STAR NOTE 145
STAR SVT TRACKING PACKAGE

Claude A. Pruneau
Physics Department
Wayne State University, Detroit, MI 48201, USA

ABSTRACT

This document reviews the SVT tracking software developed up to date for the purpose of Monte Carlo Simulations. The methods used are described in some details. A detailed USER manual is also included.

1. Introduction

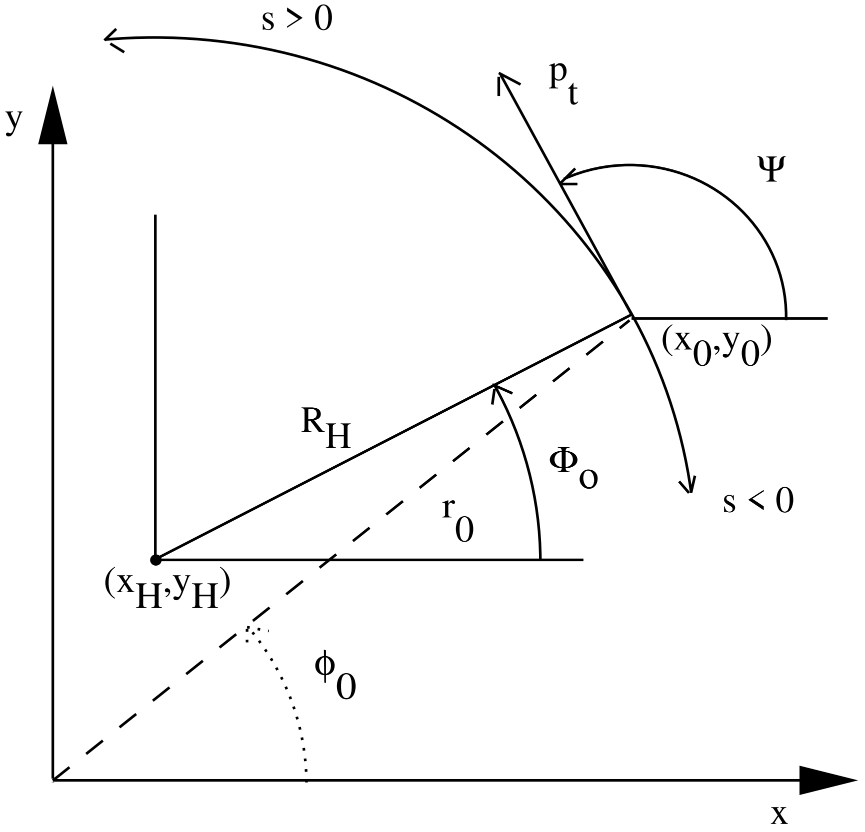
This document describes the tracking software developed to date (dec 1, 1993) to study the performance of the STAR Silicon Vertex Detector (SVT) as a tracking detector. It focuses on the technical aspects of the software rather than the results of the simulations and is meant as a specification document and usage guide for prospective users. We assume the reader is already familiar with the STAR software environment and concepts (TAS, Entity Relationship Diagram, TAS Tables etc). The user should also bear in mind the this code is still under development and tuning and improvements are forseen for the near futur.

The document is divided in three main sections: The first Section (Specifications) describes the general philosophy and the specific techniques used for track finding and fitting, etc. Section 2 lists the TAS Data Structures defined to represent the data. The track finding, track fitting and the calculation of "physics" observables is accomplished by means of many different modules. Section 3 describes in detail the functionality of these modules, their input/ouput TAS tables and provides typical MID files and KUMAC files (if relevant) to setup the module's control parameters.

2. Specifications

In this section, we first introduce the track parametrization used in the SVT tracking. Wherever possible, conventions and parameters identical to those used for the TPC are used. However, there are differences in the track finding and fitting procedures. Two track finding techniques are currently used for SVT track reconstruction. The first one, described here, uses a conventional "follow your nose" approach. The other is based on a radial mapping and has been described elsewhere [1].

2.0 Track Parametrization



$$x(t) = x_o + R_H[\cos(\Phi_o + h\,t\cos\lambda/R_H) - \cos\Phi_o]$$

$$y(t) = y_o + R_H[\sin(\Phi_o + h\,t\cos\lambda/R_H) - \sin\Phi_o]$$

$$z(t) = z_o + t\sin\lambda$$

Fig. 1: Helix Parametrization

The tracjectory of a particle in a static uniform magnetic field with a field vector parrallel to the z axis (also the beam axis) is a helix with an axis along the field lines. We use a parametrization identical to that used for TPC tracking[2] as illustrated in Fig. 1

$$
\begin{aligned}
x(s) &= x_0 + R_H[\cos(\Phi_0 + \frac{hs\cos\lambda}{R_H}) - \cos\Phi_0] &(1)\\
y(s) &= y_0 + R_H[\sin(\Phi_0 + \frac{hs\cos\lambda}{R_H)} - \sin\Phi_0] &(2)\\
z(s) &= z_0 + s\sin\lambda &(3)
\end{aligned}
$$

where
t : path length along the helix. Increases when moving in the particle's direction.
(x_0, y_0, z_0) : reference point, $s = s_0 = 0$.
λ : slope or dip angle measured from the transverse plane, range: $-\pi/2 < \lambda < \pi/2$
B : value of the magnetic field parallel to the z axis. (positive B along +z)
R_H : Helix's radius
κ : conversion factor (for radius in [cm], magnetic field in [kGs] and momentum in [GeV], $\kappa = 2.9E-4$)
q : charge of the particle in units of proton charge
h : direction of rotation of the projected helix in the xy plane
Ψ : azimuthal angle of the momentum vector
Φ_0 : helix phase, azimuthal angle of reference point relative to the center of the circle.
ϕ_0 : azimuthal angle of the reference point.

$$
\begin{aligned}
\lambda &= arcsin(dz/ds) &(4)\\
R_H &= P\frac{\cos\lambda}{(\kappa qB)} &(5)\\
h &= -sign(qB) = \pm 1 &(6)\\
\Phi_0 &= \Psi - h\pi/2 &(7)\\
& &(8)
\end{aligned}
$$

The phase of the helix is defined using the first space point on the track (x_1, y_1):

$$\Phi_0 = \arctan(\frac{y_1 - y_H}{x_1 - x_H}) \tag{9}$$

The phase Φ_0 of the helix and the coordinates of the center of the circle are then used to calculate the reference point (x_0, y_0):

$$
\begin{aligned}
x_0 &= x_H + R_H\cos\Phi_0 &(10)\\
y_0 &= y_H + R_H\sin\Phi_0 &(11)
\end{aligned}
$$

