and is described in [10]. The trigger requirements are given in a trigger specification file (Trigspec in Table 1) which is read in at initialization time. To change a trigger configuration one need only change the input file, not the actual FORTRAN code. The trigger specification is checked event by event and three logical variables are set which indicate whether the event would have passed a given trigger:

LEVEL_1_TRIGGER
LEVEL_2_TRIGGER
LEVEL_3_TRIGGER

These variables are available in the TRIG_COMMON.INC Common Block and may be accessed during analysis. A summary of the number of events satisfying each trigger is given in a table in the list file.

Example #5 of the FAST Monte Carlo shows an implementation of a one-prong and a two-prong trigger applied to the file ALL_EVENTS.DAT, which is a file of deep inelastic events. For example, 26 events satisfy level 1, 26 events satisfy level 2 and 19 events satisfy level 3 of the two-prong trigger. The subroutine which actually implements the trigger is TRIG_ANAL, which also fills a variety of useful histograms defined in TRIG_HSTDEF. These histograms may be used to determine the requirements in the trigger which will maximize the signal to background for a particular process. The trigger file specification in the example was written to enhance the detection of the $\Delta(1232)$.

3.2.6 Example 6: N* Analysis

This example [3] makes the most extensive use of the utility routines available with the LUND Monte Carlo [11]. It has been widely used as a basis for simulation of the N* physics program. Resonance decay parameters are calculated from the kinematical variables of the virtual photon and the scattered proton. The angles used in histograms are the N* center-of-mass decay angles. The acceptance is tabulated for the CLAS detector as a function of Q^2 and W.

4 Summary

FAST is a program which can be used quickly to understand how the CLAS detector may be used in a specific experiment. We have adopted specific conventions and programming environment have been developed. The initial version of FAST was developed at CEBAF on the VAX 8700. The program takes roughly an hour of CPU time to process 100k events. Thus rate estimates can be made for the various physics proposals for the CLAS detector within a reasonable time. Presently several aspects of the program may not be immediately ported to other computers. We will react to this problem as we understand how compatibility limits the use of the program. Still, modularity is retained where possible, so some sections may be used in other contexts. For example, the parametrization of the acceptance functions are self-contained. If an entirely different environment is desired for a fast Monte Carlo, FAST may be used as an example to develop such a program.

A LUND Variables

The four-momenta of each track are stored and manipulated with the routines available from the LUND Monte Carlo. This Monte Carlo has been widely used in high energy physics and has many useful features. However, by and large the FAST Monte Carlo only uses the structure to store data and retrieve

---

[3] The routines for this example include the analysis routine FAST_ANAL_NSTAR.FOR, the histogram initialisation subroutine FAST_HSTDEF_NSTAR.FOR and a summary printed with the subroutine FAST_END_NSTAR.FOR. These routines were written by Steve Dytman.

information. In this section we give the information which is most useful. Two arrays in common provide information about each track (labeled by a track number j). We adhere to the original LUND convention by use of the names "k", "n" and "p" in order that our source is compatible with their documentation.

k(j,1)/10000 — 0=undecayed and 2=decayed particle, 4=beam or target.
k(j,2) — KF = particle identification code (see Table 3).

n — Total number of tracks in event
ntrks — Number of undecayed tracks in event record

p(j,1) — x component of momenta (GeV/c)
p(j,2) — y component of momenta (GeV/c)
p(j,3) — z component of momenta (beam direction) (GeV/c)
p(j,4) — particle total energy (GeV)
p(j,5) — particle mass (GeV/c^2)

The array PMAS(KF) and function LUCHGE(KF) will return the mass and three times the charge for the particle identifier KF. More details may be obtained directly from the LUND documentation [11] if more than a casual use of their routines are required.

B FAST Subprograms

We list all the subprograms written for the FAST Monte Carlo and their arguments to be used as a quick reference. More detailed information on each one may be found in FASTMC$DEMO:PROG_SCR.LIS and PROG_INC.LIS. The latter file is a description of the common blocks used by FAST.

function accept_cer (theta,phi,p,icharge,ipol,ztarget)
function accept_cer90 (theta,phi,p,icharge,ipol,ztarget)
function accept_ch1 (theta,phi,p,icharge,ipol,ztarget)
function accept_ch2 (theta,phi,p,icharge,ipol,ztarget)
function accept_ch3 (theta,phi,p,icharge,ipol,ztarget)
function accept_phi (theta,phiin,ztarget)

Table 3: Particle identification codes used by the LUND Monte Carlo. A negative code corresponds to the antiparticle of the particle listed. This list is only partial, other entries may be found on page 37 of the JETSET documentation.

