Status of JUPITER and Satellites

$Kotoyo\ Hoshina,\ ACFA-SIM\ \&JLC-CDC$

1 Introduction

Experiments at a future linear e^+e^- collider such as JLC[1] will open up a novel possibility to reconstruct all the final states in terms of fundamental particles (leptons, quarks, and gauge bosons). This involves identification of heavy unstable particles such as W, Z, and t through jet invariant-mass measurements. High resolution energy flow measurements will thus be crucial, necessitating high resolution tracking and calorimetry as well as good track-cluster matching to avoid double counting. A large cylindrical drift chamber with small jet cells (JLC-CDC[2]) is our choice for a candidate central tracking device to fulfill these requirements. Good track-cluster matching requires, however, small track extrapolation errors, which in turn demand high r-ϕ and z resolutions. The latter forces us to introduce cells consisting of stereo wires, since the z resolution in charge division or time difference readout is typically 1 % of the wire length or worse, which is a few cm in the linear collider use.

We have already studied and published hardware aspects of common problems in designing stereo-wire geometry for a long cylindrical drift chamber with small jet cells[6]. In order to finalize the chamber design so as to achieve the best attainable energy flow resolution, however, we need to carefully optimize the layout of axial and stereo cells through detailed Monte Carlo simulations. Considering the recent advance of object-oriented technology in high energy physics software development, we had thus started the development of a full Monte Carlo simulator called JUPITER[3] based on Geant4[7] and an event reconstruction simulator called Satellites[3] based on ROOT[4]/JSF[5]. The philosophies, structures, and main features of these simulators were presented at APPI2001 and APPI2002.

The CDC in JUPITER that we reported in the past APPI meetings did not have any stereo layers implemented, since there had been and still is no official Geant4 solid available to describe stereo cells. This is in contrast with the situation with axial layers for which Geant4 provides a standard solid called `G4Tubs`, which is a ϕ segment of a cylinder. We have thus extended Geant4 to include a new solid (`TwistedTubs`), which comprises three kinds of bounding surfaces: two end planes, inner and outer hyperboloidal surfaces, and two so-called twisted surfaces that make slant and twisted ϕ boundaries. The first half of this talk will thus be devoted to descriptions of this new solid[1].

Using this new class, we installed stereo cells into JUPITER and studied the effect of the stere cells on the CDC's time stamping capability. Since the maximum drift time of the CDC will be $7 \simeq \mu$sec, which is much longer than the bunch train length of 192×1.4 nsec, all the background tracks from these surrounding bunches would be recorded as signal event trajectories without any time-stamping capability. The JLC-CDC is equipped with cells staggered from one layer to another, which provides a time-stamping system. The past estimate of the time-stamping capability assumed only axial layers and how the existence of stereo cells would affects it has been an open question. In the latter half of

[1]Although `TwistedTubs` was developed under the JUPITER environment, it is actually a general purpose Geant4 extension. We thus wrote a CPC paper[8] which describes the design philosophy of `TwistedTubs`, its realization in the Geant4 framework, algorithmic details, and results of its performance test. If you are interested in technical details, see [8]

this talk, preliminary results will be shown together with discussions on their implications.

2 Geometry of Stereo Mini-jet Cell

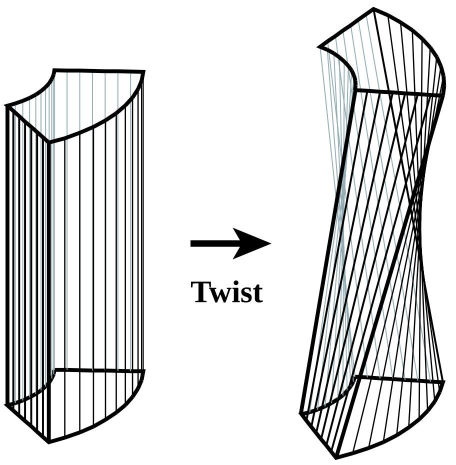


Figure 1: An exaggerated illustration of a stereo cell as formed by twisting an axial cell.

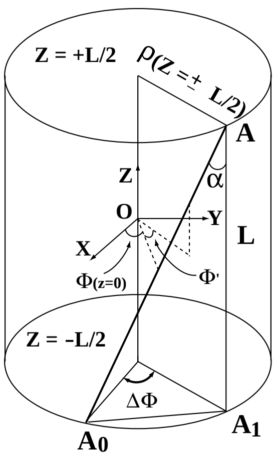


Figure 2: 3-dimensional view of a single stereo wire.

In this section we introduce, following the convention of [6], the stereo-geometrical parameters that will be needed in subsequent sections.

Consider a cylindrical tube consisting of two cylindrical layers of axial field-shaping wires strung across two disc-shaped end plates at some radii, ρ_{in} and ρ_{out}. An axial mini-jet cell is a segment cut out from this cylindrical tube by ϕ boundaries formed by cathode wires. As illustrated in Fig. 1, twisting one end plate by a twist angle $\Delta\phi$ with the other end fixed turns the axial mini-jet cell into a stereo mini-jet cell. Inspection of the figure tells us that the stereo cell consists of three pairs of three kinds of surfaces: a pair of hyperboloidal surfaces setting inner and outer radial boundaries, another pair of so-called twisted surfaces making left and right azimuthal boundaries, and yet another pair of fan-shaped flat surfaces closing the positive and negative z ends of the cell.

By construction, any of the four side walls of the stereo cell can be regarded as a locus of a stereo wire sweeping through the surface. The geometry of a given side surface can thus be completely determined by the equation for a single representative wire chosen from the stereo wires forming that surface. As depicted in Fig. 2, the representative stereo wire is uniquely specified by the radius at the ends $\rho(z = \pm L/2)$ or that at the center $\rho_c \equiv \rho(z = 0)$, the projected wire length L to the chamber axis, and the twist angle $\Delta\phi$. Notice that $\Delta\phi$ is signed and measured from A_0 to A_1. The stereo angle α, which is defined as an angle between AA_0 and AA_1, is also signed, having the same sign as $\Delta\phi$.

