# CLAS12 Geant4 Simulations

D. Carman[c], M. Defurne[c], R. De Vita[b], M. Garcon[c], Y. Gotra[a], A. Kim[d], A. Kim[d], N. Markov[c], S. Procureur[c], M. Ungaro[a], Y. Sharabian[a], C. Wiggins[a], V. Ziegler[a], List Inconplete[a]

[a]*Thomas Jefferson National Accelerator Facility, Newport News, VA, USA*
[b]*Istituto Nazionale Di Fisica Nucleare, Genova, Italy*
[c]*IRFU, CEA, Universitè Paris-Saclay, F-91191 Gif-sur-Yvette, France*
[d]*University of Connecticut, Storrs, Connecticut*

**Abstract**

The Geant4 Monte-Carlo (GEMC) package is used simulate the passage of particles thrugh the various CLAS12 detectors. The geometry is implemented through a database of geant4 volumes created either through the GEMC native API, by the CLAS12 COATJAVA geometry service or imported from the CAD engineering model. The true information is digitized with a plugin mechanism by routines specific to each detectors and include the use of the CLAS12 CCDB calibration constants to produce both ADC, TDC and Flash ADC (FADC) response functions. Theoretical models producing generated events interface with GEMC through the LUND format. The merging of simulated data with real random trigger data provides a mechanism to includes both beam and electronic background into the simulation of generated events.

## 1. Overview

GEMC [1] is a c++ framework that uses geant4 [2] to simulate the passage of particles through matter. It provides:

- application independent geometry description
- easy interface to build / run experiments
- cad/gdml imports

The simulation parameters are stored in external databases and are used to define the geant4 objects at run time. This includes:

- geometry
- materials
- mirrors
- physics list
- database constants
- digitization
- electromagnetic fields

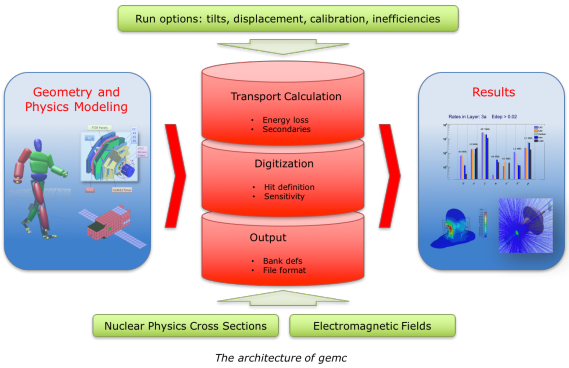



Figure 1: The architecture of GEMC. The geant4 objects are defined in external database. geant4 is used to simulate the passage of particles through materials, and hits are digitized with plugins defined by users and collected in user-defined outputs.

Particles are transported through matters and produce radiation, hits, secondaries. gemc then collects the geant4 results and produce the output specified by the user. The design of the framework is summarized in Fig. 1

### *1.1. Geometry and Materials Import*

The geometry and material are stored extrnal databases that can be mysql tables or text files that

mimic the mysql tables. The databases can be defined using the following factories:

- gemc native api (perl or python)
- geometry service
- cad (stl, ply, obj formats)
- gdml, c++ plugins (not used in CLAS12)

The gemc native api repository is `https://github.com/gemc/api`. The geometry service code repository is `https://github.com/JeffersonLab/clas12-offline-software/blob/development/common-tools/clas-jcsg/src/main/java/org/jlab/detector/geant4/v2/`

### *1.1.1. Geometry database using native api or geometry service*

An example of the code to define and store a geant4 volume in external databases is shown below. The numbers are typically hardcoded, come from a calibration databases or are defined in the geometry service. The material has a similar interface.

```
my %detector = init_det();

$detector{"name"}        = "shield";
$detector{"mother"}      = "root";
$detector{"description"} = "bst";
$detector{"color"}       = "88aaff";
$detector{"pos"}         =
"0*cm 1*cm 2*cm" ;
$detector{"material"}    = "G4_W";
$detector{"type"}        = "Tube";
$detector{"dimensions"}
= "0*mm 10*mm 20*mm 0*deg 260*deg";
$detector{"visible"}     = 1;
$detector{"style"}       = 1;
print_det(\%configuration, \%detector);
```

### *1.1.2. Importing CAD volumes from the engineering model*

The Hall-B detectors and their supports are designed with 3D CAD software. This includes a reference system and the hierarchy of all detector elements, down to details like nuts and bolts.

The CAD models are exported into STP files [3]. In order to import them into a geant4 simulation, first the volumes relevant to the simulation are selected, see Fig. 2.

The elements in the STP file are then "tessellated": several polygon shapes are created to define a geant4 volume. The software used to do this is FreeCad [4]. An example of tessellation showing the polygon shapes is shown in Fig. 7.

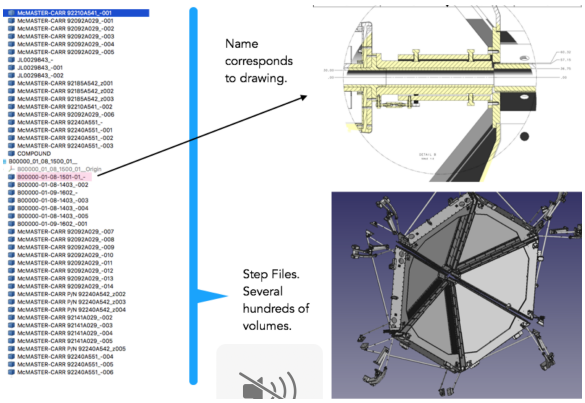



Figure 2: The selection of the volumes that will be used in the GEMC geant4 simulation. This typically involves filtering out unnecessary volumes that are not in the active region.

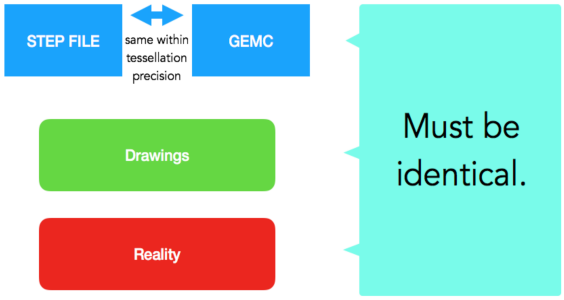



Figure 3: There are four possible representation of a volume: the one coming from the STP file and the tessellated one are exactly the same object (within the tessellation precision). In some cases, certain elements and their sizes and positions did not match the drawings. In other cases, they did not match what was built and measured, or their position did not match the survey. To eliminate these occurrences physicists and engineers worked until the final iteration of a volume was the same in all 4 models: STP/GEMC, Drawings and Reality

The simulated CAD import is as close to reality as the engineering model is close to reality. We did encounter differences between the STP files, the drawings and reality in a few occasions and worked out a workflow to eliminate any discrepancy, see Fig. 3.

An example of the cad validation is shown in Fig. 4.

## *1.2. Magnetic Fields*

The magnetic fields are loaded from ascii files. The following geant4 parameters are loaded from command line options or configuration files at run time:

- minStep: minimum step in the magnetic field

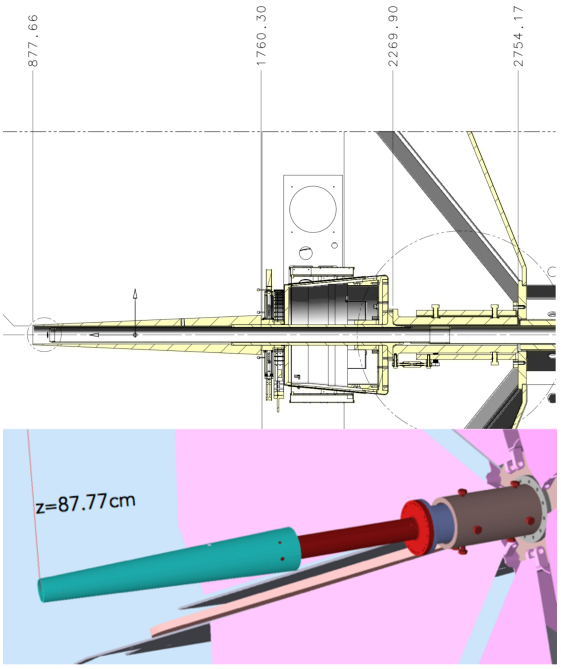



Figure 4: An example of comparing the gemc simulation to the drawings. Top: engineering drawings of the CLAS12 beamline and shielding. The start of the Moeller shielding is 87.77 cm downstream of the target center. Bottom: a geantino is shot vertically at z=87.77 cm, showing that the geant4 cone position agrees with the drawings.

- integralAlgorithm: compute the field value from the closest cell or using a linear interpolation
- interpolationMethod: interpolation algorithm, such as G4ClassicalRK4, G4SimpleRunge, etc

The implementation of the CLAS12 magnetic fields is described in 14.2.

### *1.3. Event Time Window and Hit definition*

The geant4 sensitive volumes are associated with a gemc identifier that contains hierarchical information such as mother volumes and volume copy number.

Each detector is associated with a time quantity to mimic the readout electronic time window.

The time window and identifier define a gemc hit from a series of geant4 steps: all the steps in a given identifier that are within the time window are part of the same gemc hit. The algorithm is illustrated in Fig. 5.

### *1.4. Process ID*

A "processID" method can be implemented by some digitization algorithm to modify or add gemc identifiers to each geant4 step. For CLAS12 this happens in two cases, detailed for each detector:

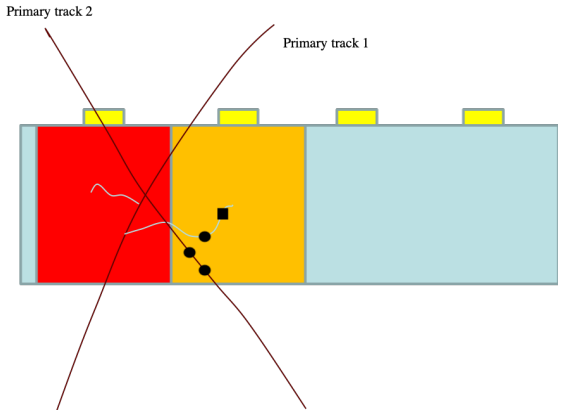



Figure 5: The GEMC Hit definition algorithm. Two sensitive cells, in red and yellow, are hit by four particles, two primaries and two secondaries. The geant4 steps in the yellow sensitive cell are drawn explicitely. The circle steps generated by both the primary track #2 and the secondary from track #1 are within the time window and therefore are all part of the same gemc hit. The single square step generated by the secondary from track #1 comes after and does not fall in the circle time window, therefore it is part of a new gemc hit.

- a paddle is hit but two outputs are produced because there is one PMT at each end of the scintillator. This is the case for CTOF, CND, FTOF.
- some hardware elements are not present in the simulation. This is the case for Drift Chambers, where the volumes do not contain the individual wires. In this case the wire id is calculated based on the position of the hit in the mother volume. In CLAS12 this mechanism is adopted by the DC, BST and Micromegas.

### *1.5. Detectors and Hit Process Plugin Mechanism*

The detector are associated with c++ digitization routines at run time, see Fig. 6

This allows the routines to be developed independently from the core code. Abstract methods can be derived in the individual detectors hit processes to define the treatment of the geant4 steps within the detector time window to provide three kind of outputs:

- a bank with digitized variables for each hit
- a bank with digitized variables for geant4 step
- a bank with an analog voltage versus time signal for each geant4 hit

The digitized banks are detailed in each detector subsitem, for example in 2.

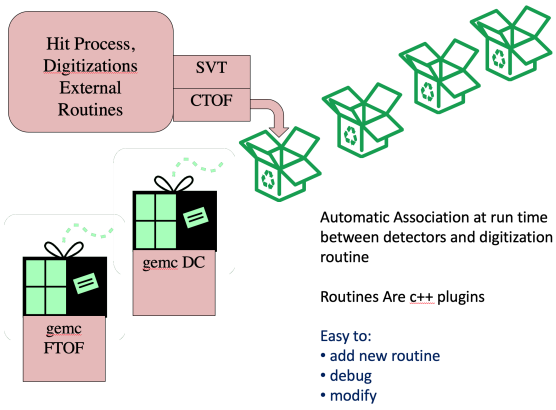



Figure 6: Hit Prcess Digitization Plugin association. The detector are associated by name with the hit process routine plugins at run time: The routine are registered in the hit process map and are called during digitization.

### *1.6. True Information*

The true information is saved in the output when requested, per detector. The true information can be integrated (one variable entry per gemc hit) or verbose (one variable entry per geant4 step). The true information variables are summarized in Table 1.

| Variable | Description |
|---|---|
| pid | ID of the FP |
| mpid | ID of the mother of the FP |
| tid | Track ID of the FP |
| mtid | Track ID of the mother of the FP |
| otid | Track ID of the ancestor of the FP |
| trackE | Total energy of the FP |
| totEdep | Total energy deposited (in MeV) |
| avg_x | Average X position (in mm) |
| avg_y | Average Y position |
| avg_z | Average Z position |
| avg_lx | Average local X position |
| avg_ly | Average local Y position |
| avg_lz | Average local Z position |
| px | x of momentum of the FP (in MeV) |
| py | y of momentum of the FP |
| pz | z of momentum of the FP |
| vx | x of the FP's origin (in mm) |
| vy | y of the FP's origin |
| vz | z of the FP's origin |
| mvx | x of the FP mother's origin |
| mvy | y of the FP mother's origin |
| mvz | z of the FP mother's origin |
| avg_t | Average time |
| nsteps | Number of geant4 steps |
| procID | Process that created the FP. |
| hitn | Hit ID |

Table 1: The true information bank. Quantities like totEdep integrates the information within one hit, but some variables only record the First Particle (FP) entering the sensitve volume.