Particle Identification code (KF)	*Particle*
1	γ
7	e
9	μ
17	π^+
18	K^+
19	K^o
23	π^o
24	η
27	ρ^+
33	ρ^o
34	ω
35	ϕ
57	Λ
41	p
42	n
43	Σ^+
44	Σ^o
45	Σ^-

```
function accept_sci (theta,phi,p,icharge,ipol,ztarget)
function accept_shower (theta,phi,p,icharge,ipol,ztarget)
function accept_shower90 (theta,phi,p,icharge,ipol,ztarget)
function Bdl (theta)
subroutine dump_level12_data
subroutine dump_level3_data
function eff_cer (pmom,ipart,theta,phi)
function eff_ch1 (pmom,ipart,theta,phi)
function eff_ch2 (pmom,ipart,theta,phi)
function eff_ch3 (pmom,ipart,theta,phi)
function eff_sci (pmom,ipart,theta,phi)
function eff_shower (pmom,ipart,theta,phi)
subroutine fast_anal (ntrks)
subroutine fast_anal1 (ntrks)
subroutine fast_anal2 (ntrks)
subroutine fast_anal3 (ntrks)
subroutine fast_anal4 (ntrks)
subroutine fast_anal_nstar (ntrks)
subroutine fast_data_read (ntrks,eof,error)
subroutine fast_data_write (ntrks,eof,error)
subroutine fast_end
subroutine fast_end_nstar
subroutine fast_end_user
subroutine fast_getpar (j,ipart,px,py,pz,E,xmass,icharge,
          pmom2,pmom,beta,theta,phi)
subroutine fast_hstdef
subroutine fast_hstdef1
subroutine fast_hstdef2
subroutine fast_hstdef3
subroutine fast_hstdef4
subroutine fast_hstdef_nstar
subroutine fast_init
subroutine fast_list (line)
Program FAST_MAIN
subroutine fast_read (ntrks,eof,error)
subroutine fast_setup (ntrks)
subroutine fast_smear (j,px,py,pz,pmom,beta,th,phi)
```

```
function find_module (theta,phi_local,surface,pmom,
        ipolar,ztarget)
subroutine parse_level12_data(new_line)
subroutine parse_level3_data(new_line,trig_num,or_num)
subroutine parse_level3_track(new_line,itrack)
function pathlength (theta,phi,surface,pmom,icharge,ztarget)
function range (theta,phi,pmom,ipart,plane,ztgt)
function rdist (theta,phi,plane,ztarget)
function response_cer (pmom,ipart,theta,phi)
function response_sci (pmom,ipart,theta,phi)
function response_shower (pmom,ipart,theta,phi)
subroutine rotate_level12_data
subroutine rotate_level3_data(trig_num,rot_num)
function sci_len (theta)
function sigma_dpp (theta,p,beta,vtxuse,opt)
function sigma_phi (theta,p,beta,vtxuse,opt)
function sigma_theta (theta,p,beta,vtxuse,opt)
function sigma_time (theta)
function sigma_vertex (theta,p,beta,vtxdef,opt)
function solve_cube (R)
Integer function target_flavor ( )
function target_mass( )
function tgtsect (theta,phi,vtx_z,vtx_r,vtx_phi,
       tgt_radius,tgt_length)
subroutine trigger12_anal
subroutine trigger3_anal
integer function trigger3_momentum_smear(pmom)
integer function trigger3_theta_smear(theta)
subroutine trig_anal (ntrks)
subroutine Trig_hstdef
subroutine trig_readin
subroutine ugtnam (lnam,filnam,lret)
```

References

[1] A similar program was written for CLEO by M.D. Mestayer and T. Reeves, "PHASTMC – A Program to Calculate tracking errors without simulating hits or finding tracks," CBX 88-04, February 17, 1988.

[2] "GEANT3," CERN Data Handling Division DD/EE/84-1, September 1987.

[3] The program MOMRES by B. Mecking was used to obtain the resolution functions. See also B. Mecking, Nucl. Instr. Meth. 203, 299 (1982).

[4] Elton S. Smith, " Fast Monte Carlo Program for the CLAS Detector," CLAS-NOTE-89-003, March 13, 1989.

[5] Elton S. Smith, " Fast Monte Carlo Program for the CLAS Detector – Update #1," CLAS-NOTE-89-009, July 10, 1989.

[6] "CEBAF End Station B," DOE Semi-Annual Review, CEBAF, September, 1988.

[7] D. Joyce, "CELEG 1.0, CEBAF Large Acceptance Spectrometer Event Generator User's Manual," CLAS-NOTE-89-004, 1988.

[8] "PAW – Physics Analysis Workstation," CERN Data Handling Division, Version 1.07, October10, 1989.

[9] "A Large Acceptance Spectrometer for CEBAF," Report to the LAS Technical Advisory Committee, CEBAF, November 17-19, 1988.

[10] David C. Doughty, " Trigger Simulation using the FAST Monte Carlo," CLAS-NOTE-89-008, July 13, 1989.

[11] "The LUND Monte Carlo Programs," includes the preprints LU TP 85-10, LU TP 86-22, LU TP 87-3, DESY 87-018 and DESY 86-131.