The stereo angle α can now be written in terms of $\rho(z = \pm L/2)$, $\Delta\phi$, and the projected wire length (L):

$$\alpha \;=\; \tan^{-1}\left(\frac{2\rho(z=\pm L/2)}{L}\sin\left(\frac{\Delta\phi}{2}\right)\right) \tag{2.1}$$

It is obvious from Fig. 2 that both the azimuthal angle and the radial position of the stereo wire become z-dependent:

$$\begin{aligned}\phi(z) &= \phi(z=0)+\phi' \\ &= \phi(z=0)+\tan^{-1}\left[\left(\frac{2z}{L}\right)\tan\left(\frac{\Delta\phi}{2}\right)\right] \end{aligned} \tag{2.2}$$

$$\begin{aligned}\rho(z) &= \sqrt{(\rho(z=0))^2+(z\tan\alpha)^2} \\ &= \sqrt{(\rho(z=\pm L/2))^2+(z^2-(L/2)^2)\tan^2\alpha}, \end{aligned} \tag{2.3}$$

where z is measured from the middle of the chamber along the chamber axis as Fig. 3 indicates.

Now we consider a hyperboloidal surface obtained as a locus of the straight line given by Eqs. 2.2 and 2.3 by sweeping $\phi(z=0)$ from ϕ_{left} to ϕ_{right} with $\rho(z=0)$ fixed. Let the outward normal to the hyperboloidal surface at a surface point $\mathbf{x}=(x,y,z)$ be $\mathbf{n}$. Apparently the outward normal $\mathbf{n}$ is in the ρ-z plane containing $\mathbf{x}$ and is perpendicular to the tangential vector thereat. Eq. 2.3 tells us that the cross section of the hyperboloidal surface by the ρ-z plane becomes a hyperbola given by

$$\begin{cases}\rho &= \sqrt{(\rho(z=0))^2+(z\tan\alpha)^2} \\ z &= z.\end{cases}$$

The tangential vector we need is thus obtained by differentiating this equation with respect to z. Normalizing it to $|\mathbf{x}|$, we obtain

$$\text{the tangential vector} \;=\; (z\tan^2\alpha,\ \rho), \tag{2.4}$$

which leads to the outward normal in the ρ-z plane: $\mathbf{n}_\rho=(\rho,-z\tan^2\alpha)$. Recalling that ρ in the ρ-z plane corresponds to (x,y) in the x-y plane, we now obtain

$$\mathbf{n}=(x,y,-z\ \tan^2\alpha). \tag{2.5}$$

The inward normal is anti-parallel with this and can be obtained by simply changing the signs of all the components.

On the contrary to the hyperboloidal surface, a twisted surface is a locus of the straight line given by Eqs. 2.2 and 2.3, when $\rho_c \equiv \rho(z=0)$ is swept from $\rho_{c,in} \equiv \sqrt{\rho_{in}^2-((L/2)\tan\alpha)^2}$ to $\rho_{c,out} \equiv \sqrt{\rho_{out}^2-((L/2)\tan\alpha)^2}$ with $\phi(z=0)$ fixed, where ρ_{in}

and ρ_{out} are the inner and outer radii at the end planes as defined before. By construction, any point on such a twisted surface can be specified by two parameters (ρ_c and z) in a local coordinate system for which x-axis is chosen such that $\phi(z=0)=0$, as depicted in Fig. 4:

$$\left\{\begin{array}{ccccc} x & = & \rho_c & & \\ y & = & \rho_c \tan\phi' & = & \rho_c\,\kappa\,z \\ z & = & z, & & \end{array}\right. \tag{2.6}$$

where $\tan\phi'$ is given, through Eq. 2.2, by

$$\tan\phi' \quad = \quad \frac{2z}{L}\cdot\tan\left(\frac{\Delta\phi}{2}\right) \tag{2.7}$$

while κ is defined to be

$$\kappa \quad \equiv \quad \frac{2}{L}\cdot\tan\left(\frac{\Delta\phi}{2}\right). \tag{2.8}$$

Notice that the $\Delta\phi$ must be smaller than π, implying that $\rho_c > 0$ in this coordinate system. Eq. 2.6 shows that, in addition to the lines corresponding to cathode wires, the twisted surface contains another kind of straight lines that appear as the sections of different z-slices of the twisted surface. Each of these lines corresponds to a ϕ boundary of a z-slice of a stereo cell and passes through the z-axis. At the point $\mathbf{x}$ on the surface, a pair of these two kinds of straight lines passing through it divides the tangential directions at that point into four regions: the surface turns from convex to concave or vice versa as crossing the boundaries. In other words, the twisted surface has a saddle shape[2].

The two kinds of straight lines contained in the twisted surface are useful when we construct a normal to the twisted surface as we shall see below. In general a normal to the surface at a given surface point $\mathbf{x}=(x,y,z)$ can be formed as a vector product of two linearly independent tangential vectors at that point. In our case such tangential vectors can be readily obtained by partially differentiating Eq. 2.6 with respect to $x=\rho_c$ and z:

$$\begin{aligned} \mathbf{e}_z &\equiv \begin{pmatrix} \frac{\partial x}{\partial z} \\ \frac{\partial y}{\partial z} \\ \frac{\partial z}{\partial z} \end{pmatrix} = \begin{pmatrix} 0 \\ \rho_c\kappa \\ 1 \end{pmatrix} \\ \mathbf{e}_x &\equiv \begin{pmatrix} \frac{\partial x}{\partial x} \\ \frac{\partial y}{\partial x} \\ \frac{\partial z}{\partial x} \end{pmatrix} = \begin{pmatrix} 1 \\ \kappa z \\ 0 \end{pmatrix}. \end{aligned} \tag{2.9}$$

The first of the above two tangential vectors, $\mathbf{e}_z$, can virtually be identified as one of the cathode wires forming the twisted surface. On the other hand, the second one, $\mathbf{e}_x$ can be viewed as a ϕ boundary of a z-slice of a stereo cell. Using Eq. 2.9, we can now easily

[2]It is well known that the hyperboloidal surface has also a saddle shape. In this case, the two straight lines over which the curvature changes its sign are in the directions of stereo angles of $\pm\alpha$.