### *1.7. Database Constants*

The mechanism to read, store and make available the CCDB [5] calibration constants is run automatically at the start of the run and every time the run number changes. Individual detector add their constants by implementing the corresponding abstract method.

The list of constants loaded is detailed in each detector implementation section.

### *1.8. Digitization*

The digitization routines are called at the end of each event, after the geant4 navigation has propagated all tracks and gema has collected all the steps into hits.

The process routine digitize each hit by iterating through all the steps in it and collecting a a number of variables onto the detector bank.

There are four different flavor of digitization that gives different outputs:

- integrated (one bank per hit): this is implemented for all CLAS12 detectors.
- step-by-step (one bank per geant4 step): used for debugging.
- voltage: the analog signal versus time calculated as response of the detector to tracks passing through it
- fadc: the same FADC bank from crate/slot/channel as written by the CLAS12 data acquisition. This is implemented for the calorimeters and the time-of-flight detectors

The digitization is detailed in each detector implementation section.

### *1.9. Output*

The output is available in two formats, identical in content: text (ascii) and evio [6], the Jefferson Lab data acquisition format. Utilities were used to convert the evio format onto ROOT [7].

The various output banks include:

- header: timestamp, event number, run number, etc.

- generated: generated particle information as seen by geant4. This bank includes a summarized information of the interaction of the particles with each detector such as number of hits, total energy deposited, etc. This summary includes the interactions of all the primary particle secondaries.
- user header: a copy of the generator
- beam radio-frequency signal: mimics the accelerator bank, a 248 MHz signal.
- detector true info, per hit or per step
- detector digitized info, per hit or per step
- detector voltage vs time
- detector fadc signal
- ancestor: the complete hierarchy of the primary and secondary particles.

The various banks are ogranized using an unique integer identifier.

### *1.10. Background Merging*

Real data can be merged with simulated events, typically from random trigger to emulate physics and electronic background in the various detectors. The data is undigitized using the inverse digitization to calculate the energy and real timing from ADC and TDC values. It is then saved in text files indexed by event number and detector ID. This makes also easy to scale the background luminosity by grouping several events into one; for example, grouping two events at 50 nA beam current give one event with background from 100 nA current. The energy information is re-digitized using the same algorithm used for the geant4 steps, producing additional hits to the ones coming from simulation.

## 2. CLAS12 Geometry Implementation

The following sections describe the implementation of the individual CLAS12 components into the simulation.

## 3. Target

The CLAS12 target components are imported from the engineering model. The STEP files are converted to tessellated STL files and imported in the GEMC simulation [8], [9].

An example of the tessellation is shown in Fig. 7.

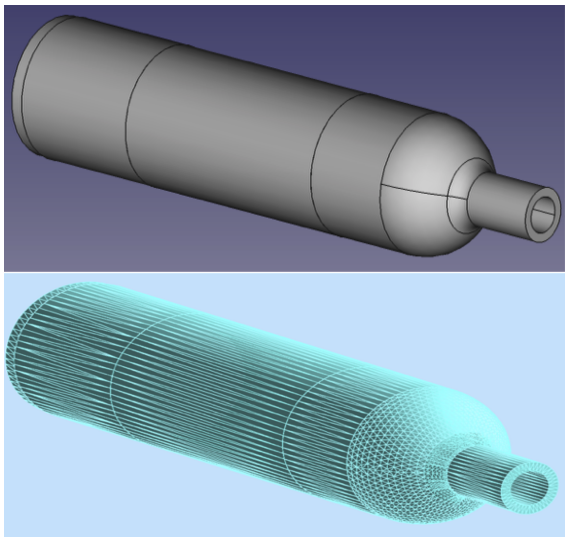


Figure 7: An example of volume from a STP file tessellated in GEMC. The volume that is shown is the target scattering chamber. Top: the CAD representation in the engineering model. Bottom: the tessellation.

Key elements of the STL import include the torlon tube to the target cell, the target aluminum windows and kapton walls and the scattering chamber, see Fig. 8.

An overview of the target in geant4 and the engineering model is shown in Fig. 9.

#### *3.0.1. Geometry Git Location*

The github location of the gemc perl api scripts and the STL files is `https://github.com/gemc/detectors/tree/master/clas12/targets`.

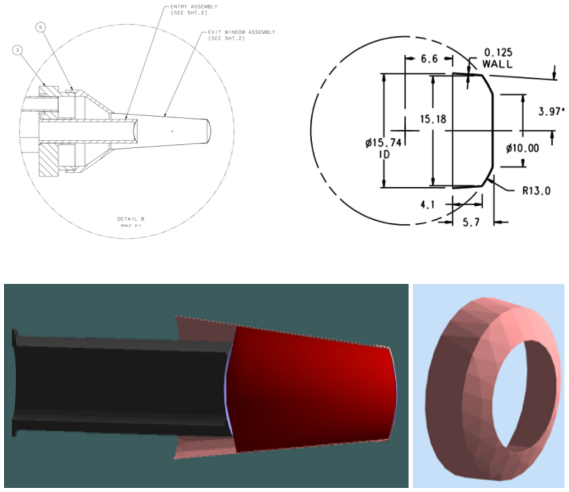



Figure 8: The CLAS12 target design. Top left: the entry assembly schematic. Top right: the liquid hydrogen cell dimensions: the outer radius is tapered down from 15 mm at z=-2.5cm to 10mm at z=2.5mm. Bottom left: The cell implementation in GEMC from the CAD drawings. From left to right (beam direction): the black torlon tube, the upstream aluminum window, the target cell, the kapton cup and the downstream aluminum window. Bottom right: the GEMC implementation of the kapton cup.

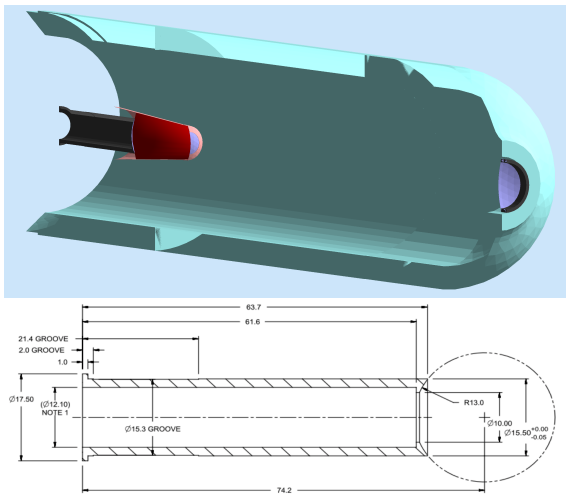



Figure 9: Top: overview of the target implementation in GEMC includes the scattering chamber (cyan color), the downstream cup near the right of the figure and 50 micron aluminum window. Bottom: the torlon base tube starts at a radius of r=6.06 mm, and ends at r = 7.75 mm. It is 63.7 mm long.

## 4. Barrel Silicon Vertex Tracker (BST)

### *4.1. Geometry*

The BST geometry is implemented through the COATJAVA geometry service. The service provides the geant4 definitions that are read by the GEMC perl api to build the geometry database.

There are three BST regions, with 10, 14 and 18 sectors for region 1, 2, and 3 respectively, see Fig. 10. Each sectors contains a module with three sensors, two readout chips, and several layers of material, including (length of line represents depth level):

- – wirebond
- —- silicon
- —— epoxy
- ——— rail
- ————- bus cable
- ————— carbon fiber
- ————— rohacell
- ————— carbon fiber
- ————- bus cable
- ——— rail
- —— epoxy
- —- silicon
- – wirebond

The sensitive sensor are assigned the silicon material and associated with the bst hit process routine. The strip identification is performed in the Process ID routine.

#### *4.1.1. Geometry Git Location*

The github location of the gemc perl api script is `https://github.com/gemc/detectors/tree/master/clas12/bst`. The geometry service definitions in coatjava is `https://github.com/JeffersonLab/clas12-offline-software/blob/development/common-tools/clas-jcsg/src/main/java/org/jlab/detector/geant4/v2/SVTGeant4Factory.java`

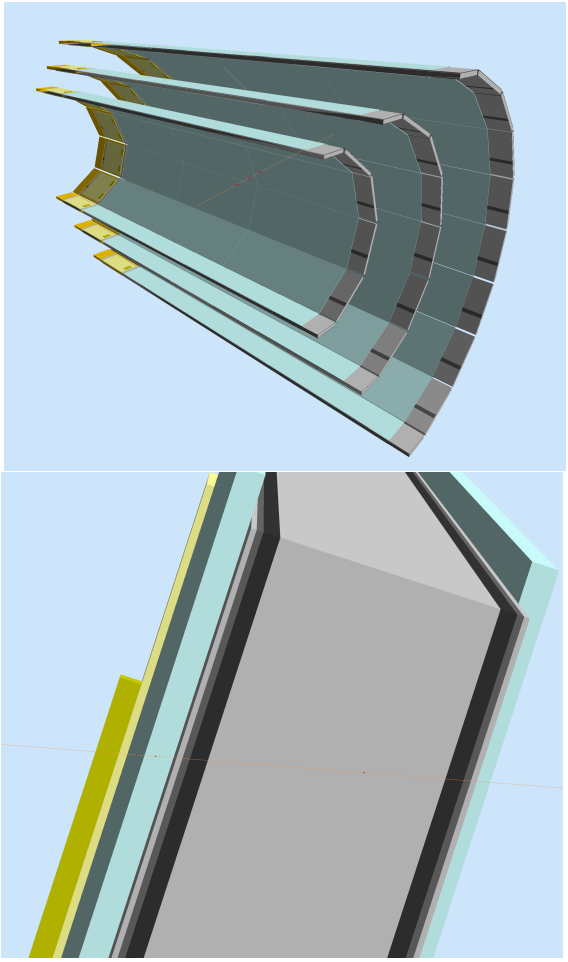


Figure 10: Top: the GEMC implementation of the BST geometry. The three regions are shown in the sliced view. The silicon sensors are in cyan color. The red track is a 2 GeV proton, leaving hits in each module crossed. Bottom: detail of a module shows the various material inside. The 320$\mu m$ silicon sensor is on top and bottom of the module. The material inside includes epoxy glue, the bus cable and support material.

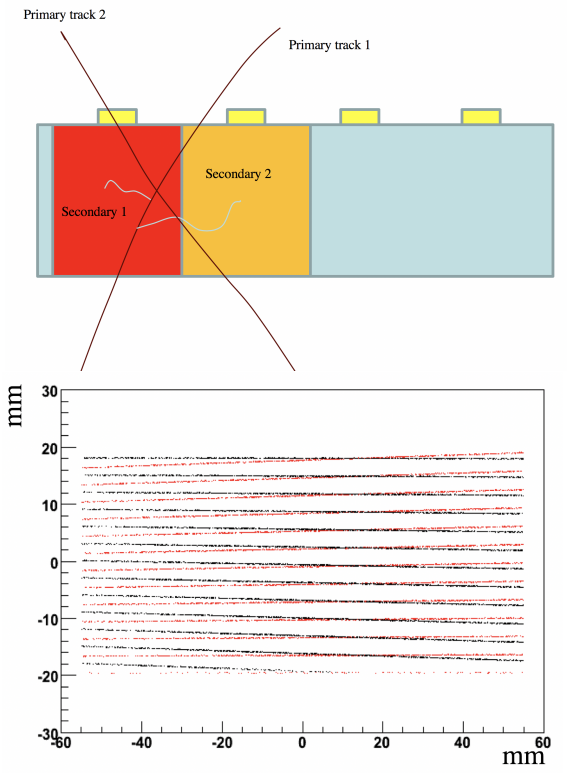



Figure 11: Top: process ID algorithm cartoon for BST. The strip ID is assigned based on the local position of the track step within the sensitive module. If a step of a primary or a secondary particle happens between the boundary of strip that was already hit, it will be assigned to that strip. Bottom: actual hit position of selected strips in the top and bottom silicon sensors of a module shows the fan-like distribution of the strips, with increasing angle. The top layer angle is the opposite of the bottom layer angle.

### *4.2. Process ID*

At each geant4 step, the local coordinate in the sensor volume are used to calculate the strip id. The algorithm includes the dead zone around the sensor, the pitch between the strips (156$\mu m$) and the angle between the strips, that varies from 0 (strip n. 1) and 3 degrees (for strip n. 256). A showcase of the strip id assignment is summarized in Fig. 11.

Due to the thickness of the silicon sensor, the electrons avalanche produced can end up in more than one strip. This is reproduced in the GEMC simulation using the hit sharing algorithm, see Fig. 12.

### *4.3. Digitization*

#### *4.3.1. ADC*

The BST digitization provides a 3 bit ADC, using the total energy deposited (after hit sharing) between 26 and 117 keV.