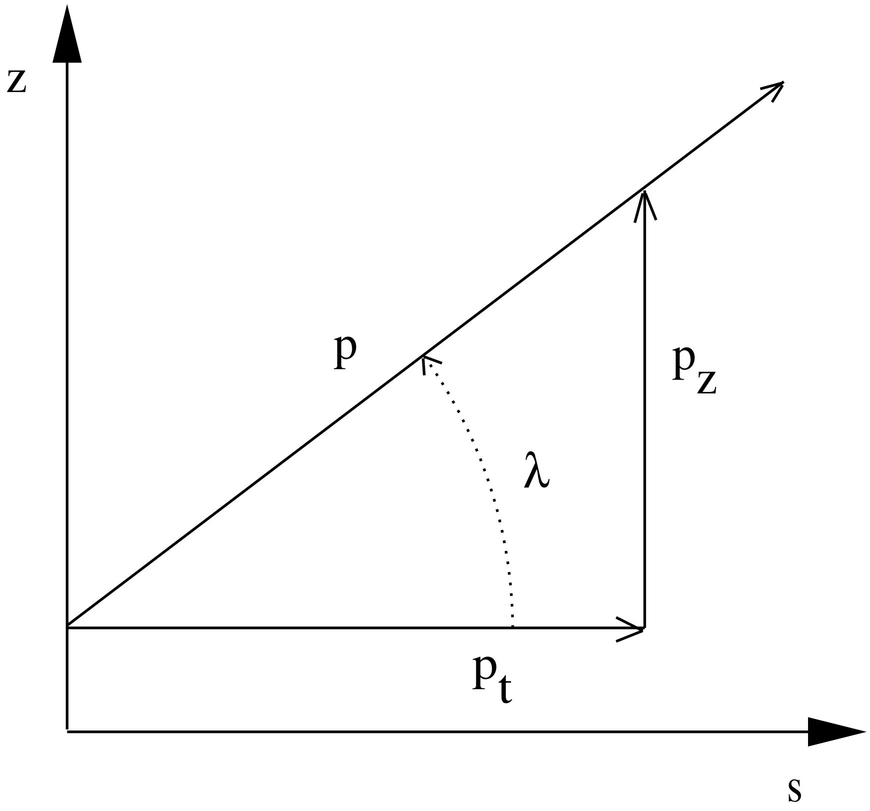


Fig. 2: Helix Parametrization: Longitudinal Motion

In addition, one should note that by construction, one has as illustrated in Fig. 2.

$$dz/ds = \tan\lambda \tag{12}$$

where

$$s = \sqrt{x^2 + y^2} \tag{13}$$

The following parameters are used to specify helix trajectories uniquely:

The reference point: (r_0, ϕ_0, z_0)

$$r_0 = \sqrt{x_0^2 + y_0^2} \tag{14}$$
$$\phi_0 = \arctan(y_0, x_0) \tag{15}$$

The azimuthal angle of the track: Ψ

$$\Psi = \Phi_0 + h\pi/2 \tag{16}$$

The dip angle, $tan\lambda$ and the inverse momentum, $1/p$, are obtained from the expression for p_z and p_t:

$$p_t = R_H|\kappa q B| \tag{17}$$
$$p_z = p_t \tan\lambda \tag{18}$$
$$p = \sqrt{p_t^2 + p_z^2} \tag{19}$$

2.1 Track Finding

Track finding is performed using a "follow your nose" approach starting with space points on the first SVT layer as illustrated in Fig. 3. We follow essentially the method used by Bielecki [3].

The reconstructed space points stored in the SVT_SPT table are first sorted by layer using a separete key index i.e. no table rows are swapt, instead an independant array is used to store the space point row indices sorted by layer number.

The track reconstruction is thus performed with three nested loops on the three layers of space points. One might eventually relax this constraint and allow 2 or more space points instead. The search for track candidates proceed as follow: Given an approximate vexter position, one consider in turn all points of the first layer. For each of those, a vector pointing from the vertex to the point is built and extrapolated to point at the second layer. The direction/position thus defined is used as the center of a rectangular cone of dimensions θ_x^{12},θ_y^{12},θ_z^{12} as illustrated on figure Fig. 3. The cone size is used to calculate the range $[x_{low}, x_{high}]$, $[y_{low}, y_{high}]$,$[z_{low}, z_{high}]$ within which points of the second layer are searched for to begin track segments. All second layer space points are examined for each first layer space point cone. If no points are found the search is repeated with a cone size increased by a user defined increment. If unsuccessful the size is again increased and the search is once