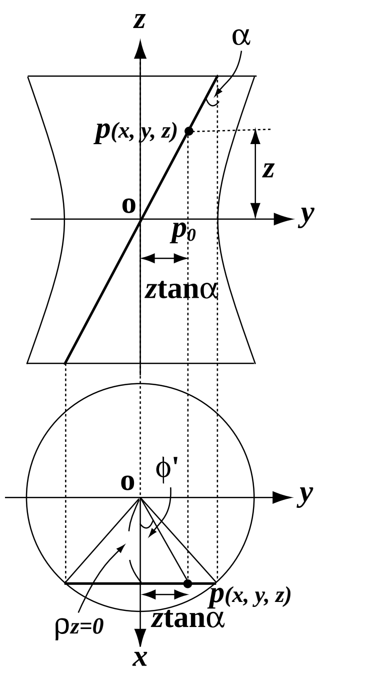


Figure 3: Hyperboloidal surface and a typical line (wire) contained in it.

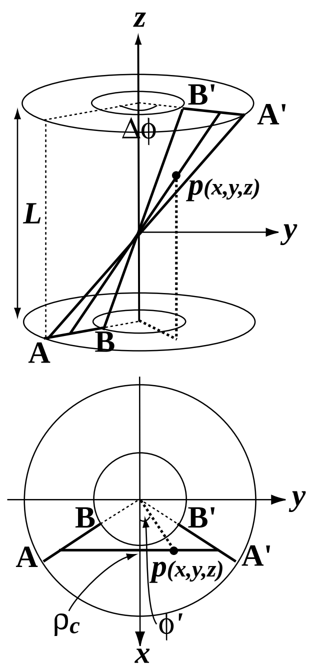


Figure 4: Similar picture to Fig. 3 for a twisted surface.

form the surface normal at the surface point **x** by taking the vector product of these two tangential vectors.

Armed with the equations given in this section, we shall, in the next section, prepare classes to represent the three kinds of surfaces: `J4HyperboloidalSurface`s, `J4TwistedSurface`s, and `J4FlatSurface`s, and assemble them to form a Geant4 solid class (`J4TwistedTubs`) to describe our stereo mini-jet cells of JLC-CDC.

3 Design of J4TwistedTubs

Before coding our new solid "`J4TwistedTubs`", we set the following guide line to fulfill the required functionality:

1. `J4TwistedTubs` should be implemented as a collection of bounding surfaces. It will thus have to have, as its data member, an array of pointers to the instances of the corresponding surface classes.

2. All of these surface classes must inherit from an abstract class named "`J4VSurface`", which carries generic information on each surface and sets basic interfaces to all the surface classes that inherit from it.

3. Each surface is orientable and should know which side is outside, but the calculation of the distance to the surface from a point `p` should not depend on whether the point is inside or outside of the solid, since the distance should only depend on the shape of the surface in question.

4. It is then the role of "`DistanceToIn`" or "`DistanceToOut`" methods of its parent `J4VSurface` class to judge the distance depending on the context such as the angle between the surface normal and the velocity vector `v`. This way, we can avoid repetition of the same code.

5. All `J4TwistedTubs` has to do will then be to simply invoke "`DistanceToIn`" or "`DistanceToOut`" for each of the bounding surfaces and choose the best[3]. The surface normal can then be calculated by calling the corresponding `GetNormal` method for the selected surface.

As described in last section, `J4TwistedTubs` consists of three pairs of three kinds of surfaces. We named these three kinds of bounding surfaces `J4HyperboloidalSurface`, `J4TwistedSurface`, and `J4FlatSurface`, all of which are descendants of `J4VSurface`. Interrelation of these classes is illustrated in Fig. 5 as a class diagram. On the other hand, Fig. 6 shows the outward direction of each surface as stored in a data member called `fOrientation`.

As sketched above, the abstract base class `J4VSurface` has the following methods:

[3]The best is usually the one that returned the shortest distance. One exception is the case in which a particle is coming into the solid from a corner or edge of the solid.

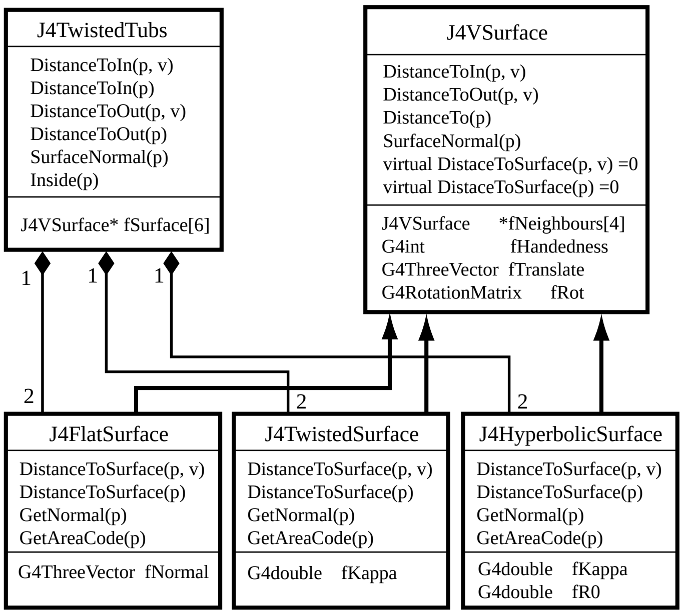


Figure 5: A class diagram in UML of `J4TwistedTubs`

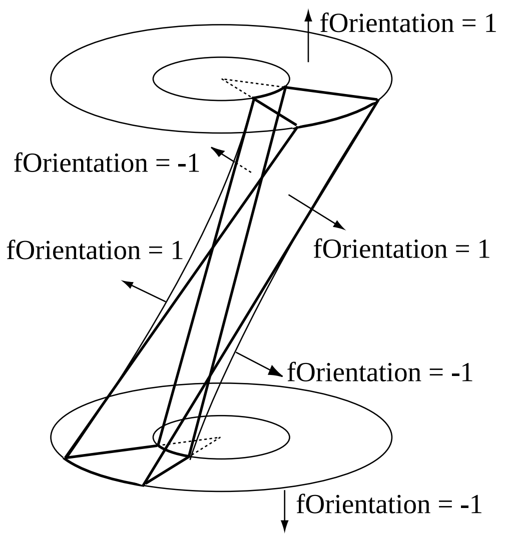


Figure 6: Values of `fOrientation` data member of `J4TwistedTubs`

<u>`J4VSurface`</u>

`G4int DistanceToSurface(const G4ThreeVector &p, G4ThreeVector *xx, G4double *distance, G4int *areacode) = 0`
which calculates the distance (`*distance`) from a point `p` to the surface as well as the point (`*xx`) of the closest approach to the surface and an `*areacode` given by `GetAreaCode` explained below, and then returns the number of the points of closest approach that is always 1.