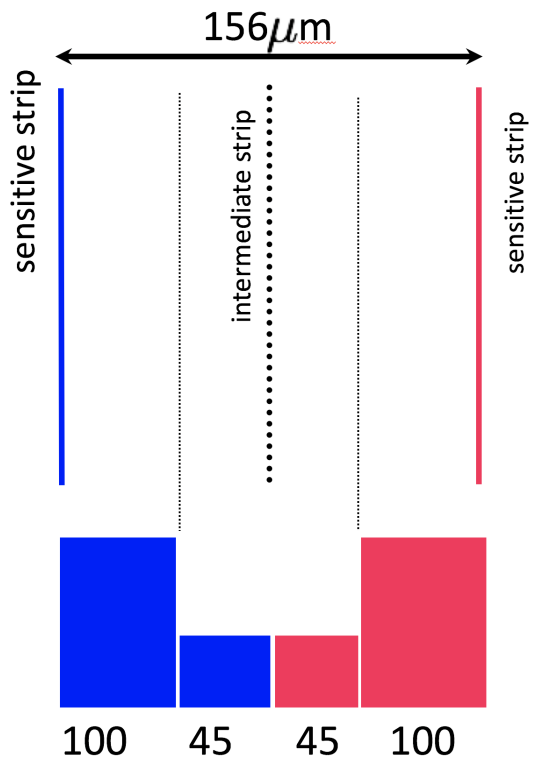



Figure 12: The BST hit sharing algorithm. If a step in a given strip happens within 39$\mu m$ of another strip, then 90% of the energy deposited will be shared equally between them (45% each), with a 10% energy loss due to capacitance between strip and backplane.

#### 4.3.2. TDC

The Bunch Cross Oscillator quantity (BCO), a random number between 0 and 255, provides the timing info associated with the hit.

### 4.4. Digitized Bank

The digitized output bank has $ID = 100$, and the variables are summarized in Table 2

#### 4.4.1. Time Window

The timewindow of the BST is set to to 132 ns.

#### 4.4.2. Process Routine Git Repository Location

The BST hit process routines are located in the repository: `https://github.com/gemc/source/tree/master/hitprocess/clas12/svt`

#### 4.4.3. Radiation dose and background rates

A detailed study of the backgrond rates coming from beam interacting with the target was done to ensure that the silicon sensor could operate in the high radiation conditions of the target proximity.

Given the nominal operating luminosity $L = 10^{35} cm^{-2} s^{-1}$, and the liquid hydrogen target of $5cm$, the beam electron rates is $R = 4.7 \times 10^{11} Hz$. This correspond to around 62,000 electrons in the BST 132ns time window.

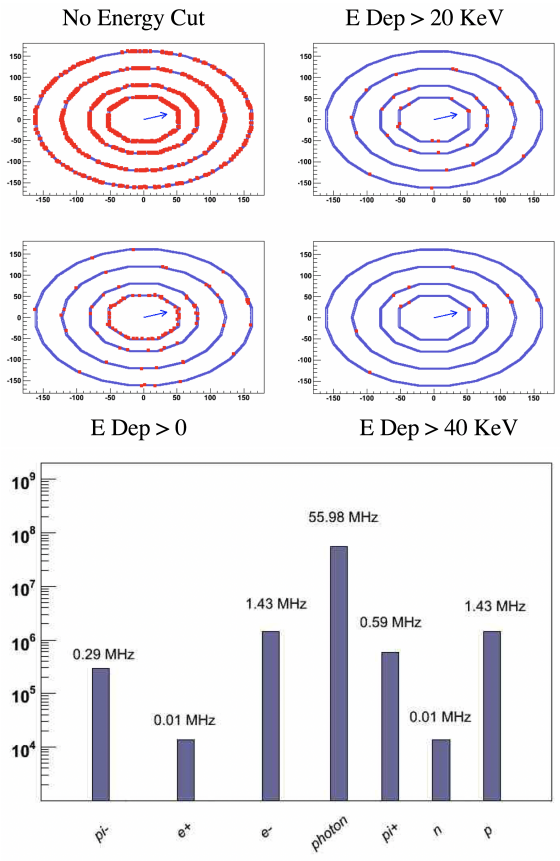
Simulations using 62,000 11 GeV electrons per event impinging on the liquid hydrogen target were analyzed.

The rates were calculated for the various region and for different detector threshold. The radiation dose and the neutron equivalent damage was estimated. Most of the radiation is released in the first two layers of the BST. The 370 rad / year are low enough to grant the BST operation for 15 years. The results of the study are summarized in Fig. 13

In addition a thin layer of tungsten was added after an analyis aimed at reducing the electromagnetic background [10].

| Variable | Description | Tag |
|---|---|---|
| layer | layer number | 1 |
| sector | sector number | 2 |
| strip | strip number | 3 |
| ADC | 3 bit ADC | 4 |
| bco | 8 bit time info | 5 |
| ADCHD | 13 bit ADC | 6 |
| time | Time information | 7 |
| hitn | hit number | 99 |

Table 2: The digitized BST bank




| Layer | GeV/s | GeV/(s cm²) | mrads/s | mrads/(s cm²) | rad/year | rad/(year cm²) |
|---|---|---|---|---|---|---|
| 1a | 7147.8 | 19.05 | 4.39 | 0.011708 | 138480 | 369.24 |
| 1b | 3889.4 | 10.37 | 2.38 | 0.006371 | 75352 | 200.92 |
| 2a | 4578.2 | 4.06 | 0.93 | 0.000833 | 29566 | 26.27 |
| 2b | 3551.7 | 3.15 | 0.72 | 0.000646 | 22936 | 20.38 |
| 3a | 3828.5 | 1.51 | 0.34 | 0.000137 | 10988 | 4.34 |
| 3b | 3295.7 | 1.30 | 0.29 | 0.000118 | 9459 | 3.73 |
| 4a | 3145.5 | 0.93 | 0.21 | 6.360e-05 | 6771 | 2.00 |
| 4b | 2743.3 | 0.81 | 0.18 | 5.547e-05 | 5905 | 1.74 |

Figure 13: Summary of radiation doses and background rates on the BST. Top: a display shows the occupancy of the various BST layers for different thresholds. Middle: the rate breakdown for different particles for a threshold of $20keV$. The current hardware threshold is $30keV$. Bottom: table showing the fluences and radiation doses in different BST layers.

## 5. Barrel Micromegas Tracker (BMT)

### *5.1. Geometry*

The Micromegas geometry is implemented through the native gemc geometry API. There are three micromegas regions, divided azimutally in three idential sectors. Each sector contain a cover layer with copper ground, the PCB with the readout strips, the kapton support, the mesh layer and the ionizing gas:

- – overlay
- —- copper ground
- —— pcb
- ——— strips
- ————- kapton
- ————— gas
- ————— mesh

The sensitive volume containing the gas is assigned the argon isobutan gas and associated with the bmt hit process routine. The geometry is summarized in Fig. 14

The strip identification is performed in the Process ID routine.

#### *5.1.1. Geometry Git Location*

The github location of the gemc perl api script is `https://github.com/gemc/detectors/tree/master/clas12/micromegas`.

### *5.2. Process ID*

At each geant4 step, the local coordinate in the sensor volume are used to calculate the strip id. The algorithm includes the lorentz angle based on the magnetic field strength, the angula, pitch between the strips, the dead zones of the sensitive parts. A virtual electron avalanche is simulated based the energy deposited. The avalanche is deposited onto one strip or distributed among several to account for the energy sharing.

### *5.3. Digitization*

#### *5.3.1. ADC*

The Micromegas digitization provides the ADC value calculated using the total energy deposited (after hit sharing).

#### *5.3.2. TDC*

There is no timing information in the output.

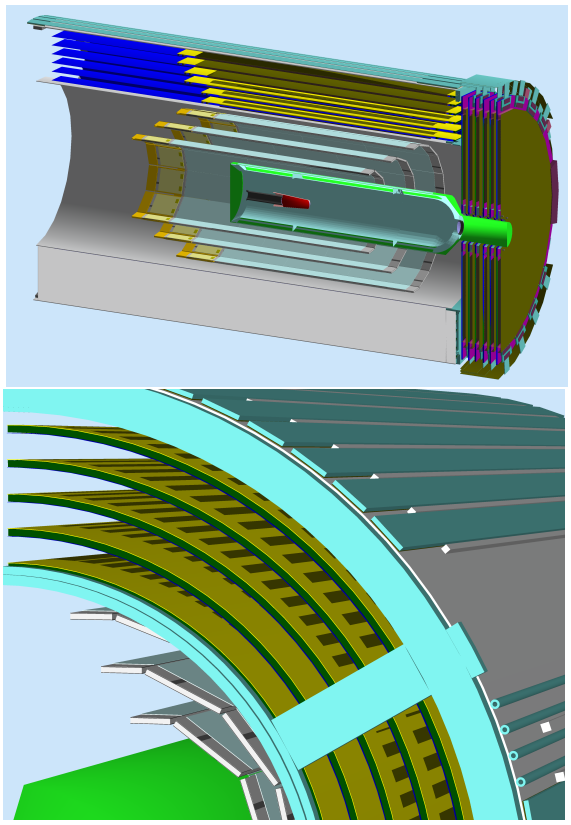


Figure 14: Top: an overall view of the central detector trackers. The target is surrounded by 3 layers of silicon vertex trackers and 3 layers of micromegas. Bottom: detail of the micromegas GEMC geometry, showing the overlay cover, the copper ground and pcb strips

#### 5.3.3. Summary of CCDB Table used

- /geometry/cvt/mvt/bmt_layer_noshim
- /geometry/cvt/mvt/bmt_strip_L1
- /geometry/cvt/mvt/bmt_strip_L2
- /geometry/cvt/mvt/bmt_strip_L3
- /geometry/cvt/mvt/bmt_strip_L4
- /geometry/cvt/mvt/bmt_strip_L5
- /geometry/cvt/mvt/bmt_strip_L6

### 5.4. Digitized Bank

The digitized output bank has $ID = 500$, and the variables are summarized in Table 3

| Variable | Description | Tag |
|---|---|---|
| layer | layer number | 1 |
| sector | sector number | 2 |
| strip | strip number | 3 |
| Edep | energy deposited | 4 |
| ADC | ADC | 5 |

Table 3: The digitized micromegas bank

#### 5.4.1. Time Window

The timewindow of the Micromegas is set to to 132 ns.

#### 5.4.2. Process Routine Git Repository Location

The BST hit process routines are located in the repository: `https://github.com/gemc/source/tree/master/hitprocess/clas12/micromegas`

## 6. Central Time-Of-Flight (CTOF)

### 6.1. Geometry

The CLAS12 CTOF paddles and light guides are imported from the engineering model. The Step files are converted to tessellated STL files and imported directly in the GEMC simulation [11]. The STL files are downloaded using the coatjava geometry service, as the same files are used in reconstruction. The paddles are assigned the scintillator material and associated with the ctof hit process routine. The light guides are also associated to the scintillator material, but they are treated as passive material and not associated with a sensitive detector.

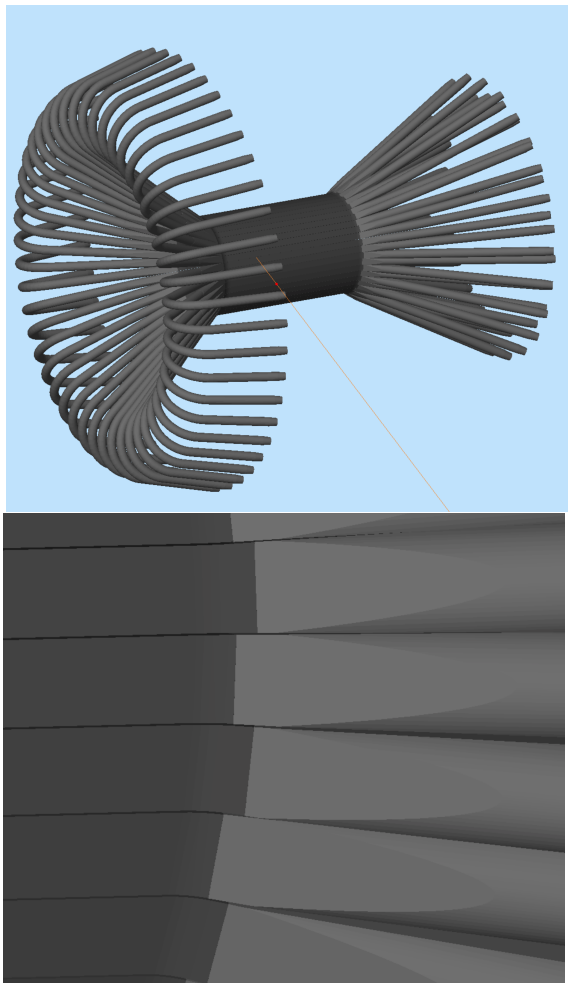


Figure 15: Top: the GEMC implementation of the CTOF geometry. The paddles and light guides imported directly from the enginerring model. The orange line shows a 2 GeV proton leaving a hit (red dot) in one of the paddles. Bottom: a zoom-in of the implementation shows the details of scintillator/light guides junction.

Each volume is typically tessellated by about 1,000 faucets. The simulation geometry captures complicated details such as the shape of the scintillator/light guides junction, see Fig. 15 bottom.

#### 6.1.1. Geometry Git Location

The github location of the gemc perl api script is `https://github.com/gemc/detectors/tree/master/clas12/ctof`. The coatjava geometry service is `https://github.com/JeffersonLab/clas12-offline-software/blob/development/common-tools/clas-jcsg/src/main/java/org/jlab/detector/geant4/v2/CTOFGeant4Factory.java`

### 6.2. Process ID

Each hit in the paddles is produces in two identical hits with the identifier variable "side" sets to 0 (for the left side PMT) and 1 (for right side PMT). The hits are then processed independently through the ctof hit process routine.

### 6.3. Digitization

#### 6.3.1. ADC

The energy deposited is reduced based on the position on the paddle using the calibration attenuation length. It is then corrected by a gain factor to account for the fact that the HV are adjusted so that the geometeric mean sqrt(L*R) is independent of counter length.