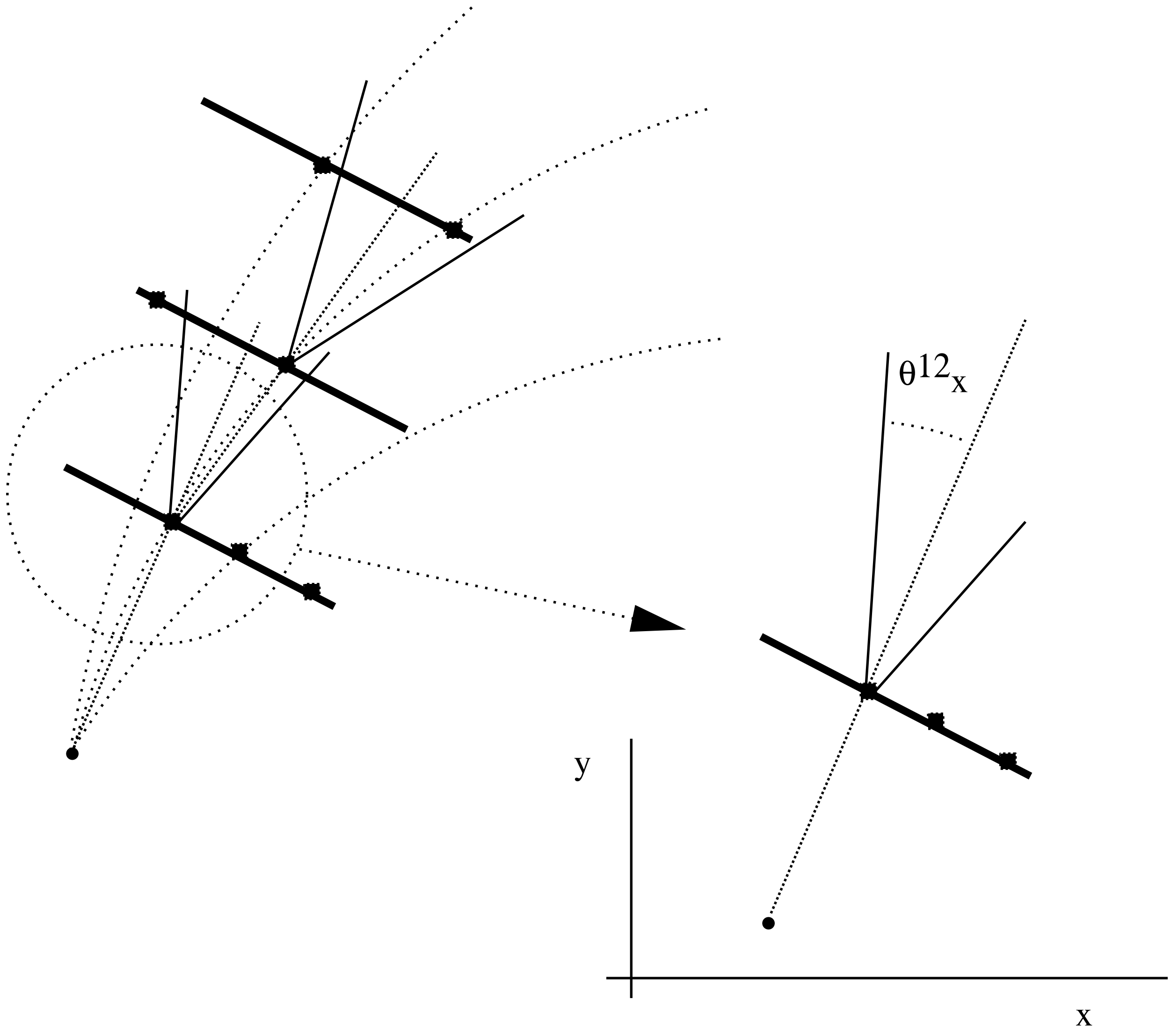


Fig. 3: Space Point Search in a Cone

again repeated. The maximum number of retries is controlled by a user defined parameter. For each point found on the second layer, a new vector is built from the point on the first layer to the point on the second layer and is extrapolated to point at the third layer. A similar cone as for layer two is applied. Its minimal size is determined by user defined parameters θ_x^{23}, θ_y^{23},θ_z^{23}. Here also, the search extends on all space points of the third layer and is repeated with an increased cone size if the search is unsuccesful. Once three space points have been found to be within a cone and to form a potential track candidate, their row index (in table SVT_SPT) are stored in the track candidate table SVT_CTRACK. The search continues until all space points of the first layer have been exhausted.

2.2 Track Fitting

2.2.1 Introduction

The SVT track fitting algorithm uses the helix parametrization described in the previous section. We assume the magnetic field is sufficiently uniform and the energy loss small enough to neglect variations of the radius of curvature along the track. In view of the relative compactness of the SVT detector and its small equivalent thickness, this seems a natural simplication. With a uniform, constant axial field, the helicoidal movement is decomposed into a circle as projected in the xy plane and a straight line as projected in the sz plane. We discuss the circle and linear fitting procedure separately. As the SVT detector provides in general only 3 space points, the first track finding/fitting pass on the data includes the estimated primary vertex as part of the helix.

2.2.1 Circle Fitting

Circle fitting is reduced to a linear fitting problem by means of a conformal mapping of the xy plane onto a uv plane as defined below:

$$u = \frac{x}{(x^2+y^2)} \tag{20}$$

$$v = \frac{y}{(x^2+y^2)} \tag{21}$$

Such a comformal mapping is applicable only if one the fitted points lies close or at the origin. Currently, this is realized by forcing the primary vertex to be exactly on the circle. To do this, all points are translated in x and y by a distance equal to the distance between the estimated vertex (x_v, y_v) and the origin.

$$x' = x - x_v \tag{22}$$

$$y' = y - y_v \tag{23}$$

The origin $(0, 0)$ being on the circle, the equation of the circle

$$R_H^2 = (x - x_H)^2 + (y - y_H)^2 \tag{24}$$

reduces with the conformal mapping defined as above to

$$v \;=\; a(1) + a(2)u \tag{25}$$

2.2.2 Linear Fitting

2.3 Physics Observables

Once the helix parameters have been obtained by the fitting routine, the routine SVT_TRACKING_KINE calculates the total momentum, the momentum vector, the emission angles at the production vertex currently assumed to be the primary vertex.

In addition, one also calculates the transverse momentum,$p_\perp$, and the rapidity of the particles.

2.4 Geometry Definition

The SVT geometry definition is accomplished with the SVG package. For the purpose of the simulation, we define the position and orientation of each detector wafer as illustrated on Fig. 4. with the following quantities:
$\vec{x}$: position of the center of the wafer in the STAR reference frame.
$\vec{n}$: vector normal to the drift plane of the wafer.

$\vec{d}$: vector indicating the drift direction.

$\vec{t}$: vector indicating the transverse direction.

$\omega(3)$: size of the wafer in the $\vec{d}$, $\vec{t}$ and $\vec{n}$ direction.

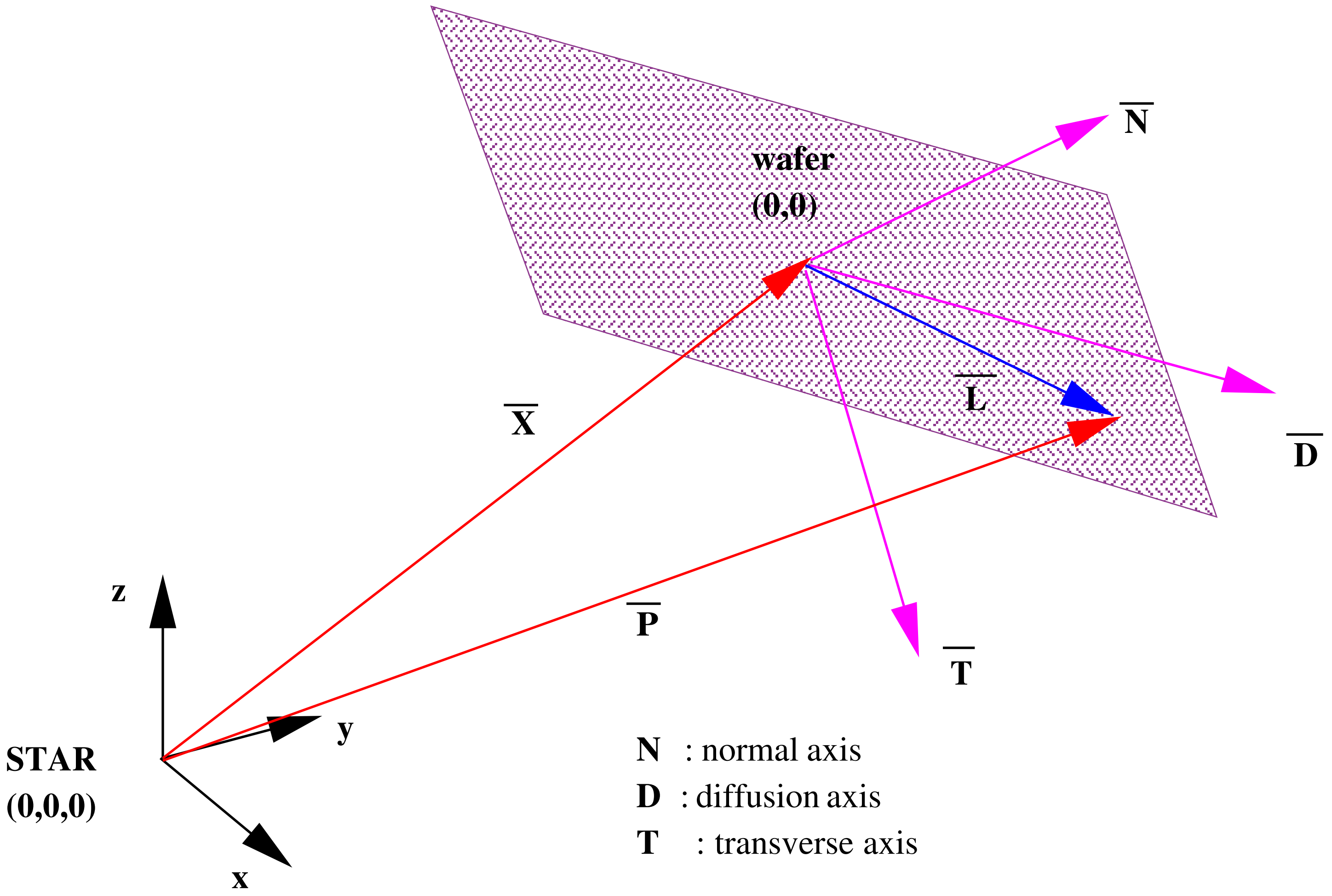
For the purpose of the simulation, it was convenient to parametrize and automatize the generation of the position and orientation of the wafers as well as the many other volumes and materials required for proper GEANT simulations of the impact of the geometry and the presence of all extra materials required for mechanical support and cooling of the detectors.