`G4int DistanceToSurface(const G4ThreeVector &p, const G4ThreeVector &v, G4ThreeVector xx[], G4double distance[], G4int areacode[], G4bool isvalid[], EValidate validate) = 0`
which calculates the distance(s) (`distance[]`) along the velocity vector `v` to the surface from the point `p`, the intersection(s) (`xx[]`) of the particle track with the surface, and the areacode(s) (`areacode[]`) of intersection(s), and then returns the number of intersections. If so flagged by `validate`, it does boundary check for each of the candidate intersection(s) and passes the test result through `isvalid[]`. The return value then becomes the number of valid intersection(s).

`G4ThreeVector GetNormal(const G4ThreeVector &xx, G4bool isglobal) = 0`
which calculates and returns the normal at a surface point `xx`. The surface point `xx` and the normal to be returned are assumed to be given in the coordinate system of its mother solid or in the local frame of the surface in question, depending on whether `isglobal` is `TRUE` or not, respectively.

`G4Int GetAreaCode(const G4ThreeVector &xx, G4bool withTol) = 0`
: which decides whether the intersection `xx` given by `DistanceToSurface(p,v)` is on a boundary, or at a corner, or inside or outside of the surface with or without tolerance as flagged by `withTol`, and then returns a corresponding area code.

`G4double DistanceToIn(const G4ThreeVector &p, const G4ThreeVector &v, G4ThreeVector &xx)`
: which invokes `DistanceToSurface(p,v)` of its descendant concrete class and judges the validity of each of the resultant intersection(s) and the corresponding distance(s), examining the sign of the distance, the angle of the particle velocity `v` to the surface normal at the intersection point on the surface. If the intersection is on a boundary or at a corner of the surface, it also checks the angle to the surface normal(s) of the adjacent surface(s). It then returns the distance if valid, or infinity otherwise, together with the valid intersection `xx`, if any.

`G4double DistanceToOut(const G4ThreeVector &p, const G4ThreeVector &v, G4ThreeVector &xx)`
: which invokes `DistanceToSurface(p)` of its descendant concrete class and judges the validity of each of the resultant intersections(s) and the corresponding distance(s), taking into account the sign of the distance and the angle between the surface normal and the particle velocity `v`, and then returns the distance to the caller if valid, or infinity otherwise, with the valid intersection passed through `xx`, if any.

`G4double DistanceTo(const G4ThreeVector &p)`
: which just passes the return value of `DistanceToSurface(p)` without any additional judgment. This functions is used in `DistanceToIn(p)` or `DistanceToOut(p)` of `J4TwistedTubs`.

Notice that the first four of these member functions are pure virtual and should be implemented in its derived concrete classes. The implementations of these four functions in the three derived surface classes:
`J4HyperboloidalSurface`, `J4TwistedSurface`, and `J4FlatSurface`, are complicated but straightforward and can be carried out based on their mathematical representations in the previous section. Leaving out technical details for our CPC paper[8], we will move on to the test of the new solid (`J4TwistedTubs`) in the next section.

4 Test of J4TwistedTubs

In order to test the new solid class `J4TwistedTubs`, we prepared a test program (for stereo cells) as follows. We first constructed a world volume with a cubic shape of 6m $\times$ 6m $\times$ 6m and put, at its center, a cylindrical tube-type layer with inner and outer radii of 30cm and 130cm, respectively, and a length of 260cm. In this cylindrical layer, we placed a `J4TwistedTubs` object, which has a twist angle ($\Delta\phi$) and a ϕ-width of $\pi/3$, inner and

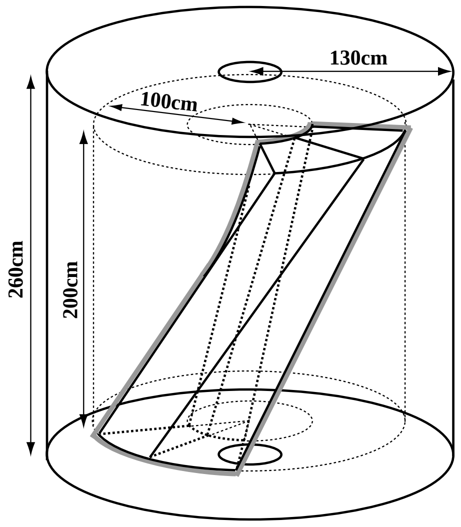


Figure 7: Schematic view of the test geometry

outer radii at the endcaps of 50cm and 100cm, respectively, and a z-length of 200cm. Into this mother `J4TwistedTubs` volume, installed were two daughter `J4TwistedTubs` objects which have half the ϕ-width but otherwise the same geometry. All of these volumes are made of air. A schematic view of the test geometry is shown in Fig. 7.

In order to evaluate the computing speed of `J4TwistedTubs`, we also prepared another test program (for axial cells) with the mother and the two daughter `J4TwistedTubs` objects replaced by ordinary cylindrical tubs (`G4Tubs`).

4.a Test with Geantinos

Geant4 provides a hypothetical particle called "*geantino*" designed primarily for geometry tests. A geantino does not interact with any materials and flies along a straight line trajectory. It only makes a hit when it crosses a boundary of volumes which are *sensitive*[4]. We first carried out a geantino test of `J4TwistedTubs`, using the aforementioned geometry test program with only the daughter `J4TwistedTubs` objects made sensitive. Fig. 8 is a 3-dimensional view of hit points made by 10000 geantinos injected from 4π steradian around the surrounding cylinder. As seen from the figure all the hit points are correctly on the surface of the daughter `J4TwistedTubs` volumes.

4.b Test with Higgs Events

Unlike geantinos, a real charged particle makes a curved trajectory in a central tracker due to the magnetic field applied to it, and interacts with various detector materials,

[4]Geant4 generates a hit when (1) a physics process took place on a flying particle or (2) its track crossed a boundary of a volume registered as a sensitive detector.