The corrected energy is converted to the theoretical number of photos $N_{th}$ using the constant $500\gamma/MeV$. A poissonian is used to calculate the actual number of photons $N_{actual}$ and the resulting "smeared" energy is the converted to adc using the FADC conversion factor.

#### 6.3.2. TDC

The absolute hit time is corrected by:

- the effective velocity (from CCDB)
- the time walk correction, calculated from the smeared energy
- a panel to panel factor (from CCDB)
- a left/rigth time offset factor (from CCDB)
- an RF correction (from CCDB)

The time is then smeared by a $\sigma$ resolution read from CCDB using a gaussian function and then digitized using a TDC conversion factor.

#### 6.3.3. Summary of CCDB Table used

- /calibration/ctof/attenuation
- /calibration/ctof/effective_velocity
- /calibration/ctof/status
- /calibration/ctof/gain_balance
- /calibration/ctof/time_walk
- /calibration/ctof/time_offsets
- /calibration/ctof/tdc_conv
- /geometry/ctof/ctof
- /daq/tt/ctof

### 6.4. Digitized Bank

The digitized output bank has $ID = 400$, and the variables are summarized in Table 4

| Variable | Description | Tag |
|---|---|---|
| sector | sector number | 1 |
| layer | layer (1: 1A, 2: 1B, 3: 2B) | 2 |
| paddle | paddle number | 3 |
| side | PMT side (0 Left, 1 Right) | 4 |
| ADC | ADC | 5 |
| TDC | TDC | 6 |
| ADCu | ADC unsmeared | 7 |
| TDCu | TDC unsmeared | 8 |
| hitn | hit number | 99 |

Table 4: The digitized CTOF bank

#### 6.4.1. Process Routine Git Repository Location

The CTOF hit process routine location in git is `https://github.com/gemc/source/blob/master/hitprocess/clas12/ctof_hitprocess.cc`

## 7. Central Neutron Detector (CND)

### 7.1. Geometry

The CND geometry is implemented through the native gemc geometry API. The paddles are Geant4 generic trapezoids, see Fig. 16. The scintillator junctions are G4Polycons. The paddles are assigned the scintillator material and associated with the cnd hit process routine.

#### 7.1.1. Geometry Git Location

The github location of the gemc perl api script is `https://github.com/gemc/detectors/tree/master/clas12/cnd`.

### 7.2. Digitization

#### 7.2.1. ADC

The energy deposited is reduced based on the position on the paddle using the calibration attenuation length. Two signals are then propagated, one directly to the PMT and one through the scintillator junction. A 30% factor takes into account losses at junctions.

The corrected energy is converted to the theoretical number of photos $N_{th}$ using the constant $1210\gamma/MeV$. A poissonian is used to calculate the actual number of photons $N_{actual}$ and the resulting "smeared" energy is the converted to adc using the FADC conversion factor.

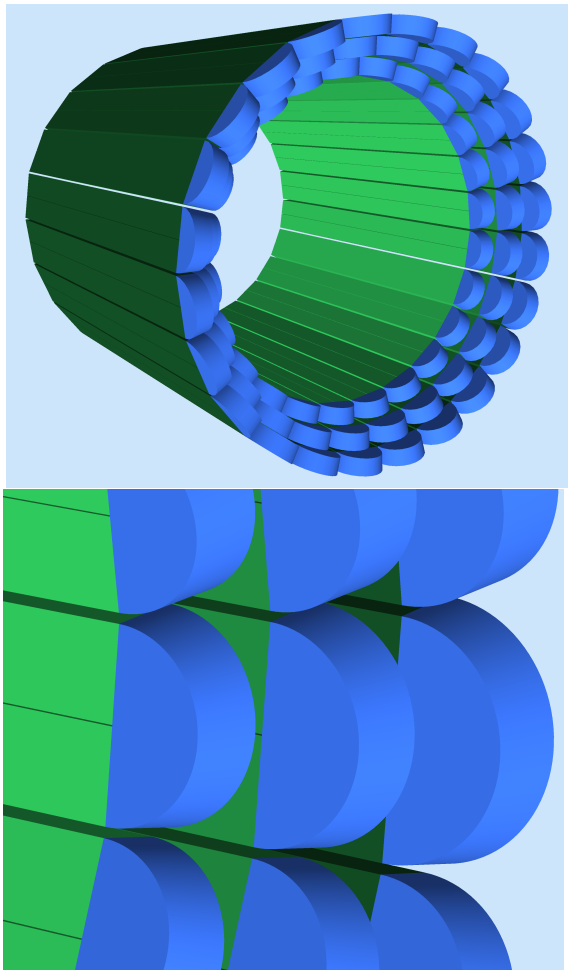


Figure 16: Top: overall view of the CND detector. Three layers of scintillators are placed at increasing z. Pairs of scintillators are connected through a scintillator u-shaped junction. Bottom: enlarged view of the junctions.

#### 7.2.2. TDC

The absolute hit time is corrected by:

- the effective velocity (from CCDB)
- a left/rigth time offset factor (from CCDB)
- the Birks Attenuation factor

The time is then smeared by a $\sigma$ resolution read from CCDB using a gaussian function and then digitized using a TDC conversion factor.

#### 7.2.3. Summary of CCDB Table used

- /calibration/cnd/Status_LR
- /calibration/cnd/TDC_conv
- /calibration/cnd/Attenuation
- /calibration/cnd/EffV
- /calibration/cnd/Energy
- /calibration/cnd/UturnTloss
- /calibration/cnd/TimeOffsets_LR
- /calibration/cnd/TimeOffsets_layer

### 7.3. Digitized Bank

The digitized output bank has $ID = 300$, and the variables are summarized in Table 5

| Variable | Description | Tag |
|---|---|---|
| sector | sector number | 1 |
| layer | layer number | 2 |
| component | component number | 3 |
| ADCL | ADC Left | 4 |
| ADCR | ADC Right | 5 |
| TDCL | TDC Left | 6 |
| TDCR | TDC Right | 7 |
| hitn | hit number | 99 |

Table 5: The digitized CND bank

#### 7.3.1. Time Window

The timewindow of the CND is set to 5 ns.

#### 7.3.2. Process Routine Git Repository Location

The CND hit process routine location in git is `https://github.com/gemc/source/blob/master/hitprocess/clas12/cnd_hitprocess.cc`

## 8. High Threshold Cerenkov Counter (HTCC)

### 8.1. Geometry

The HTCC geometry is implemented through the native gemc geometry API. The elliptical mirrors are made through a subtraction of two G4Ellipsoid. They are contained inside an HTCC mother volume made with a G4Polycone, see Fig. 17. The faces of the PMT are the sensitive volumes, associated with the quartz-glass material and associated with the htcc digitization routine.

The refractive index of the $CO_2$ and its transparency is included in the material optical properties and taken into account during the geant4 transportation of the phtotons.

The mirror and winston cones reflectivities are associated with the mirror optical properties and are taken into account and taken into account during the geant4 transportation of the phtotons.

The quartz window PMT quantum efficiency are associated with the PMT face optical properties and are taken into account in the digitization routine.

#### 8.1.1. Geometry Git Location

The github location of the gemc perl api script is `https://github.com/gemc/detectors/tree/master/clas12/htcc`.

### 8.2. Digitization

Photons that impinged on the PMT faces are processed with the digitization routine. For each photon collected undergoes the quantum efficiency algorithm at its wavelength to decided if it's finally detected.

#### 8.2.1. Timing

The time average of all the photons is saved in the output after a time shift coming from the calibration database.

#### 8.2.2. Summary of CCDB Table used

- /calibration/htcc/status
- /calibration/htcc/time
- /daq/tt/htcc

### 8.3. Digitized Bank

The digitized output bank has $ID = 600$, and the variables are summarized in Table 6

#### 8.3.1. Time Window

The timewindow of the HTCC is set to 5 ns.

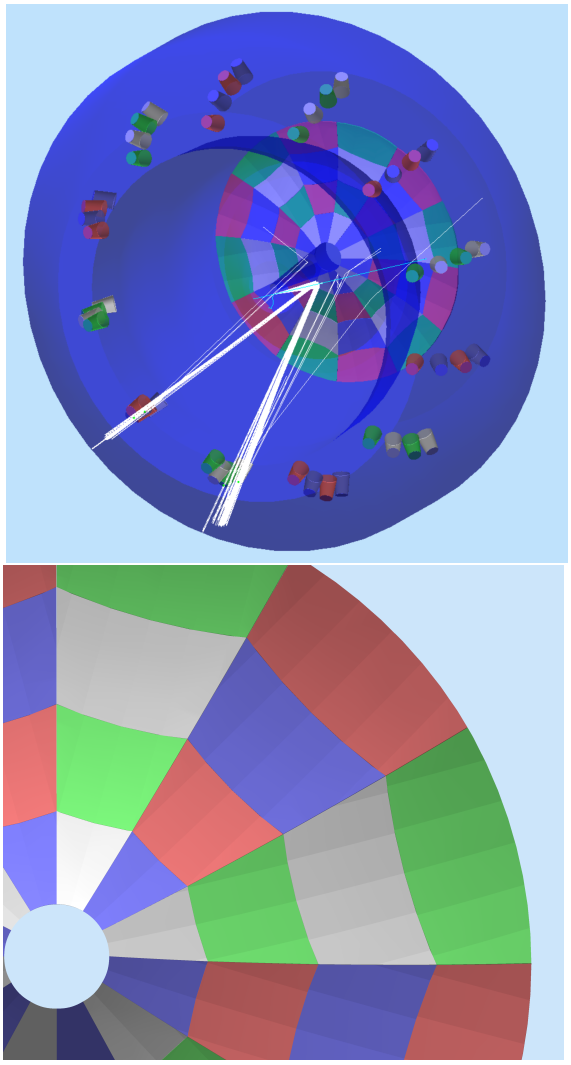


Figure 17: Top: an electron passing through the HTCC gas volume and emitting Cherenkov photons. The light cone hits two mirrors and it is re-directed to the corresponding PMTs. Bottom: details of the mirrors implementation. These are subtractions of ellipsoid geant4 volumes intersected with planes to cut the sides.

| Variable | Description | Tag |
|---|---|---|
| sector | clas12 sector | 1 |
| ring | theta index | 2 |
| half | half-sector | 3 |
| nphe | number of photoelectrons | 4 |
| time | average time of the hit | 5 |
| hitn | hit number | 99 |

Table 6: The digitized HTCC bank

#### *8.3.2. Process Routine Git Repository Location*

The HTCC hit process routine location in git is `https://github.com/gemc/source/blob/master/hitprocess/clas12/htcc_hitprocess.cc`

## 9. Forward Tagger (FT)

### *9.1. Geometry*

The FT detectors are:

- a micromegas tracker
- an hodoscope
- a calorimeter

The three structures are implemented in GEMC using the native perl api script, except for the inner shield (see beamline), which is coming from the CAD engineering mode.

The FT geometry is shown in Fig. 18.

#### *9.1.1. Geometry Git Location*

The github location of the gemc perl api script is `https://github.com/gemc/detectors/tree/master/clas12/ft`.

### *9.2. Digitization*

#### *9.2.1. ADC*

For FTCal hits, the energy deposited is converted first to the charge produced at the end of the electronics chain composed by the photosensor (APD) and preamplifier, and then to adc. The first conversion is based on the measured charge for cosmic rays that deposit a known energy in the crystals, while the second conversion is based on the FADC conversion factor. A smearing on the final adc values is added, accounting for the Poisson distribution of photo-electrons produced by the photosensor, the Gaussian noise of the photosensor and of the preamplifier. All parameters, number of

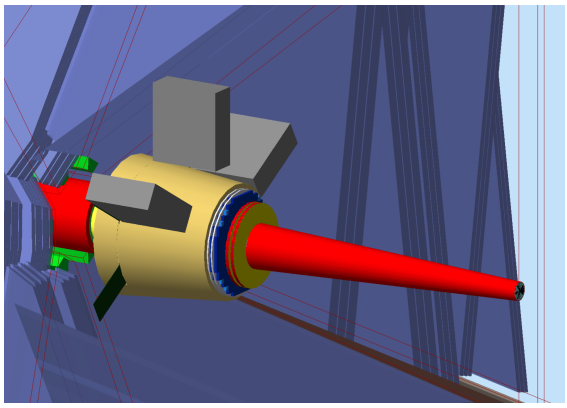


Figure 18: The three detectors in the FT geometry implementation in gemc. The red disks form the micromegas tracker. The hodoscope crystals are the blue boxes, while the calorimeter is encompassed in the yellow cone. The grey volumes contains the electronics.

photo-electrons per MeV of energy deposited, RMS of the APD noise and of the preamplifier input noise, have been tuned to experimental data.

The same approach is adopted to process FTHodo hits, in which the deposited energy is first convereted to charge and then to adc. The smearing in this case accounts only for the Poisson distribution of the measured number of photo-electrons, which dominates over other sources because of the relatively small number of photo-electrons per MeV of energy deposition.

#### 9.2.2. TDC

The TDC of FTCal hits is computed from the time of the energy deposition, accounting for the speed of the scintillation light in the crystal and the distance to the photosensor, assuming a known time-to-TDC conversion factor. A Gaussian smearing on the resulting TDC is added based on a fixed RMS derived from the experimental measurements.