The generation of the SVT geometry is governed by a configuration table SVT_CONFIG and subsidary wafer shape table SVT_SHAPE. The configuration table defines how many SVT layers are to be used, how many ladders per layer, how many wafers on each ladder, the size, position of the layers, ladders etc etc. We use the following parameters:

```
\noindent cap_nshape = number of shape parameters
\noindent cap_shape_id = GEANT shape id for end caps
\noindent cap_trackmed = tracking medium for end caps
\noindent frame_trackmed = tracking medium for frame
\noindent layer_drift(10) = drift direction indicator 3=Z=beam axis
\noindent n_ladder(10) = Number of ladders in layer(x)  x-->1-6
\noindent n_layer = number of layers
```

SVT Geometry

Local to Global/Global to Local Space Point Mapping



N : normal axis

D : diffusion axis

T : transverse axis

T = N x D

X : wafer center

P : space point in global coordinate

L : space point in local coordinate

L = P − X ; **P = X + L**

$L_x = L \,.\, T$

$L_y = L \,.\, D$

$L_z = L \,.\, N$

Fig. 4: Parametrization of the wafer position and orientation

```
\noindent n_wafer(10) = Number of wafers on ladder of layer(x)
\noindent be_support_thick(10) = berylium support
\noindent be_support_width(10) = berylium support
\noindent cap_position = distance of svt end cap from origin
\noindent cap_shape(20) = shape parameter values
\noindent frame_al_length(10) = frame
\noindent frame_al_thick(10) = frame
\noindent frame_al_width(10) = frame
\noindent frame_cs_length(10) = frame
\noindent frame_cs_thick(10) = frame
\noindent frame_cs_width(10) = frame
\noindent frame_length(10) = length of frame
\noindent frame_si_length(10) = frame
\noindent frame_si_thick(10) = frame
\noindent frame_si_width(10) = frame
\noindent frame_thick(10) = frame thickness
\noindent frame_width(10) = frame width
\noindent layer_back(10) = layer back
\noindent layer_lateral(10) = lateral offset for ladder
\noindent layer_longi(10) = longitudinal offset for ladder
\noindent layer_phioff(10) = phi offset for 1st ladder of layer
\noindent layer_radial(10) = radial offset (wafer staggering)
\noindent layer_radius(10) = radius of layers
\noindent layer_shape(10) = shape id for layer
\noindent size(3) = TUBE inner,outer radius and half lenght
\noindent water_channel_thick(10) = water channel
\noindent water_channel_width(10) = water channel
\noindent x(3) = position of the SVT in STAR
```

The meaning of some of these parameters is illustrated in the following figures.

2.5 Vertex Finding

Currently, no vertex finding routines based on SVT or TPC tracks are available. So, we fake the determination of the vertex with a Gaussian random number generator. We use a sigma of 50 microns.

2.6 SVT Gaussian Resolution Simulator

The SRS package now includes three routines for the study of detector resolution effects. Two simple routines have been written, SVT_SPT_DIRECT and SVT_SPT_DIRECT_GS, to provide the ability to study track reconstruction and track fitting and resolution with no space point smearing or with controlled gaussian smearing (_GS). However, one should use the full resolution simulator [5] for a detailed study of effects of the response of the drift detectors.

2.7 dE/dx and Particle Identification

A package called SPR is under development for the reconstruction of the particle's identity with the energy loss measured with SVT [4].

2.8 Evaluation Package (STE)

In order to study the performance of the STK package, we wrote an evaluator module (STE). The evaluation of the STK package involves two aspects. In one, one wishes to determine the capacity of the code to recognize tracks. The evaluator determines the number of monte carlo tracks that traverse the SVT, the total number of space points and the number on each layer. It also determines, for each track, the number of monte carlo space points, the number of reconstructed space points for each candidate and determines what fraction of the space points are rightly or wrongly assigned.

The second aspect concerns the momentum resolution of the detector. Both the Monte Carlo tracks reconstructed with the DIRECT branch and those reconstructed with the FILLER or GROUPER branch are compared with the actual momentum of the particles as output by GEANT. The evaluator calculates the differences in p_x, p_y, p_z, p_{tot}, and a few more ... (see a detailed list in section 3)

3. Data Structures

In this section, we list all data structures involved in the definition of the geometry, the simulation of the vertex finding, the space point reconstruction, the track fingind and track fitting as well as the evaluation of performance of the track finding/fitting algorithm.

The data structures are listed in a somewhat arbitrary order. For each structure, we include the package name, a short description of the content and role of the structure, and a complete list and definition of the variables as currently defined.