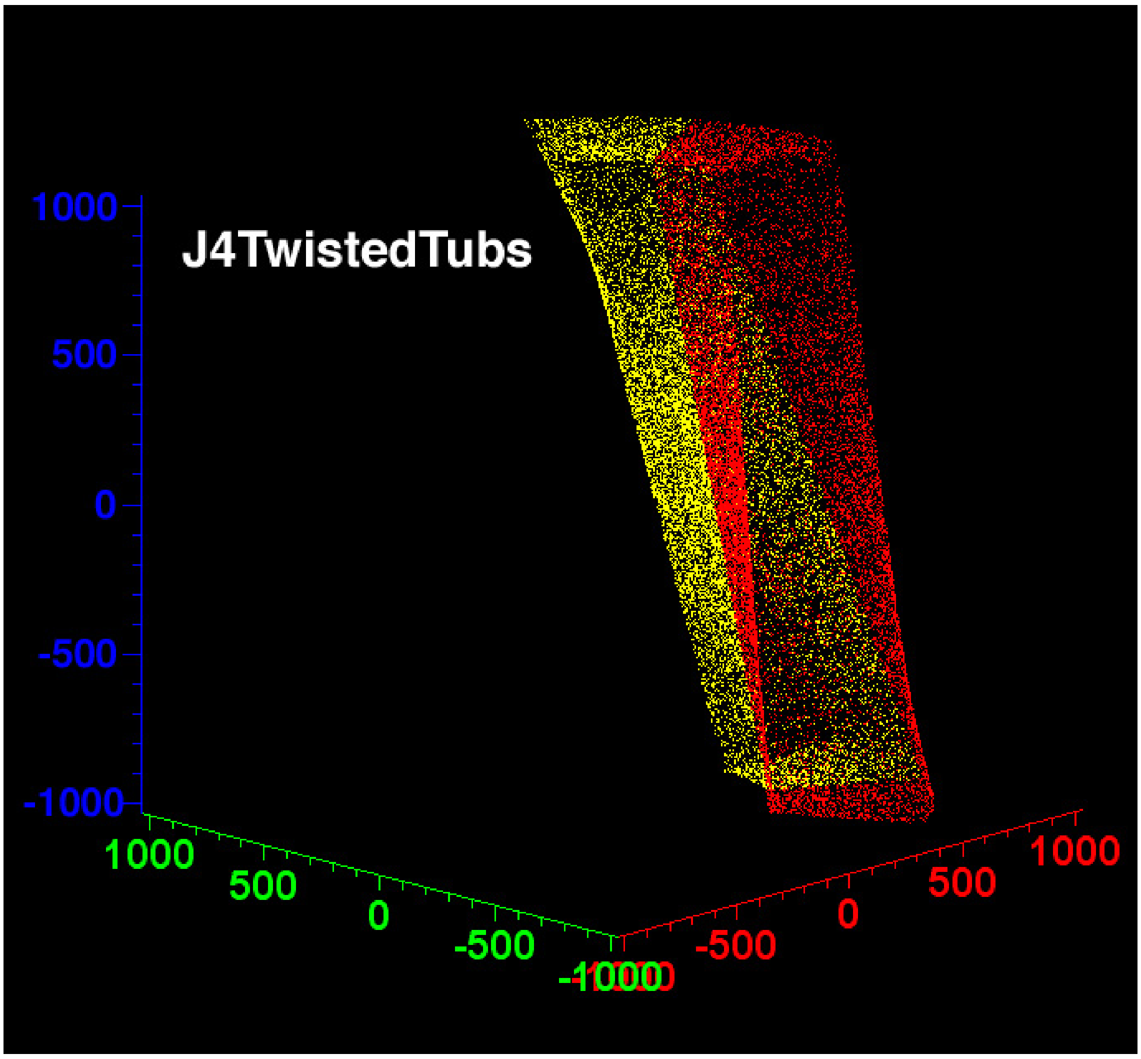


Figure 8: 3-dimensional view of hit points of geantinos on the daughter `J4TwistedTubs` volumes