Similarly, the TDC of FTHodo hits is derived from the time of a given energy deposition, adding a fixed offset before the conversion from time to TDC and a Gaussian smearing. As in previous cases, all relevant parameters have been tuned to the observed detector response.

#### 9.2.3. Summary of CCDB Table used

(. Calorimeter)

- /calibration/ft/ftcal/status
- /calibration/ft/ftcal/noise
- /calibration/ft/ftcal/charge_to_energy
- /calibration/ft/ftcal/time_offsets
- /daq/tt/ftcal

(. Hodoscope)

- /calibration/ft/fthodo/status
- /calibration/ft/fthodo/noise
- /calibration/ft/fthodo/charge_to_energy
- /calibration/ft/fthodo/time_offsets
- /daq/tt/fthodo

### 9.3. Digitized Bank

The digitized output bank has $ID = 700$, 800 and 900 for the FT tracker, the FT hodoscope and the FT calorimeter respectively. The variables are summarized in Table 7

| Variable | Description | Tag |
|---|---|---|
| | Tracker | |
| sector | sector | 1 |
| layer | layer | 2 |
| component | component | 3 |
| adc | adc | 4 |
| | Hodoscope | |
| sector | sector | 1 |
| layer | layer | 2 |
| component | component | 3 |
| adc | adc | 4 |
| tdc | tdc | 5 |
| | Calorimeter | |
| sector | sector | 1 |
| layer | layer | 2 |
| component | component | 3 |
| adc | adc | 4 |
| tdc | tdc | 5 |

Table 7: The digitized FT banks for the tracker, hodoscope and calorimeter

#### 9.3.1. Time Window

#### 9.3.2. Background merging algorithm

#### 9.3.3. Process Routine Git Repository Location

The FT hit process routines are: `https://github.com/gemc/source/blob/master/hitprocess/clas12/ft_cal_hitprocess.cc` and `https://github.com/gemc/source/blob/master/hitprocess/clas12/ft_hodo_hitprocess.cc`.

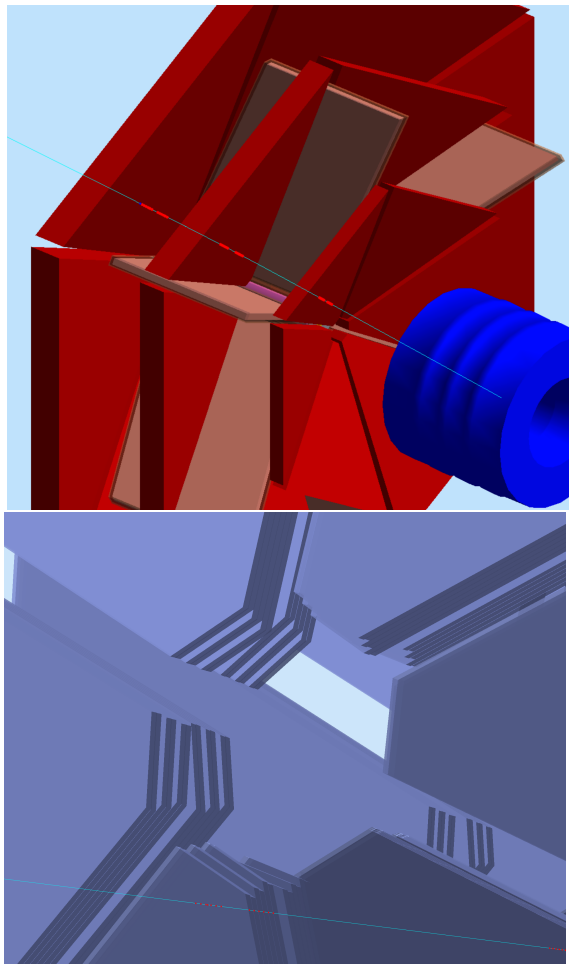


Figure 19: Top: the GEMC implementation of the DC geometry. The red trapezoids are the mother volumes corresponding to DC regions 1, 2 and 3. The cyan track is an electron, hitting the various layers inside each region (a red dot / layer is a hit). Bottom: a zoom in near the beamline shows the Stereo angle layers. Six layers represent a DC superlayer.

## 10. Drift Chambers (DC)

### *10.1. Geometry*

The DC geometry is implemented through the COATJAVA geometry service. The service provides the geant4 definitions that are read by the GEMC perl api to build the geometry database.

Each layer is a generic G4Trapezoid, tilted by $+6^O$ or $-6^O$ depending if they are part of a stereo superlayer or not. The 12 layers in each region (6 per superlayer) are placed in a region mother volume made of air, see Fig. 19. The layers are assigned the $CO_2$ material and associated with the dc hit process routine. The wire identifications is performed in the Process ID routine.

#### *10.1.1. Geometry Git Location*

The github location of the gemc perl api script is `https://github.com/gemc/detectors/tree/master/clas12/dc`. The geometry service definitions in coatjava is `https://github.com/JeffersonLab/clas12-offline-software/blob/development/common-tools/clas-jcsg/src/main/java/org/jlab/detector/geant4/v2/DCGeant4Factory.java`.

### *10.2. Process ID*

At each geant4 step, the local vertical position $y$ in the tilted g4Trap is computed. Knowing the distance between each wire $\delta Y$ (given by the total height of the trapozoid divided by the number of wires, 112), which is a constant as we're not taking into account wire sagging, the wire ID is given by $n_i = y/\delta y$.

### *10.3. Digitization*

#### *10.3.1. ADC*

There is no ADC calculation.

#### *10.3.2. TDC*

First, the distance of closest approach *DOCA* is extrapolated for the hit. At each geant4 step, the distance of the track from the wire is calculated. The *DOCA* is extracted among the points with energy deposited greatr than $50eV$, for which the sum of the step time + $DOCA/DV$ (where DV is the drift velocity) is minimal.

An initial time $T_i$ is calculated with a time to distance function which is the inverse of what extrapolated in calibration and used in reconstruction to go from TDC to *DOCA*. The function takes into account:

- the distance from the wire, in cm
- the cell size in superlayer
- the polar angle of the track
- magnitude of field in tesla

A time walk correction function is applied to $T_i$ that includes discrete ionisation effects based on the following input:

- the distance from the wire, in cm
- the cell size in superlayer
- an adjustable factor (in mm) times $beta^2$ of the particle, the factor is adjusted according to data at small distances from the wire

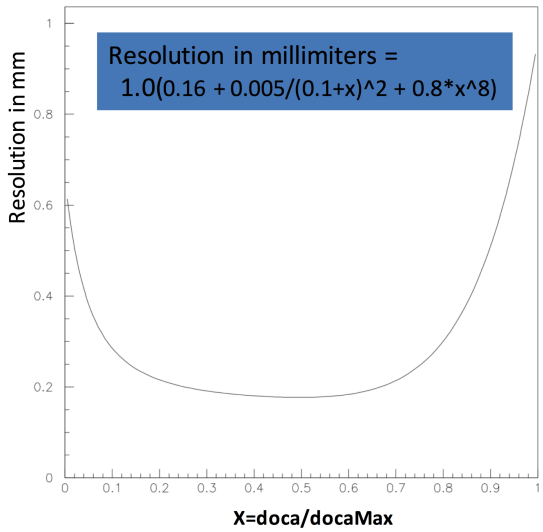



Figure 20: The fit to the data resolution provides parameters that are put in the CCDB database and read by the digitization routine at simulation run time.

- a parameter to adjust the matching between the two time walk distributions
- the velocity of the particle

The resulting $T_w$ is then used in a landau-function to mimic the detector response function.

An intrinsic random time walks correction $\sigma_{TW}$ due to multiple scattering is calculated and the time is smeared with a gaussian function using $\sigma_{TW}$ as the resolution.

An example of the resolution function that represents real LTCC data is shown in Fig. 20.

Finally, a random number is thrown and if it's above the efficiency function, calculated based on *DOCA*, the hit is rejected.

### *10.3.3. Summary of CCDB Table used*

- /calibration/dc/signal_generation/intrinsic_inefficiency
- /calibration/dc/signal_generation/dc_resolution
- /calibration/dc/time_to_distance/time2dist
- /geometry/dc/superlayer

## *10.4. Digitized Bank*

The digitized output bank has $ID = 1300$, and the variables are summarized in Table 8

| Variable | Description | Tag |
|---|---|---|
| sector | sector index | 1 |
| layer | layer index | 2 |
| wire | wire index | 3 |
| tdc | tdc value | 4 |
| LR | Left/Right ambiguity | 5 |
| doca | 2D distance of closest approach | 6 |
| sdoca | smeared doca | 7 |
| time | doca / drift velocity | 8 |
| stime | sdoca / drift velocity | 9 |

Table 8: The digitized DC bank

### *10.4.1. Time Window*

The timewindow of the DC is set to 500 ns.

### *10.4.2. Process Routine Git Repository Location*

The DC hit process routine location in git is `https://github.com/gemc/source/blob/master/hitprocess/clas12/dc_hitprocess.cc`

### *10.4.3. Background Rates*

A detailed study of the backgrond rates coming from beam interacting with the target was done to ensure that the DC occupancy stays within limits that do not affect the reconstruction efficency, typically below 3%.

Given the nominal operating luminosity $L = 10^{35} cm^{-2} s^{-1}$, and the liquid hydrogen target of $5cm$, the beam electron rates is $R = 4.7 \times 10^{11} Hz$. This correspond to around 124,000 electrons in the DC 250ns time window of region 1.

Various analyses [9], [12], [13], were performed using 124,000 electrons / event to study the DC occupancy response to variations of hardware position and beamline configurations. The results are summarized in Fig. 21.

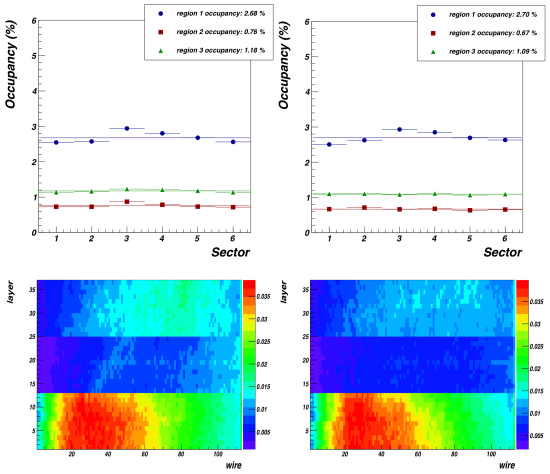



Figure 21: Results for DC rates for electron in-bending (left column) and out-bending (right column). Top: the occupancies are below 3% for region 1 and below 1.2% for region 3. Bottom: layer versus wire hit distribution: the two torus polarities show very similar distributions.

## 11. Low Threshold Cherenkov Counter

### *11.1. Geometry*

The LTCC mirror geometry is implemented through the native gemc geometry API. The elliptical mirrors are made through a subtraction of two G4Ellipsoid. The hyperbolic mirror are built using geant4 polycon approximating the mathematical shape using about 30 segments. The cylindrical mirrors are made of G4Tubs.

The LTCC winston cones are of three types: small, medium and large. Three CAD models are tessellated and imported in the simulation, and then are copied into 36 WC / sectors using the perl api.

Finally, the LTCC Box, mirror support structure and additional support hardware are imported directly from the engineering CAD models. The Fig. **??** shows details of the geometry implementation.

The refractive index of the $C_4F_10$ and its transparency is included in the material optical properties and taken into account during the geant4 transportation of the phtotons.

The mirror and winston cones reflectivities are associated with the mirror optical properties and are taken into account and taken into account during the geant4 transportation of the phtotons.

The quartz window PMT quantum efficiency are associated with the PMT face optical properties and are taken into account in the digitization routine.

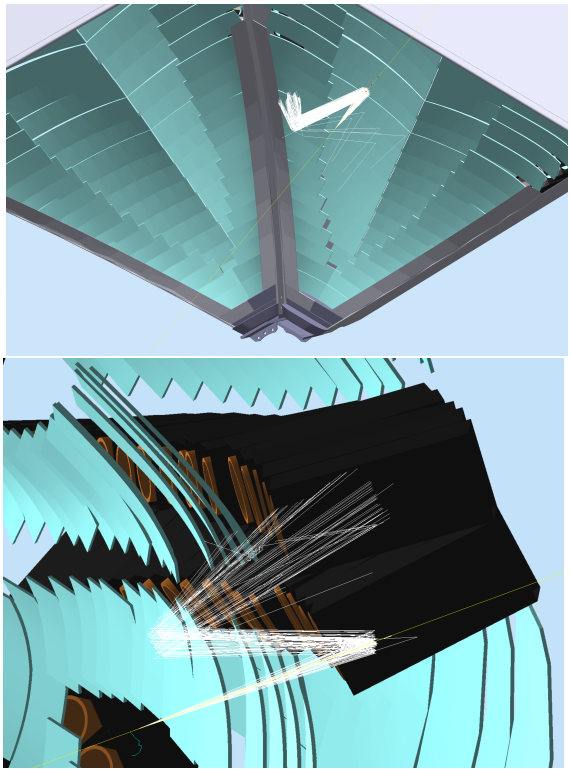


Figure 22: Top: a 6 GeV pion passing through the TCC gas volume and emitting Cherenkov photons. The light cone bounces from the elliptical to the hyperbolic to the winston cone and finally reaches the PMT. The frame of the LTCC, imported from the CAD engineering model, is visible. Bottom: details of the light path through the mirrors.

#### 11.1.1. Geometry Git Location

The github location of the gemc perl api script is `https://github.com/gemc/detectors/tree/master/clas12/ltcc`.