3.1 SVT_VTX

3.1.1 Package: SVR

3.1.2 Description:

This table contains the position of the Primary vertex and its time of occurence.

```
Type      Name     Comment
=======================================================
        INTEGER*4 flag     Status flag.
        INTEGER*4 id       Identifier for Vertex row
        REAL*4    cov(6)   covariance matrix
        REAL*4    t        Time of Vertex (ns)
        REAL*4    x(3)     Space point coordinates in STAR frame

Note:  Status Flag: The status flag is not currently defined.
Note:  Covariance Matrix: cov(6)
```

We apply the convention:

$$
\begin{aligned}
cov(1) &= \sigma_x^2 && (26)\\
cov(2) &= \sigma_x \sigma_y && (27)\\
cov(3) &= \sigma_x \sigma_z && (28)\\
cov(4) &= \sigma_y^2 && (29)\\
cov(5) &= \sigma_y \sigma_z && (30)\\
cov(6) &= \sigma_z^2 && (31)
\end{aligned}
$$

3.2 SVT_SPT

3.2.1 Package: SCS

3.2.2 Description: SVT Space Points Table

This tables includes a parametric representation of reconstructed SVT space points.

```
Type      Name        Comment
=======================================================
        INTEGER*4 flag        Status of point.
        INTEGER*4 id          Identifier of reconstructed space point
        INTEGER*4 id_cluster  Link to the cluster this point came from
        INTEGER*4 id_mctrack  Link to Monte Carlo track
        INTEGER*4 id_track    Link to reconstructed track
        INTEGER*4 id_wafer    Identifier in svt_geom table of wafer
        REAL*4    cov(3)      Errors on X (diagonal for now)
        REAL*4    de(2)       Fitted or Estimated dE in Si (+error)
        REAL*4    x(3)        Coordinates in STAR reference system
```

3.3 SRS_DIRECT

3.3.1 Package: SRS

3.3.2 Description: Space Points Smearing Parameters

This table includes the standard deviations for the Gaussian smearing of SVT space points with the SVT_SPT_DIRECT_GS routine.

```
Type      Name     Comment
=======================================================
        REAL*4    sd       sigma in drift direction
        REAL*4    st       sigma transverse direction
```

3.4 SVT_FILLER

3.4.1 Package: STK

3.4.2 Description: Parameters for svt_tracking_fill routine

```
Type      Name     Comment
=======================================================
        REAL*4  d12       average distance between layers 1 & 2
        REAL*4  d23       average distance between layer 2 & 3
```

```
REAL*4  thx12     cone angle -x layer 1/2 to 3/4
REAL*4  thx23     cone angle -x layer 3/4 to 5/6
REAL*4  thy12     cone angle -y layer 1/2 to 3/4
REAL*4  thy23     cone angle -y layer 3/4 to 5/6
REAL*4  thz12     cone angle -z layer 1/2 to 3/4
REAL*4  thz23     cone angle -z layer 3/4 to 5/6
```

3.5 SVT_CTRACK/SVT_DTRACK

3.5.1 Package: STK

3.5.2 Description:

This table includes for every track candidates (or Monte Carlo tracks), a list of associated space points.

```
Type      Name     Comment
=======================================================
        INTEGER*4 id       Identifier number for track candidate
        INTEGER*4 id_mtrk  Link to MKINE table (MC tracks, GEANT
                           KINE bank output)
        INTEGER*4 id_rtrk  Link to SVT_TRACK table (reconstructed
                           track helix parameters)
        INTEGER*4 nspt     Number of active spt in candidate
        INTEGER*4 spt(6)   Links to SVT_SPT table (Space points)
```

3.6 SVT_TRACK/SVT_MCTRACK

3.6.1 Package: STK

3.6.2 Description: Helix Parameters of Reconstructed/MC tracks

This table includes helix parameters for all reconstructed tracks and some dedx information and particle id information.

```
Type      Name       Comment
=======================================================
        INTEGER*4 flag       Status flag
        INTEGER*4 id         Reconstructed/MC track Identifier
        REAL*4    chisq(2)   Chi squares of circle and line fits
        REAL*4    cov(15)    Covariant matrix (see belo for convention)
        REAL*4    dedx(2)    dE/dx information with error
        REAL*4    invp       (charge)/momentum at reference point
        REAL*4    phi0       Azimuthal angle at reference point
        REAL*4    pid        Particle ID reconstructed with SVT alone
        REAL*4    Psi        Azimuthal angle of momentum vector at
                             reference point.
        REAL*4    r0         Radial distance from beam axis for
                           reference point.
        REAL*4    tanl       Tangent of dip angle
        REAL*4    z0         z coordinate of reference point
```

3.7 SVT_KINE/SVT_MCKINE

3.7.1 Package: STK

3.7.2 Description: Kinematic Observables Reconstructed with SVT alone

This table includes reconstructed kinematic observables such as the the 3-moment, the polar and azimuthal angle of the particle's trajectory at its production vertex.

```
Type      Name     Comment
=======================================================
        INTEGER*4 id         Identifier
        INTEGER*4 id_vtx    Link to SVT_VTX table
        INTEGER*4 pid        GEANT Particle Identication Number
        REAL*4    p(3)       Reconstructed 3-momentum
        REAL*4    ph         Azimuthal angle (degrees)
        REAL*4    pt         Transverse momentum
        REAL*4    ptot       Total momentum
        REAL*4    sec        Flag 1=primary, 2=secondary
        REAL*4    th         Polar angle (degrees)
```

3.8 SVT_GEOM

3.8.1 Package: SVG

3.8.2 Description: SVT Wafer Geometry

This table includes the position and orientation of the SVT wafer in the STAR reference frame. The orientation of the wafers is defined by means of three vectors: one pointing in the drift direction, one point in the direction normal to the plane of the wafer, and one perpendicular to both of these.

```
Type      Name     Comment
=======================================================
        INTEGER*4 id       Wafer id number
        INTEGER*4 id_shape Wafer shape identifier
        INTEGER*4 ladder   Ladder number wafer is on
        INTEGER*4 layer    Layer number wafer is on.
        INTEGER*4 num_chip Chip number within ladder
        REAL*4  d(3)       Unit vector in Drift direction
        REAL*4  n(3)       Unit vector normal to wafer
        REAL*4  t(3)       Unit vector transverse to drift and normal
        REAL*4  x(3)       Coordinates of wafer center in STAR frame
```

3.9 SVT_SHAPE

3.9.1 Package: SVG

3.9.2 Description: SVT Wafer Shape Definition

This table includes the definition of the shape and dimensions of the SVT wafers.

```
Type      Name       Comment
=======================================================
        INTEGER*4 id         Type of SDD shape
```

```
INTEGER*4 n_shape    Number of shape parameters
REAL*4    shape(10)  Custom shape parameters of SDD wafer
```