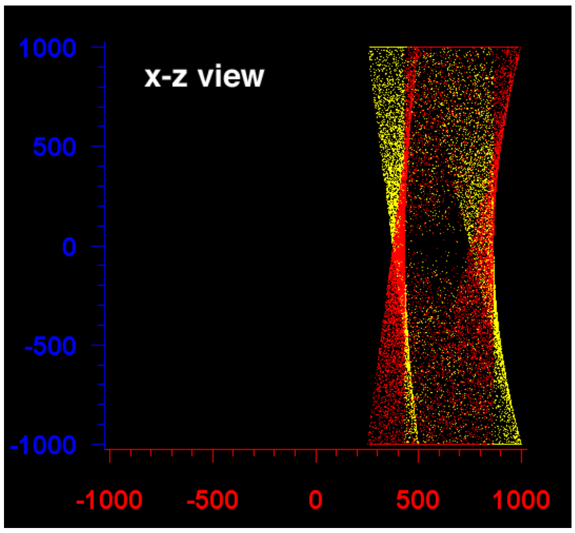


Figure 9: x-z view

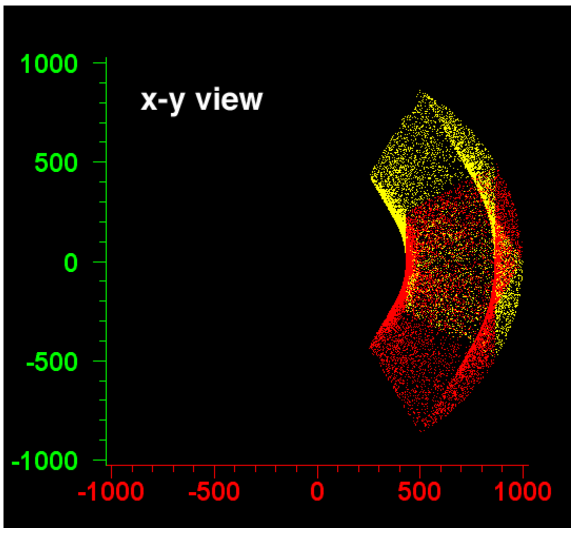


Figure 10: x-y view

producing, for instance, low energy tracks such as δ-rays which might complicate the particle tracking through the detector volumes. In order to stress-test our `J4TwistedTubs` under more realistic environment, we also tried Higgs events ($e^+e^- \to ZH$ at $\sqrt{s} = 350$ GeV) generated by Pythia[9]. In this case, the number of tracks with a transverse momentum of 1 GeV or greater is around 50 per event. Because of the lack of a calorimeter in this test program, however, relatively low energy tracks curl up and pass through the sensitive volumes multiple times. Taking into account this curl-up effect and the geometrical acceptance of the `J4TwistedTubs` volumes being about $(\pi/3)/(2\pi) = 1/6$, the average number of tracks passing through the twisted volumes was estimated to be around 10 per event. Processing of 1000 Higgs events, which thus correspond to roughly 10000 tracks hitting the sensitive volumes, took 11647 seconds on a Power Macintosh 800MHz (Dual CPU) with 1GB memory for the `J4TwistedTubs` test program. The same test took 9242 seconds on the same platform for the axial cell test program. The 25% extra CPU time consumed by `J4TwistedTubs` is acceptable for our purpose, considering the expected CPU time necessary for calorimeter simulation.

5 Bunch-separation capability of JLC-CDC

Owing to our new solid for stereo cells, now we can estimate the bunch-separation capability of the cylindrical stereo mini-jet cell drift chamber.

The bunch separation capability is essential for the JLC-CDC. As Fig.11 shows, the beam of JLC consists of bunch-trains, each of which contains 192 bunches with an interval of 1.4 nsec. The duration of one bunch train is thus 268.9 nsec. Since this train length is much shorter than the sweeping time of ionization electrons in a drift cell($< 6.7\mu$sec), background events from any bunches in a train can overlap with an important physics event.

In order to separate the signal event from these background events, we should be able to time-stamp tracks recorded in the CDC on a track-by-track basis. We can realize such time-stamping capability by staggering drift cells in adjacent layers. Fig.12 illustrates how. Suppose we have a mini-jet track from a beam bunch at $t = T'_0$ and a signal event at $t = T_0 > T'_0$. Since the drift cells are staggered from layer to layer, the ionization electrons generated by the mini-jet track will drift over some length L in the opposite directions in the adjacent cells. This will create discontinuity between track segments as shown by the dotted lines in Fig.11. The signal track, on the other hand, will look like the solid line in the same figure, where one will find no discontinuity.

It is thus expected that we will be able to eliminate background tracks by checking the goodness of the fit (χ^2) or something equivalent. There is, however, no article confirming the speculation especially in the case of stereo mini-jet cell chambers. In order to quantify the time-stamping capability of staggered mini-jet cells and to investigate the effect of the stereo layers, we studied the time-stamping performance of both a axial only geometry and a axial + stereo geometry.

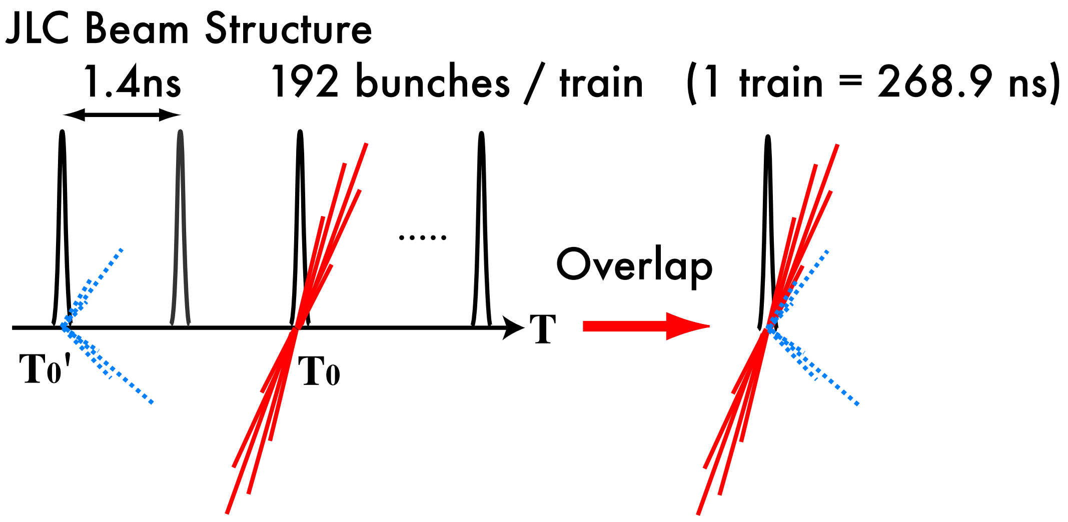


Figure 11: Schematic view of the overlapping events

5.a Analysis procedure

The procedure of analysis is the following:

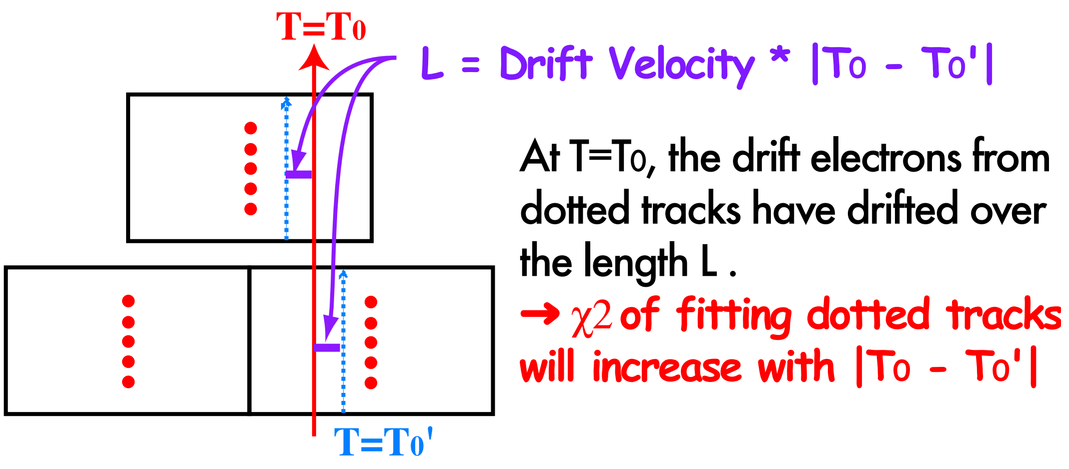
1. Generate 250 GeV single-muon tracks by JUPITER (2000 event).
2. Move T_0 and consequently the hit points artificially in Satellites.
3. Fit each single-muon track to a helix in Satellites.
4. Calculate χ^2.