### 11.2. Digitization

Photons that impinged on the PMT faces are processed with the digitization routine. For each photon collected undergoes the quantum efficiency algorithm at its wavelength to decided if it's finally detected. The adc is calculated and smeared using the single photoelectron peak position and sigma from the calibration database.

#### 11.2.1. Timing

The time average of all the photons is saved in the output.

#### 11.2.2. Summary of CCDB Table used

- /calibration/ltcc/spe

### 11.3. Digitized Bank

The digitized output bank has $ID = 1400$, and the variables are summarized in Table 9

| Variable | Description | Tag |
|---|---|---|
| sector | clas12 sector | 1 |
| side | left or right index | 2 |
| segment | segment | 3 |
| adc | adc | 4 |
| time | average time of the hit | 5 |
| nphe | number of photoelectrons arrived | 6 |
| npheD | number of photoelectrons detected | 7 |
| hitn | hit number | 99 |

Table 9: The digitized LTCC bank

#### 11.3.1. Time Window

The timewindow of the LTCC is set to 5 ns.

#### 11.3.2. Process Routine Git Repository Location

The LTCC hit process routine location in git is `https://github.com/gemc/source/blob/master/hitprocess/clas12/ltcc_hitprocess.cc`

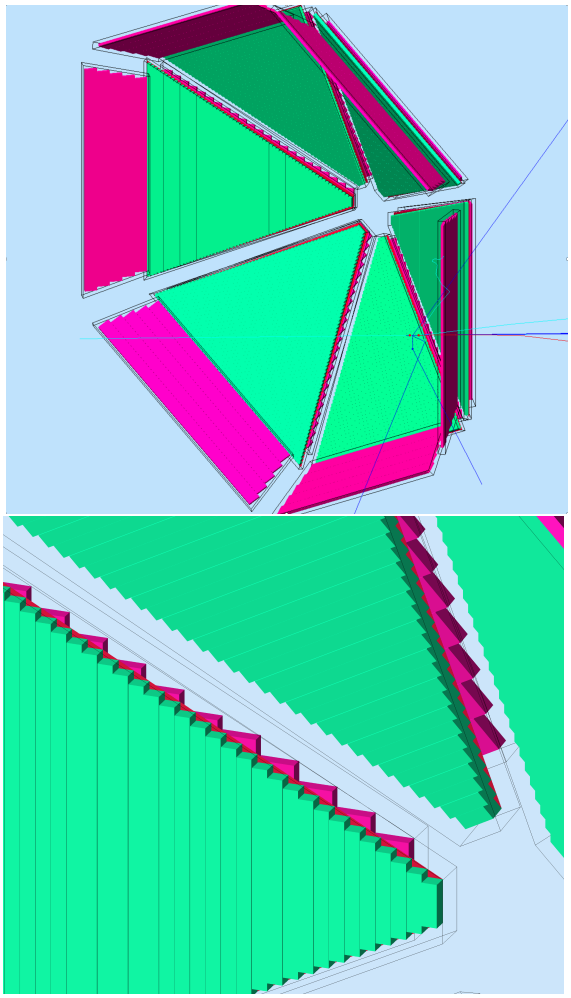


Figure 23: Top: the GEMC implementation of the FTOF geometry. The paddles are G4Boxes, embedded in trapezoid representing the mother volumes of each panel. Bottom: a zoom-in of the implementation shows the details of the individual paddles for Panel 1B (green) and Panel 1A (purple)

## 12. Forward Time-Of-Flight (FTOF)

### 12.1. Geometry

The FTOF geometry is implemented through the COATJAVA geometry service. The service provides the geant4 definitions that are read by the GEMC perl api to build the geometry database.

All scintillators are geant4 volumes. The paddles are assigned the scintillator material and associated with the ftof hit process routine. Each scintillator is a G4Box embedded in a G4Trapezoid mother volume made of air, see Fig. 23.

#### 12.1.1. Geometry Git Location

The github location of the gemc perl api script is `https://github.com/gemc/detectors/tree/master/clas12/ftof`. The geometry service definitions in coatjava is `https://github.com/JeffersonLab/clas12-offline-software/blob/development/common-tools/clas-jcsg/src/main/java/org/jlab/detector/geant4/v2/FTOFGeant4Factory.java`

### 12.2. Process ID

Each hit in the paddles is produces in two identical hits with the identifier variable "side" sets to 0 (for the left side PMT) and 1 (for right side PMT). The hits are then processed independently through the ftof hit process routine.

### 12.3. Digitization

#### 12.3.1. ADC

The energy deposited is reduced based on the position on the paddle using the calibration attenuation length. It is then corrected by a gain factor to account for the fact that the HV are adjusted so that the geometeric mean sqrt(L*R) is independent of counter length.

The corrected energy is converted to the theoretical number of photos $N_{th}$ using the constant $500\gamma/MeV$. A poissonian is used to calculate the actual number of photons $N_{actual}$ and the resulting "smeared" energy is the converted to adc using the FADC conversion factor.

An example of the smeared ADC for the right paddles as a function of position is shown in Fig. 24.

#### 12.3.2. TDC

The absolute hit time is corrected by:

- the effective velocity (from CCDB)
- the time walk correction, calculated from the smeared energy
- a panel to panel factor (from CCDB)
- a left/rigth time offset factor (from CCDB)
- an RF correction (from CCDB)

The time is then smeared by a $\sigma$ resolution read from CCDB using a gaussian function and then digitized using a TDC conversion factor.

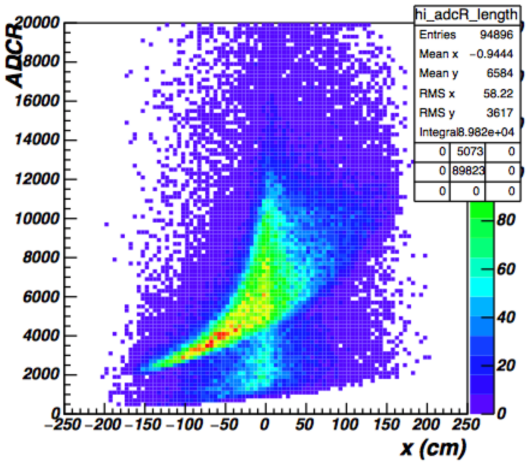



Figure 24: The ADC of the right paddles PMTs as a function of the relative paddle position of the hit in the paddle. The effects of attenuation length and smearing using realistic constant from the CCDB database makes the FTOF simulation response very similar to the real data.

#### 12.3.3. Summary of CCDB Table used

- /calibration/ftof/attenuation
- /calibration/ftof/effective_velocity
- /calibration/ftof/status
- /calibration/ftof/gain_balance
- /calibration/ftof/time_walk
- /calibration/ftof/time_offsets
- /calibration/ftof/tdc_conv
- /daq/tt/ftof

### 12.4. Digitized Bank

The digitized output bank has $ID = 1000$, and the variables are summarized in Table 10

#### 12.4.1. Time Window

The timewindow of the FTOF is set to 400 ns.

#### 12.4.2. Process Routine Git Repository Location

The FTOF hit process routine location in git is `https://github.com/gemc/source/blob/master/hitprocess/clas12/ftof_hitprocess.cc`

| Variable | Description | Tag |
|---|---|---|
| sector | sector number | 1 |
| layer | layer (1: 1A, 2: 1B, 3: 2B) | 2 |
| paddle | paddle number | 3 |
| side | PMT side (0 Left, 1 Right) | 4 |
| ADC | ADC | 5 |
| TDC | TDC | 6 |
| ADCu | ADC unsmeared | 7 |
| TDCu | TDC unsmeared | 8 |
| hitn | hit number | 99 |

Table 10: The digitized FTOF bank

## 13. Electromagnetic Calorimeter (EC) and preshower calorimeter (PCAL)

### *13.1. Geometry*

Both EC and PCAL calorimeters geometry is implemented through the COATJAVA geometry service. The service provides the geant4 definitions that are read by the GEMC perl api to build the geometry database.

All scintillators are geant4 volumes. The paddles are assigned the scintillator material and associated with the ftof hit process routine. Each scintillator is a G4Trap embedded in a G4Trapezoid mother volume made of air, see Fig. 26 and Fig. **??**.

#### *13.1.1. Geometry Git Location*

The github location of the gemc perl api script for ec is `https://github.com/gemc/detectors/tree/master/clas12/ec` and for pcal is `https://github.com/gemc/detectors/tree/master/clas12/pcal` The coatjava geometry service are `https://github.com/JeffersonLab/clas12-offline-software/blob/development/common-tools/clas-jcsg/src/main/java/org/jlab/detector/geant4/v2/ECGeant4Factory.java` and `https://github.com/JeffersonLab/clas12-offline-software/blob/development/common-tools/clas-jcsg/src/main/java/org/jlab/detector/geant4/v2/PCALGeant4Factory.java` for ec and pcal respectively.

### *13.2. Digitization*

The digitization is the same for both the EC and the PCAL calorimeters.

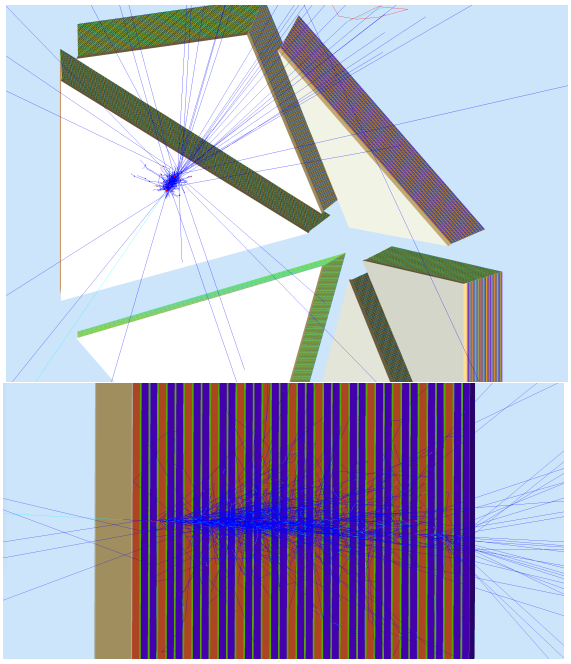


Figure 25: Top: a 4 GeV electron (cyan track) showering in the GEMC implementation of the EC geometry. The paddles are G4Boxes, embedded in trapezoid representing the mother volumes of each panel. The paddles layers are alternating with trapezoid made of lead, for a sampling fraction of about 0.3. Bottom: a zoom-in transverse view of electron shower.

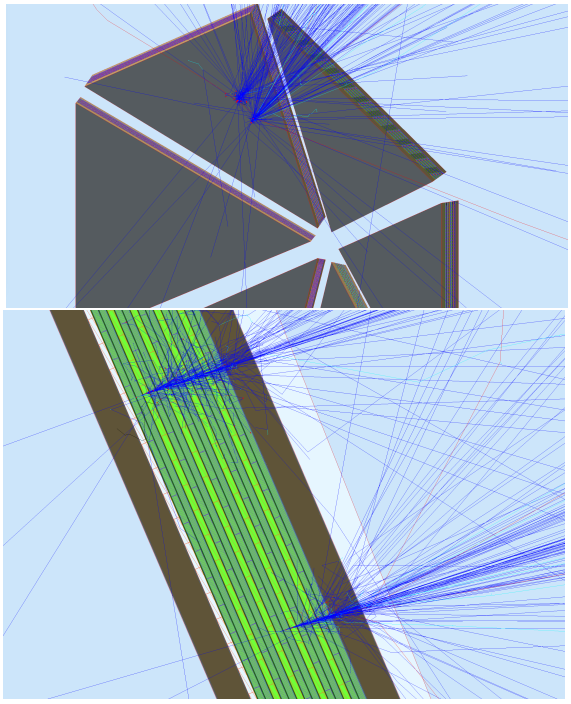


Figure 26: Top: a 4 GeV $\pi^0$ decayed in two photons hitting the GEMC implementation of the PCAL geometry. The paddles are G4Boxes, embedded in trapezoid representing the mother volumes of each panel. The paddles layers are alternating with trapezoid made of lead, for a sampling fraction of about 0.3. Bottom: a zoom-in transverse view of showers.

#### *13.2.1. ADC*

The energy deposited is reduced based on the position on the paddle using the calibration attenuation length. A number of photoelectrons is generated using a poissonian distributuion based on the attenuated energy. After a paddle resolution $\sigma_{res}$ is calculated from the calibration constants, fluctuations in PMT gain are taking into account using a Gaussian form with $\sigma_{res}$. A conversion factor is used to produce an ADC output.

#### *13.2.2. TDC*

The absolute hit time is corrected using the attenuation length and an additional factor accounts for the time-walk correction.

#### *13.2.3. Summary of CCDB Table used*

These CCDB tables are used for both EC and PCAL:

- /calibration/ec/gain
- /calibration/ec/attenuation
- /calibration/ec/timing
- /calibration/ec/effective_velocity
- /daq/tt/ec

### *13.3. Digitized Bank*

The EC digitized output bank has $ID = 1600$. The PCAL digitized output bank has $ID = 1500$.

For both systems the variables are summarized in Table 11

| Variable | Description | Tag |
|---|---|---|
| sector | sector number | 1 |
| stack | stack number | 2 |
| view | view | 3 |
| strip | strip number | 4 |
| ADC | ADC Left | 5 |
| TDC | TDC Right | 6 |
| hitn | hit number | 99 |

Table 11: The digitized EC bank

#### *13.3.1. Time Window*

The timewindow of both PCAL and EC is set to 400 ns.