3.10 SVT_CONFIG

3.10.1 Package: SVG

3.10.2 Description: SVT Wafer Configuration Definition

This table includes all parameters that define the configuration of the SVT detector.

```
Type      Name                     Comment
================================================================
INTEGER*4 cap_nshape                  Number of shape parameters
INTEGER*4 cap_shape_id                GEANT shape id for end caps
INTEGER*4 cap_trackmed                Tracking medium for end caps
INTEGER*4 frame_trackmed              Tracking medium for frame
INTEGER*4 layer_drift(10)             Drift direction indicator 3=Z=beam axis
INTEGER*4 n_ladder(10)                Number of ladders in layer(x)  x-->1-5
INTEGER*4 n_layer                     Number of SVT layers
INTEGER*4 n_wafer(10)                 Number of wafers on ladder of layer(x)
REAL*4    be_support_thick(10)
REAL*4    be_support_width(10)
REAL*4    cap_position
REAL*4    cap_shape(20)
REAL*4    frame_al_length(10)
REAL*4    frame_al_thick(10)
REAL*4    frame_al_width(10)
REAL*4    frame_cs_length(10)
REAL*4    frame_cs_thick(10)
REAL*4    frame_cs_width(10)
REAL*4    frame_length(10)            Length of frame
REAL*4    frame_si_length(10)
REAL*4    frame_si_thick(10)
REAL*4    frame_si_width(10)
REAL*4    frame_thick(10)             Frame thickness
REAL*4    frame_width(10)             Frame width
REAL*4    layer_back(10)              Layer back
REAL*4    layer_lateral(10)           Lateral offset for ladder
REAL*4    layer_longi(10)             Longitudinal offset for ladder
REAL*4    layer_phioff(10)            Phi offset for 1st ladder of layer
REAL*4    layer_radial(10)            Radial offset (wafer staggering)
REAL*4    layer_radius(10)            Radius of layers
REAL*4    layer_shape(10)             shape id for layer
REAL*4    size(3) TUBE                Inner,outer radius and half lenght
REAL*4    water_channel_thick(10) Water channel thickness
REAL*4    water_channel_width(10)
REAL*4    x(3)                        Position of the SVT in STAR
```

3.11 SVT_TEFF

3.11.1 Package: STE

3.11.2 Description: SVT Tracking Efficiency

This table includes variables used to evaluate the track reconstruction effiency of the SVT tracker.

```
Type      Name     Comment
=======================================================
        INTEGER*4 nctrk(7)   number of candidate trks with nspt=i
        INTEGER*4 nctrktot   total number of trk candidates
        INTEGER*4 ndiff(7)   number of tracks with nrspt-ngspt
        INTEGER*4 ngtrk(7)   number of trks with i good spts
        INTEGER*4 ngtrktot   total number of perfect reconstruction
        INTEGER*4 nrtrk(7)   number mc trks with i spts
        INTEGER*4 nrtrktot   total number of mc tracks
        INTEGER*4 nspt(6)    number of spts on layer i
        INTEGER*4 nsptnot(6) number of spts not assigned on layer i
        INTEGER*4 nspttot    total number of spts
```

3.12 SVT_TEFF

3.12.1 Package: STE

3.12.2 Description: SVT Tracking Efficiency

This table includes variables used to evaluate the momentum resolution of the SVT tracker.

```
Type      Name     Comment
=======================================================
        INTEGER*4 id       sequential index
        INTEGER*4 ncspt    number of candidate spts on track
        INTEGER*4 ngspt    number of correct assignment (spt)
        INTEGER*4 nrspt    number of real spts on track
        INTEGER*4 pid      particle id
        REAL*4    dp(3)    difference real/calculated momentum
        REAL*4    dph      difference real/calculated
        REAL*4    dpt      diff pt
        REAL*4    dptot    diff ptot
        REAL*4    drp(3)   reconstructed track p(3)
        REAL*4    drph     reconstructed track   ph
        REAL*4    drpt     reconstructed track   pt
        REAL*4    drptot   reconstructed track   ptot
        REAL*4    drsecrec trk primary or secondary
        REAL*4    drth rec trk th
        REAL*4    dsec     difference MC/reconstructed origin
        REAL*4    dth      difference MC/calculated th
        REAL*4    eta      pseudo rapidity
```

```
REAL*4    p(3)     particle 3-momentum
REAL*4    ph       real angle (azimuthal)
REAL*4    pt       transverse momentum
REAL*4    ptot     total momentum
REAL*4    sec      1 = primary; -1 = secondary; nature of MC track
REAL*4    th       polar angle of MC track
REAL*4    y        rapidity
```

4. USAGE

4.1 Introduction

The tracking code currently includes two branches: DIRECT and FILLER. The DIRECT branch is meant for Monte Carlo studies only. In this branch, tracks are reconstructed using the link to the MKINE table carried in the SVT_SPT table. In other words, the program looks at the answer to build a list of space points for track "candidates". Such an approach permits to study in detail the effects of the geometry and the multiple coulomb scattering on the occupency, the number of space points per track and most importantly the momentum resolution of the detector on its own.

The FILLER branch is a "follow your nose" approach to track finding. For every space points on the first SVT layer, it connects to space points on the second and third layers if these lie within a cone of adjustable dimension.

A third branch, GROUPER, is also available [1]. This approach uses a simple radial mapping technique and tries to find groups of space points in the mapped space.

If the user wishes to call the SVT Tracking Evaluator (STE), he (she) must first call the DIRECT branch and then call the FILLER or GROUPER branch. The recommanded mode of operation is illustrated in Fig. 5.

Apart from the generation of the space point lists SVT_DTRACK and SVT_CTRACK which require for obvious reasons two different approaches and input data, the two branches are quite similar in functionality. The fitting of tracks to reconstruct helix parameters, in particular, is accomplished with identical routines with the same input and output tables. However, for technical reason, we use two different routine names in the two branches: SVT_TRACKING_DFIT (direct branch) and SVT_TRACKING_FIT (filler or grouper branch). So, in order to have the same tables as input and output arguments (which simplifies the maintenance of these two routines) we use two copying routines: one to go from the SVT_DTRACK table to the SVT_CTRACK table before a call to the fitting routine and another to go from the SVT_TRACK table to the SVT_MCTRACK after the call to the fitting routine.

In order to provide the capability for the user to study the effects of the Coulomb multiple scattering and the effects of the finite detector resolution, we have included two routines for the generation of the space points table SVT_SPT in addition to the SVT fast resolution simulator. They are called SVT_SPT_DIRECT and SVT_SPT_DIRECT_GS. See their description below.

Currently, as no vertex finding routines is fully functional in TAS, we have implemented a fake vertex finding routine. Rather than finding the primary vertex this routine actually smears the vertex position used by GEANT with a Gaussian distribution.