The artificial T_0 shift was chosen to be 0 ns, 14 ns, 21 ns, and 28 ns, which correspond to 0, 10, 15, and 20 bunches, respectively. Figs.13 and 14 are typical χ^2 distributions: the former represents a hypothetical configuration which includes axial cells only, while the latter includes both axial and stereo cells. The dotted histograms are from the helix fit to the simulated data, while the solid lines are expected χ^2 distributions in the case of $ndf = 95$[5]. Their good agreement demonstrates that our helix fitting is working correctly.

5.b Results

The χ^2 distributions for different T_0's are compared in Figs.15 and 16. As expected, the peak values of the χ^2 distributions increase with the input T_0 values. The peak values are, however, significantly different between the axial only and the axial + stereo cases.

[5]In the case of the 3T design, the CDC has 50 sampling points, each provides a drift time and a z coordinate from charge division. Subtracting the number of helix parameters of 5, we have $2 \times 50 - 5 = 95$ as the number of degrees of freedom.



T=T0
L = Drift Velocity * |T0 - T0'|
At T=T0, the drift electrons from dotted tracks have drifted over the length L .
→ χ2 of fitting dotted tracks will increase with |T0 - T0'|
T=T0'

Figure 12: A staggered cell structure

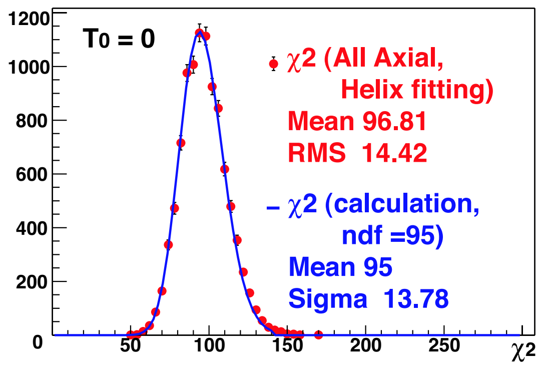


Figure 13: Axial cells installed in all layers (0-9th layers)

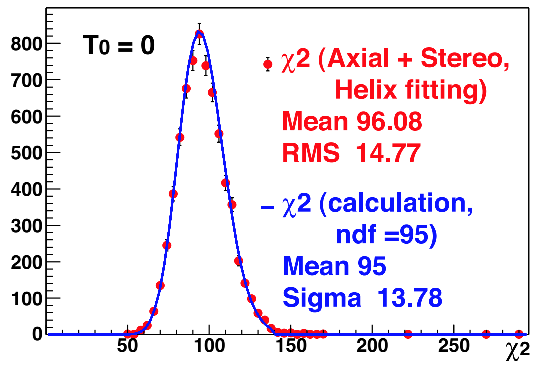


Figure 14: Axial cells installed in 0, 3, 6, and 9th layers. Stereo cells installed in the rest.

This is because the discontinuity between staggered cells of adjacent stereo layers can be compensated by adjusting $\tan\lambda$ within the range that σ_z allows. This is reflected not only in the difference of the peak values but also in the difference of the widths of the distributions. The RMS increases with T_0 up to 43.5 for the axial only and 56.5 for the axial + stereo geometries at $T_0 = 28$ nsec (20-bunches away).

The results imply that with this simple χ^2 method, a stereo mini-jet cell drift chamber can separate background tracks occurring > 20-bunches away. The number of mini-jet background events is, however, estimated to be ~ 1 per 10-bunches. It is hence desirable to have a better time-stamping resolution. To improve the time-stamping capability, we added one more free parameter T_0 into our helix fitting program. Fig.17 shows the χ^2 distribution for the axial + stereo geometry at $T_0 = 21$ns, which is remarkably improved from the corresponding plot in Fig.16. The distribution is consistent with the expected

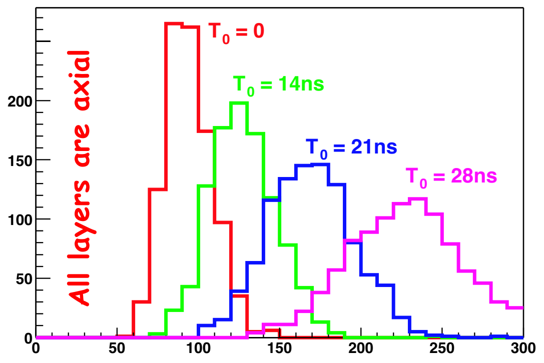


Figure 15: Axial cells installed in all layers (0-9th layers)

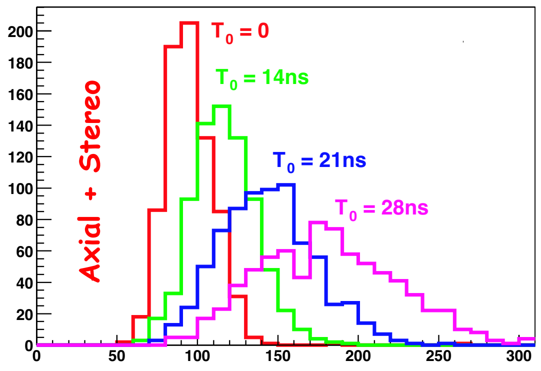


Figure 16: Axial cells installed in 0, 3, 6, and 9th layers. Stereo cells installed in the rest.

χ^2 distribution for $ndf = 94$.

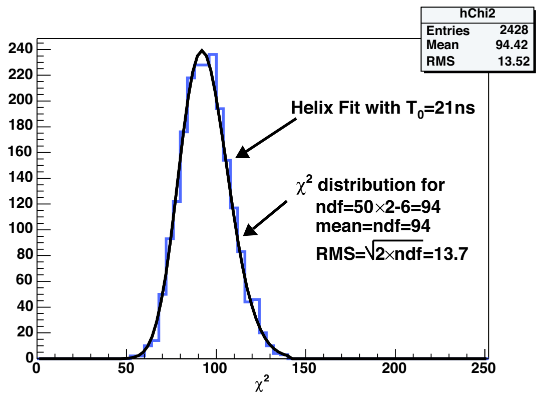


Figure 17: χ^2 distribution for the axial + stereo geometry at $T_0 = 21$ns.