#### 13.3.2. Process Routine Git Repository Location

The calorimeter hit process routines are `https://github.com/gemc/source/blob/master/hitprocess/clas12/ec_hitprocess.cc` and `https://github.com/gemc/source/blob/master/hitprocess/clas12/pcal_hitprocess.cc`.

## 14. Solenoid and Torus Magnets

### 14.1. Geometry

The solenoid geometry is produced with the GEMC perl api. The solenoid is a single polycone volume, shown in Fig. 27 in a section view with the target and an 11 GeV electron beam.

The torus geometry is imported from the engineering CAD model through 54 tessellated volumes. Among the volumes:

- the bore heat shield and hub components
- the back and front plants
- the stainless steel coil frames
- the copper coils
- internal shielding around the hub

The torus hub is protected from the beampipe background with additional tungsten shielding. The torus geometry is shown in Fig. 28.

### 14.2. Magnetic Field Maps

The field maps are imported in the simulation using ascii files. Both fields can be scaled by an arbitrary number. Both fields are defined in the Hall-B coordinate system and both can be shifted and tilted by additional delta amounts.

#### 14.2.1. Solenoid

The solenoid field map has cylindrical symmetry around the z- axis, so the map is defined in the transverse / longitudinal plane and then rotated when requested by the geant4 navigation. The integration method used in the simulation is the Range Kutta 4.

The solenoid field near the target points along the z direction and its uniformity is $\Delta B/B < 10^{-4}$.

The field map has 1200 points along the z-axis, from -3m to 3m. It has 600 points in the transverse coordinate, from 0 to 3m. The field grid values are linearly interpolated to the (x,y,z) coordinate requested by geant4.

Table 12 shows the field map ascii data structure.

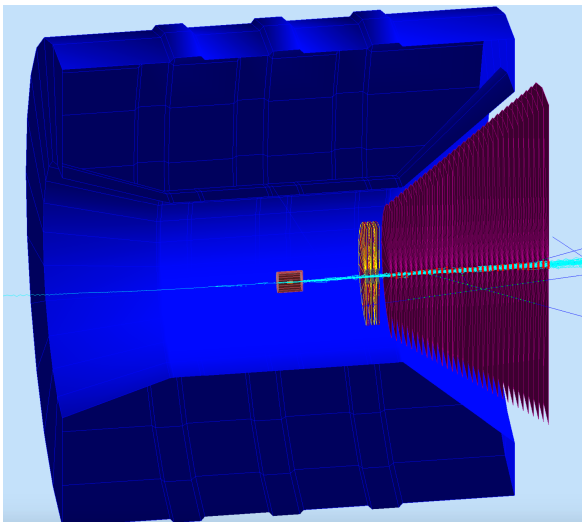


Figure 27: An 11 GeV electron beam is impinging on a 5 cm liquid hydrogen target. At the full $10^{35} cm{-}2s{-}1$ luminosity, this correspond to 124,000 electrons in a 250 ns window. Normally this would produce a storm of moeller electrons that would saturate any detector in the forward or central region. However the solenoid focus all of them along the beam line, providing the most effective shield to CLAS12.

| T (m) | Z (m) | $B_T$ | $B_L$ |
|---|---|---|---|
| 0.005 | -0.025 | 0.000013 | 5.000880 |
| 0.005 | -0.020 | 0.000044 | 5.000822 |
| 0.005 | -0.015 | 0.000073 | 5.000704 |
| 0.005 | -0.010 | 0.000101 | 5.000529 |
| 0.010 | -0.025 | 0.000028 | 5.000928 |
| 0.010 | -0.020 | 0.000089 | 5.000867 |
| 0.010 | -0.015 | 0.000148 | 5.000747 |
| 0.010 | -0.010 | 0.000203 | 5.000570 |

Table 12: Solenoid ascii field map values around the target. The field values are in Tesla. $T = \sqrt{(X^2 + Z^2)}$

#### 14.2.2. Torus

The torus field can be imported using a synmetric map or a full 3D map. The symmetric map is defined in half the CLAS12 sector. It is symmetric around the sector midplane and copied in each sector when requested by the geant4 navigation. The 3D map covers the entire cartesian space and accounts for field deviations due to coil movements or imperfections.

The field map has 251 points along the z-axis, from 1m to 3m. It has 2501 points in the transverse coordinate, from 0 to 5m. It has 16 azimuthal points from 0 to $30^0$. The field grid values are linearly interpolated to the (x,y,z) coordinate requested by geant4.

The torus field in mid sector is perpendicular to the z-axis and is typically 2.5 tesla. Table 13 shows the field

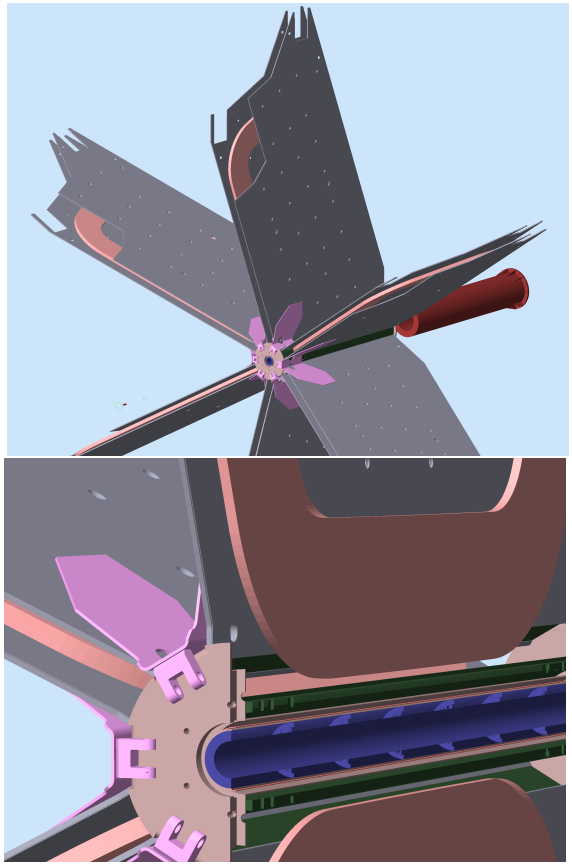


Figure 28: Top: the GEMC implementation of the torus hardware. The volumes are imported from the CAD engineering model. The stainless steel frame embeds the copper coils, immersed in the cold hub tube. Bottom: a section view of the torus in the vicinity of the beamline. The warm and cold hub are visible, along with the tungsten shielding (blue colors).

map ascii data structure.

| $\phi$ (deg) | T (m) | Z (m) | $B_x$ | $B_y$ | $B_z$ |
|---|---|---|---|---|---|
| 0.0 | 190.0 | 338.0 | 0 | 4.51275 | 0 |
| 0.0 | 190.0 | 340.0 | 0 | 4.50136 | 0 |
| 0.0 | 190.0 | 342.0 | 0 | 4.48789 | 0 |
| 0.0 | 190.0 | 344.0 | 0 | 4.47235 | 0 |
| 0.0 | 190.0 | 346.0 | 0 | 4.45472 | 0 |
| 0.0 | 190.0 | 348.0 | 0 | 4.43502 | 0 |
| 0.0 | 190.0 | 350.0 | 0 | 4.41323 | 0 |
| 0.0 | 190.0 | 352.0 | 0 | 4.38935 | 0 |

Table 13: Torus ascii field map values near mid-sector. The field values are in kGauss. $T = \sqrt{(X^2 + Z^2)}$

#### *14.2.3. Geometry Git Location*

The github location of the gemc perl api script for the solenoid and the torus CAD volumes is `https://github.com/gemc/detectors/tree/master/clas12/magnets`.

## 15. Beamline

The CLAS12 beamline is made up of several pieces, each discussed below. The positioning and composition of the beamline depends on the run configuration, that can be:

- FTOn: Forward Tagger present and operational. The Moller shield starts at z=877 mm from the target center to.
- FTOff: FT is present but not operational. The FT tracker is replaced by shielding. The Moller shield starts at z=430 mm from the target center, and additional shielding is present to connect it to the FT.

### *15.1. Vacuum pipe*

A stainless steel vacuum pipes that contain the eletron beam. The pipe starts downstream of the target at $z = 80cm$ and changes dimensions inside the torus and downstream of the torus as detailed in Table 14

### *15.2. Moeller Shielding*

The Moeller shielding is composed by the following elements, shown in Fig. 30 for the FTOn configuration and in Fig. 31 for the FTOff configuration.

- FT On and FT Off configurations:

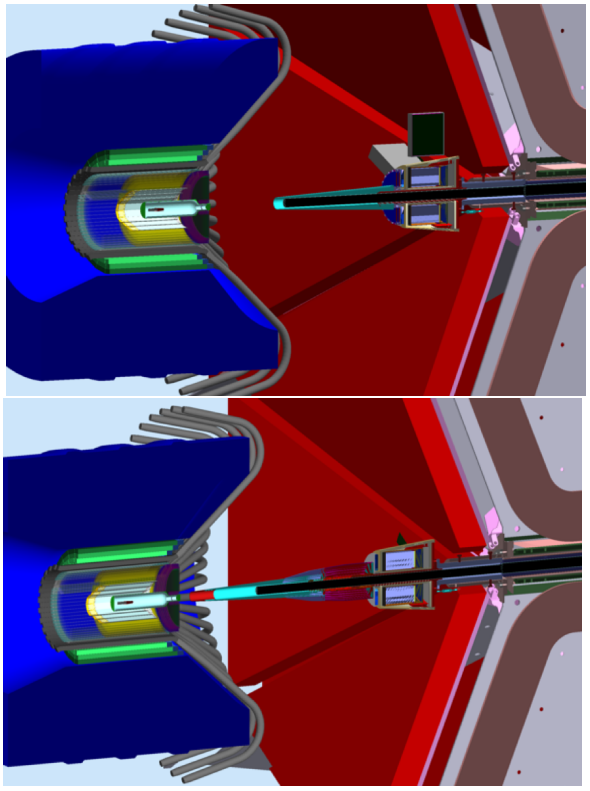


Figure 29: The two possible CLAS12 configurations. Top: FTOn. To clear its acceptance at forward angles (2.50 – 4.50 degrees) the Moller shield (cyan color) is attached to the FT tracker, starting at $z$ = 877 mm from the target. Bottom: FToff; the FT is present but not operational. The FT tracker is replaced with a shield. The Moller cone is placed at $z$ = 430 mm from the target and additional shielding minimize background in Region 1 Drift Chambers.

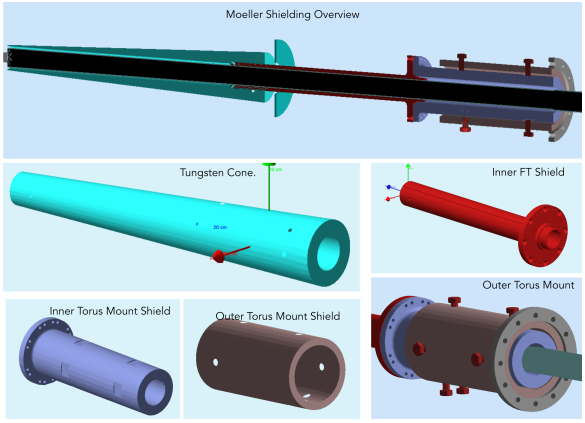



Figure 30: The moeller shielding for the FT On configuration. Top: section of the overall overview of the cone, FT support and torus mount. Various individual components are shown: the tungsten cone, the inner FT shield, and the structure of the torus mount.

| | Thickness (mm) | Inner Radius (mm) |
|---|---|---|
| Upstream | 1.6 | 26.9 |
| Inside | 1.6 | 33.3 |
| Downstream | 3.2 | 60.3 |

Table 14: The vacuum pipe three dimensions sets (in mm) upstream, inside and downstream of the torus

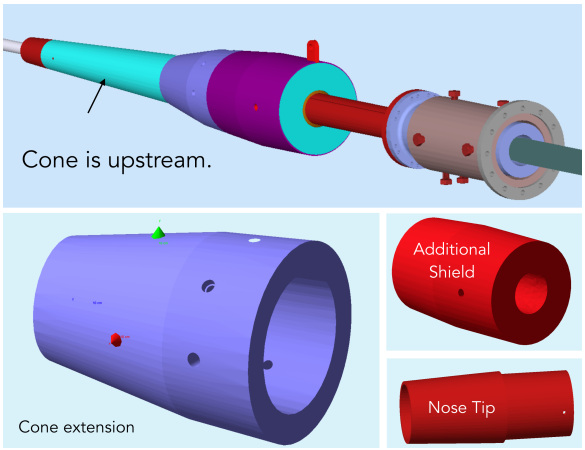



Figure 31: The moeller shielding for the FT Off configuration. Top: section of the overall overview of the cone, FT support and torus mount. the cone tip extension and the additional shielding that replace the FT tracker is also shown.

  - a Tungsten cone with increasing thickness.
  - a tungsten pipe and flange inside the FT
  - a structure to mount the FT and the shielding onto the torus frame, composed by:
    * an inner stainless steel shield and flange
    * an outer tungsten shield
    * nine copper screw to adjust the alignment of the FT and shields upstream of the torus

- Only FT Off configuration:
  - a Tungsten cone tip to extend the moeller shield cone
  - additional lead structure to replace the FT tracker with shielding

### *15.3. Torus and downstream Shielding*

Additional shielding is placed around the vacuum pipe inside the torus hub in the form of tungsten bricks. Downstream of the torus a shielding downstream of the torus in the form of a connecting tungsten nose and a long lead pipe.