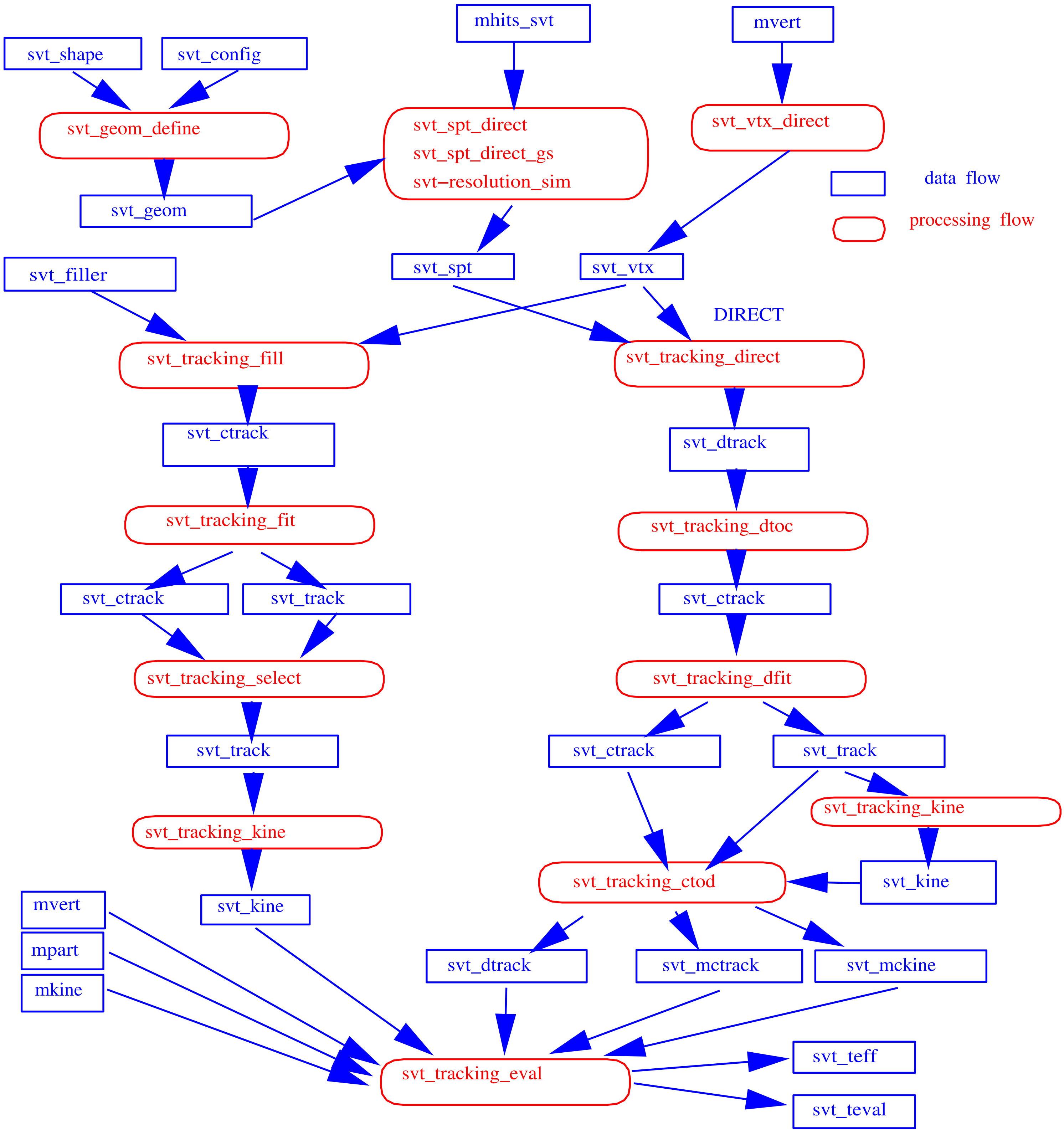


Fig. 5: Parametrization of the wafer position and orientation

Finally, a tracking evaluator has been written and is described in details below.

In the following, we give a brief description of the name, the functionality and the calling sequence of all routines used in the STK package. We also include for each routine the recommanded mid calling sequence and whenever appropriate a kumac list describing the setting of input/control parameters.

4.2 SVT_VERTEX_DIRECT

4.2.1 Description:

This module fills the table SVT_VTX with the postion of the primary vertex used by geant/monte-carlo (stored in table MVERT). The position is smeared along the 3 axis with a gaussian random distribution using parameters from table SVT_VTX_DIRECT.

4.2.2 Entity Relationship Diagram:

4.2.3 Input/Ouput Tables:

```
mvert          : GEANT Vertex table
svt_vtx        : Smeared vertex
svt_vtx_direct : Smearing Parameters
```

4.2.4 MID file:

```
table_lib mct
table_lib stk
table_lib svr
#
module svt_vertex_direct svt_vertex_direct
#
module_table svt_vertex_direct mvert          read   1
module_table svt_vertex_direct svt_vtx        read   2
module_table svt_vertex_direct svt_vtx_direct update 3
```

4.2.4 Parameter Definition (KUMAC file):

```
tas/set/value svt_vtx_direct(1).sx(1)  0.005   ! 50 microns
tas/set/value svt_vtx_direct(1).sx(2)  0.005   ! 50 microns
tas/set/value svt_vtx_direct(1).sx(3)  0.005   ! 50 microns
```

4.3 SVT_SPT_DIRECT

4.3.1 Description:

Makes a straight copy of the GEANT space point informations in table MHITS_SVT to the "reconstructed" space point table SVT_SPT. No smearing in this copy - error on the space points is therefore recorded as zero. The dedx information is copied as is - no smearing, no calibration or gain adjustments made.

4.3.2 Entity Relationship Diagram:

4.3.3 Input/Ouput Tables:

```
mhits_svt     : GEANT Hits in the SVT
svt_spt       : SVT Space Points Table
```

4.3.4 MID file:

```
table_lib scs
table_lib mct
#
module svt_spt_direct  svt_spt_direct
#
module_table svt_spt_direct mhits_svt    read  1
module_table svt_spt_direct svt_spt      write 2
```

4.4 SVT_SPT_DIRECT_GS

4.4.1 Package: SRS

4.4.2 Description:

Makes a smeared copy of the GEANT space point informations in table MHITS_SVT to the "reconstructed" space point table SVT_SPT. Gaussian smearing is performed in this copy - error on the space points is recorded as the standard deviation of the smearing. The dedx information is copied as is - no smearing, no calibration or gain adjustments made. The smearing is performed in the wafer plane; i.e. a mapping to the reference frame of the wafer is performed (by svtgtol) smearing is done in both the transverse and drift directions and the point is mapped back to the STAR global coordinate system (svtltog).

4.4.3 Input/Ouput Tables:

```
srs_direct     : Space Point Smearing Parameters
svt_geom       : SVT Wafer geometry
mhits_svt      : GEANT output SVT Hits
svt_spt        : SVT Reconstructed Space Points
```

4.4.4 MID file:

```
table_lib scs
table_lib mct
table_lib svg
table_lib srs
#
module svt_spt_direct_gs  svt_spt_direct_gs
#
module_table svt_spt_direct_gs srs_direct   read  1
module_table svt_spt_direct_gs svt_geom     read  2
module_table svt_spt_direct_gs mhits_svt    read  3
module_table svt_spt_direct_gs svt_spt      write 4
```

4.5 SVT_TRACKING_DIRECT

4.5.1 Package: STK

4.5.2 Description:

This module copies the table svt_dtrack to the table svt_ctrack with no modification of its content.