The T_0 distributions from the fit are shown in Figs.18 and 19 for the axial only and the axial + stereo geometries, respectively. We can see that the resultant T_0 gaussian-distributes itself with mean values right at the input T_0 values and with an σ_{T_0} of 2.15 nsec for the axial only and 2.84 nsec for the axial + stereo geometries, which is independent of the input T_0 values[6]. This resolution indicates that the cylindrical stereo mini-jet cell drift chamber can separate background tracks that are > 7 bunches away at a 3.5 σ level, even though the time-stamping resolution is 30% worse than the axial only case. These preliminary results indicate that the time-stamping resolution satisfies the requirements.

[6]Naive estimate based on the spatial resolution, the drift velocity, and the number of sampling points gives 1.8 nsec for the axial only geometry: $\sigma_{T_0} \simeq \sigma_{xy}/v_{drift}\sqrt{N_{sample}}$. This is comparable with our simulated result of 2.15 nsec.

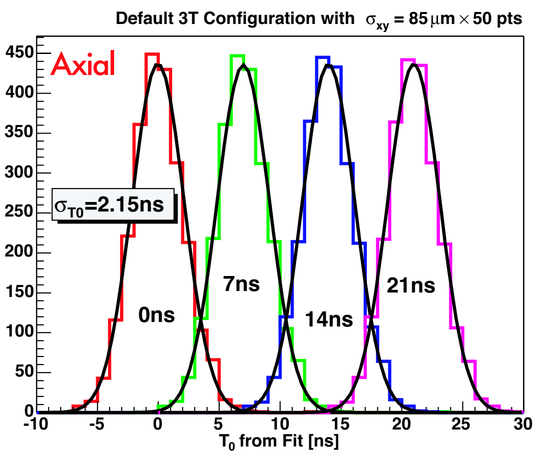


Figure 18: Axial cells installed in all layers (0-9th layers)

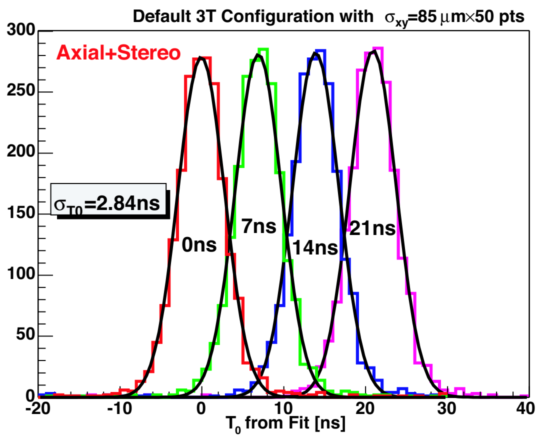


Figure 19: Axial cells installed in 0, 3, 6, and 9th layers. Stereo cells installed in the rest.

6 Summary

We have developed a new Geant4 solid called `J4TwistedTubs` in order to handle stereo mini-jet cells of a cylindrical drift chamber like JLC-CDC. This new solid consists of three kinds of surfaces, each of which is represented respectively by `J4HyperboloidalSurface`, `J4FlatSurface`, and `J4TwistedSurface`. These three surface classes correspond to inner and outer hyperboloidal surfaces, two end planes, and two so-called twisted surfaces that make slant and twisted ϕ-boundaries, respectively.

We have implemented stereo cells with the new solid, and tested them using geantinos and Pythia events ($e^+e^- \to ZH$ at $\sqrt{s} = 350$ GeV). The stereo cells consumed 25% more CPU time than ordinary axial cells did. We found this acceptable, considering the expected CPU time necessary for calorimeter simulation.

Using the new solid, we installed stereo cells into JUPITER. We then estimated the time-stamping capability of the staggered cell structure as of our JLC-CDC using the JUPITER generated Monte Carlo events. For the axial only geometry we obtained a time-stamping resolution of 2.15 nsec which is comparable with the naive analytic estimate of 1.8 nsec. We found that the inclusion of stereo layers deteriorated the time-stamping resolution by 30% and yielded a T_0 resolution of 2.84 nsec, for the 3T design of the JLC-CDC. This time-stamping resolution implies that the JLC-CDC can clearly separate the background tracks occurring > 7 bunches away, and is compliant to the requirement from the estimated background rate of 1 event per 10 bunches.

Acknowledgments

The author would like to thank K. Fujii and O. Nitoh for the collaboration from which this work emerged. The author would also like to thank A. Miyamoto, T. Aso, N. Khalatyan,

R. Kuboshima, and all the other members of the ACFA physics working group and JLC software group for useful discussions and helps. She is also grateful to Geant4 users group and developers. In particular, a discussion with G. Cosmo has been very useful.

References

[1] JLC group, KEK Report 92-16, December, 1992.

[2] ACFA Linear Collider Working Group, KEK Report 2001-11, August (2001), http://www-jlc.kek.jp/subg/offl/jim/index-e.html

[3] Proceedings of the APPI Winter Institute, KEK Proceedings 2002-08, July (2002)

[4] http://root.cern.ch/

[5] http://www-jlc.kek.jp/subg/offl/jsf/index.html

[6] N.Khalatyan, K.Fujii, H.Okuno,T.Abe, K.Hoshina, Y.Kato, Y.Kurihara, H.Kuroiwa, T.Matsui, O.Nitoh, A.Sugiyama, K.Takahashi, T.Watanabe, T.Yoshida, Nucl. Inst. and Meth. **A428** (1999) 403.

[7] http://wwwinfo.cern.ch/asd/geant4/G4UsersDocuments/UsersGuides/ForToolkitDeveloper/html/index.html

[8] K. Hoshina, K. Fujii, O. Nitoh (Tokyo U. of Agric. Tech.) e-Print Archive: hep-ex/0303014
KEK-PREPRINT-2002-140, Mar 2003. 24pp.
to appear in CPC soon.

[9] http://www.thep.lu.se/t̃orbjorn/Pythia.html