These components are shown in Fig. 32.

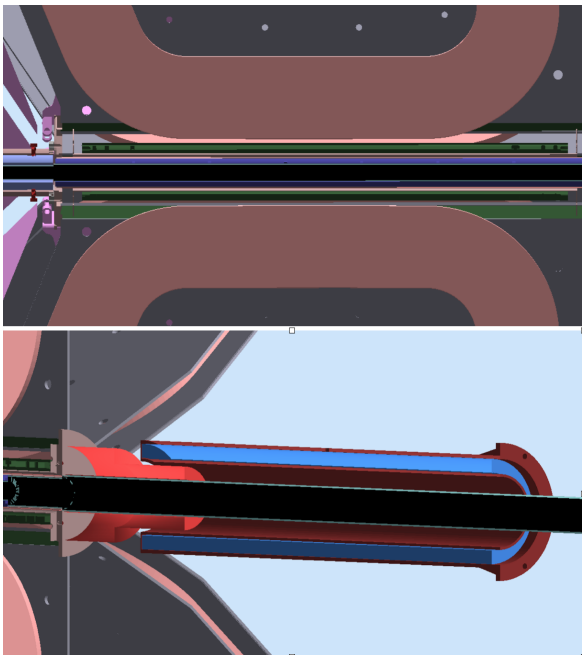


Figure 32: Torus and Downstream shielding.

### *15.4. Geometry Source*

The beamline geometry is entirely imported from the engineering CAD model.

#### *15.4.1. Geometry Git Location*

The github location of the gemc geometry CAD volumes is `https://github.com/gemc/detectors/tree/master/clas12/cadBeamline`.

## 16. Summary of geometry and digitization

## 17. Signals and Readout

simulations Signals and Readout description
EVIO ADC FADC evio2root for some banks

## 18. Performance

The performance of the CLAS12 simulations is measured by comparing the predicted background rates from beam interactions with the target with the actual experiment rates. The banchmarks are also quantified for each detector geometry and digitization routine.

### *18.1. Comparison of rates with data*

On December 17 2017, the nominal luminosity of $10^{35}s{-}1cm{-}2$ (75 nA on a 5 cm liquid hydrogen target) was achieved in CLAS12. The rates in various detectors were measured.

The Drift Chamber occupancy was compared for both FTon and FTOff configurations. The integrated occupancy in region 1, 2 and 3 are summarized in tables 15 and 16. The time window of the experiment at the time was different than what was in the simulation, so the data has been scaled accordingly. The predicted rates agree quite well with the measured ones. The occupancy distribution in the different layers and wires also show good agreement between simulation and experiment, see Fig. 33 and Fig. 34.

| Region | Data (rescaled) | GEMC |
|---|---|---|
| 1 | 2.8% | 2.68% |
| 2 | 0.6% | 0.76% |
| 3 | 1.5% | 1.18% |

Table 15: Drift Chambers rates comparison between simulation and real data for the FT On configuration

| Region | Data (rescaled) | GEMC |
|---|---|---|
| 1 | 1.7% | 1.12% |
| 2 | 0.3% | 0.44% |
| 3 | 0.9% | 0.73% |

Table 16: Drift Chambers rates comparison between simulation and real data for the FT Off configuration

Simlar agreements were found in rates in other CLAS12 detectors: HTCC, EC and PCAL, FTOF and FT.

### *18.2. Benchmarks*

The event rate has been measured in a single core for single and multiple tracks. The numbers reported here refer to averages over several 2017 laptop and desktops and farm nodes.

The full CLAS12 geometry has been used, with Range Kutta field integration algorithm and linear interpolation for both solenoid and torus fields.

Single meson tracks in the forward region are simulated with an event rate of about 10 Hz. Electron rates are about half as fast due to shower in the EC and PCAL calorimeters and Cherenkov photons production in both HTCC and LTCC, for an average of 5 Hz.

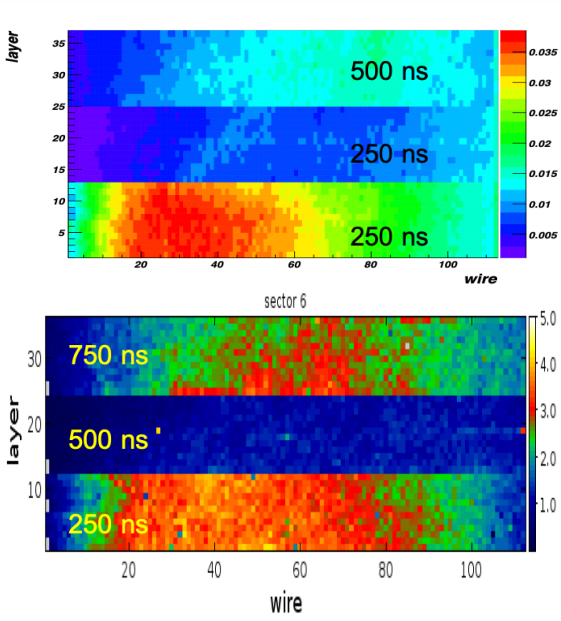



Figure 33: Distribution of background events in the Drift Chamber region 1 for the FT On configuration. Top: gemc. Bottom: data.

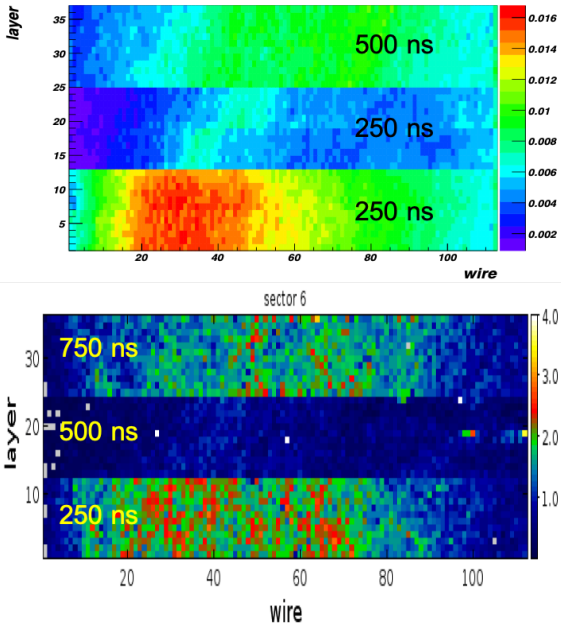



Figure 34: Distribution of background events in the Drift Chamber region 1 for the FT Off configuration. Top: gemc. Bottom: data.

A quantitative study of the event rate for 3 particles (2 in the forward detector, one in the central) details the time spend in each detector geometry and digitization and in each magnetic field. The particles generated are:

- one 7 GeV electron between 15 and 25 degrees in theta.
- one 2 GeV gamma between 15 and 25 degrees in theta
- one 2 GeV proton at 90 degrees in theta

The results are shown in table 17. The final rate for the 3 particles in the complete CLAS12 setup is 1.9 Hz.

| | Rate (Hz) | ms / event | % of total |
|---|---|---|---|
| Target | 555.8 | 1.8 | 0.3 |
| SVT | 365.9 | 0.9 | 0.2 |
| MM | 48.9 | 17.7 | 3.4 |
| CTOF | 48.7 | 0.1 | 0.0 |
| Solenoid | 15.8 | 42.8 | 8.1 |
| HTCC | 11.5 | 24.0 | 4.5 |
| FT | 11.2 | 2.4 | 0.4 |
| DC | 8.1 | 33.2 | 6.3 |
| LTCC | 6.5 | 30.6 | 5.8 |
| FTOF | 6.1 | 9.6 | 1.8 |
| PCAL | 2.8 | 196.2 | 37.1 |
| EC | 2.0 | 134.5 | 25.4 |
| Torus | 1.9 | 34.7 | 6.6 |

Table 17: Drift Chambers rates comparison between simulation and real data for the FT Off configuration

Background rate can be estimated by setting a time window to define the time length of one event, and generating a number of electrons proportional to the proposed beam current. For the nominal luminosity of $10^{35}s{-}1cm{-}2$ this correspond to 124,000 electrons in a 250 ns time window. In this case the time to complete one event varies between one and two minutes depending on CPU flavor and available memory.

## 19. Distribution and Documentation

The gemc framework is documented on the gemc website [1]. This includes the latest news and releases, examples, procedure details and all options documentation.

The software is distributed in two ways: with docker, or by downloading the source from the its public repository.

#### 19.0.1. Docker

A docker container with the necessary libraries to run GEMC and the reconstruction software is created in the JeffersonLab hub repository [14]

The container is tagged, and every tag contains a set version of these libraries:

- event generators such as generate-dis, dvcsgen generator executables
- gemc with the clas12 geometry
- CLARA
- Coatjava

Collaborators access these containers and all the software inside by using the command:

```
docker run -it --rm jeffersonlab/clas12simulations:prod bash
```

The only requirement is the docker app, avaialble in Windows, Mac and Linux OS flavors.

#### 19.0.2. Source Code Dowload

The code repository is `https://github.com/gemc/source`. To compile GEMC several libraries are needed:

- clhep: Class Library for High Energy Physics [15]
- xercesc: validating XML parser [16]
- geant4: the libraries to simulate the passage of particles through matter [2]
- qt: a C++ graphic library [17]
- evio: the CLAS12 data format [6]
- CCDB: the calibration database based on mysql [5]

#### 19.0.3. Documentation and code repository

The main documentation is on the gemc website [1]. It includes example on how to create and run custom geometry and use various features like generators, geometry factories, hit definitions, output, etc.

The code is kept on the guthub repository `https://github.com/gemc/source`. Gemc is released on a semi-annual cycle.

#### 19.0.4. Clas12 Tags

The CLAS12 development of both hit process routines and geometry is kept in a separate repository, `https://github.com/gemc/clas12Tags`, that also contains documentation on how to run with configurations corresponding to the various experiment, how to change the beamline configuration, and switch between code and geometry releases.

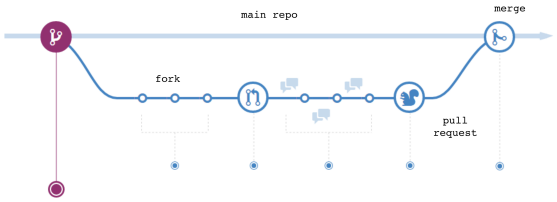



Figure 35: Typical workflow of contribution to the code: the main repository is forked and worked on. During the process comments can be made part of the development process. When the users make the pull request, the changes are validated and then merged to the master repository, or sent back with comments if there were problems.

#### 19.0.5. Contribution to code and geometry

The code repository is public. The development contributions mechanism is illustrated in Fig. 35: collaborators fork the repository and make pull request that are validated by the code author.

## 20. Acknowledgements

We appreciate the work of the Geant4 collaboration.

## 21. Conclusion

## 22. References

[1] gemc.
URL `https://gemc.jlab.org`

[2] S.Agostinelli, et al., Geant4a simulation toolkit, Nuclear Instruments and Methods in Physics Research 506 (3) (2003) 250–303. `doi:10.1016/S0168-9002(03)01368-8`.

[3] Step-file.
URL `https://en.wikipedia.org/wiki/ISO_10303-21`

[4] freecad.
URL `https://www.freecadweb.org`

[5] ccdb.
URL `https://github.com/JeffersonLab/ccdb/wiki`

[6] evio.
URL `https://coda.jlab.org/drupal/content/event-io-evio`

[7] I. Antcheva, et al., ROOT: A C++ framework for petabyte data storage, statistical analysis and visualization, Comput. Phys. Commun. 182 (2011) 1384–1385. `doi:10.1016/j.cpc.2011.02.008`.

[8] M. Ungaro, Corrections to clas12 target design (2017).
URL `https://misportal.jlab.org/mis/physics/clas12/viewFile.cfm/2017-013.pdf?documentId=48`

[9] V. Burkert, S. Stepanyan, J. Tan, M. Ungaro, Beam position study on a 3.5 mm shifted target cell. (2017).
URL `https://misportal.jlab.org/mis/physics/clas12/viewFile.cfm/2017-018.pdf?documentId=54`

[10] V. Burkert, Y. Gotra, M. Ungaro, Study of tungsten shielding around the target. (2018).
URL `https://misportal.jlab.org/mis/physics/clas12/viewFile.cfm/2018-005.pdf?documentId=59`

[11] R. D. Vita, M. Ungaro, Importing clas12 cad models of target and beamline in the gemc simulation. (2017).
URL https://misportal.jlab.org/mis/physics/clas12/viewFile.cfm/2017-017.pdf?documentId=53

[12] R. D. Vita, L. E. A. Kim, R. Miller, S. Stepanyan, J. A. Tan., M. Ungaro, C. Wiggins, M. Zarecky, Corrections to clas12 vacuum beamline (2017).
URL https://misportal.jlab.org/mis/physics/clas12/viewFile.cfm/2017-012.pdf?documentId=47

[13] R. D. Vita, D. S. Carman, C. Smith, S. Stepanyan, M. Ungaro, Study of the electromagnetic background rates in clas12 (2017).
URL https://misportal.jlab.org/mis/physics/clas12/viewFile.cfm/2017-016.pdf?documentId=52

[14] jlabdocker.
URL https://cloud.docker.com/u/jeffersonlab

[15] clhep.
URL http://proj-clhep.web.cern.ch/proj-clhep

[16] xercesc.
URL https://xerces.apache.org/xerces-c

[17] qt.
URL https://www.qt.io