4.5.3 Entity Relationship Diagram:

4.5.4 Input/Ouput Tables:

```
svt_geom        : SVT Wafer geometry
svt_spt         : SVT Reconstructed Space Points
svt_vtx         : Smeared vertex
svt_dtrack      : List of MC track space points
```

4.5.5 MID file:

```
module svt_tracking_direct   svt_tracking_direct
#
module_table svt_tracking_direct svt_geom       read  1
module_table svt_tracking_direct svt_spt        read  2
module_table svt_tracking_direct svt_vtx        read  3
module_table svt_tracking_direct svt_dtrack     write 4
```

4.6 SVT_TRACKING_DFIT

4.6.1 Package: STK

4.6.2 Description:

Perform a fit of tracks in the SVT. A pure helicoidal movement is assumed so the movement along the z-axis is decoupled from the movement along a circle. The circle fit is performed using a conformal mapping technique which assumed that the circle passes through or close to the origin allowing a reduction of the problem to a linear fit.

This version assumes the tracks pass through the primary vertex. A linear translation is performed on all SVT points to force the main (primary) vertex to be on the circle. The space points are in the table svt_spt. The table svt_ctrack is a list, for each track (candidate) of the associated space points. The vertex is therefore not included explicitely in the fit as it would result in a division by zero in the conformal mapping. It is however included in the straight line fit (z vs s).

4.6.3. Entity Relationship Diagram:

4.6.4 Input/Ouput Tables:

```
svt_spt         : SVT Reconstructed Space Points
svt_vtx         : Smeared vertex
svt_ctrack      : List of track candidates
svt_track       : Helix parameters of fitted candidates
```

4.6.5 MID file:

```
module svt_tracking_dfit   svt_tracking_dfit
#
module_table svt_tracking_dfit svt_spt        read  1
module_table svt_tracking_dfit svt_vtx        read  2
module_table svt_tracking_dfit svt_ctrack     read  3
module_table svt_tracking_dfit svt_track      write 4
```

4.7 SVT_TRACKING_CTOD

4.7.1 Package: STK

4.7.2 Description:

This module copies the table svt_track, svt_ctrack, svt_kine to the table svt_mctrack, svt_dtrack and svt_mckine respectivley with no modification of their content.

4.7.3 Entity Relationship Diagram:

4.7.4 Input/Ouput Tables:

```
svt_ctrack     : List of track candidates
svt_track      : Helix parameters of fitted candidates
svt_kine       : Reconstructed kinematic for selected candidates
svt_dtrack     : List of MC track space points
svt_mctrack    : Helix parameters of fitted MC tracks
svt_mckine     : Reconstructed kinematic for MC tracks
```

4.7.5 MID file:

```
module svt_tracking_ctod   svt_tracking_ctod
#
module_table svt_tracking_ctod svt_ctrack       read  1
module_table svt_tracking_ctod svt_track        read  2
module_table svt_tracking_ctod svt_kine         read  3
module_table svt_tracking_ctod svt_dtrack       write 4
module_table svt_tracking_ctod svt_mctrack      write 5
module_table svt_tracking_ctod svt_mckine       write 6
```

4.8 SVT_TRACKING_FILLER

4.8.1 Package: STK

4.8.2 Description:

This module fills the table SVT_CTRACK with all possible combinations of svt space points falling into a cone - It has to be followed by a call to svt_tracking_fit to perform a layer to layer helix fit assuming a 3 layer design with the appropriate MCS. The space points will be used only if they have not been assigned previously to a track by another module (like the grouper).

4.8.3 Entity Relationship Diagram:

4.8.4 Input/Ouput Tables:

```
svt_spt        : SVT Reconstructed Space Points
svt_vtx        : Smeared vertex
svt_ctrack     : List of track candidates
svt_filler     : Tracking Finding Parameters
svt_config     : SVT Configuration Table
```

4.8.5 MID file:

```
module svt_tracking_fill   svt_tracking_fill
#
module_table svt_tracking_fill svt_spt        read   1
module_table svt_tracking_fill svt_vtx        read   2
```

```
module_table svt_tracking_fill svt_ctrack     write  3
module_table svt_tracking_fill svt_filler     update 4
module_table svt_tracking_fill svt_config     update 5
```

4.8.6 Parameter Definition (KUMAC file):

```
set/value svt_filler(1).d12   4.
set/value svt_filler(1).d23   3.
set/value svt_filler(1).thx12 0.5
set/value svt_filler(1).thx23 0.5
set/value svt_filler(1).thy12 0.5
set/value svt_filler(1).thy23 0.5
set/value svt_filler(1).thz12 0.5
set/value svt_filler(1).thz23 0.5
```

4.9 SVT_TRACKING_FIT

4.9.1 Package: STK

4.9.2 Description:

Perform a fit of track (candidates) in the SVT. A pure helicoidal movement is assumed so the movement along the z-axis is decoupled from the movement along a circle. The circle fit is performed using a conformal mapping technique which assumed that the circle passes through or close to the origin allowing a reduction of the problem to a linear fit.

This version assumes the tracks pass through the primary vertex. A linear translation is performed on all SVT points to force the circle the main (primary) vertex to be on the circle. The space points are in the table svt_spt. The table svt_ctrack is a list, for each track (candidate) of the associated space points. The vertex is therefore not included explicitely in the fit as it would result in a division by zero in the conformal mapping. It is however included in the straight line fit.

4.9.5 Entity Relationship Diagram:

4.9.4 Input/Ouput Tables:

```
svt_spt        : SVT Reconstructed Space Points
svt_vtx        : Smeared vertex
svt_ctrack     : List of track candidates
svt_track      : Helix parameters of fitted candidates
```

4.9.5 MID file:

```
module svt_tracking_fit   svt_tracking_fit
#
module_table svt_tracking_fit svt_spt        read  1
module_table svt_tracking_fit svt_vtx        read  2
module_table svt_tracking_fit svt_ctrack     read  3
module_table svt_tracking_fit svt_track      write 4
```

4.10 SVT_TRACKING_SELECT
4.10.1 Package: STK
4.10.2 Description:
Selects among the tracks sharing a common first layer point the track with the lowest chi-square.
4.10.3 Entity Relationship Diagram:
4.10.4 Input/Ouput Tables:

```
svt_ctrack     : List of track candidates
svt_track      : Helix parameters of fitted candidates
```

4.10.5 MID file:

```
module svt_tracking_select svt_tracking_select
#
module_table svt_tracking_select   svt_ctrack  update 1
module_table svt_tracking_select   svt_track   update 2
```

4.11 SVT_TRACKING_KINE
4.11.1 Package: STK
4.11.2 Description:
Calculates the kine observables of all reconstructed SVT tracks. Currently, all tracks are assumed to originate from the main vertex. So the py, pz and ph angle are calculated after extrapoaltion to the main vertex.
4.11.3 Entity Relationship Diagram:
4.11.4 Input/Ouput Tables:

```
svt_vtx        : Smeared vertex
svt_track      : Helix parameters of fitted candidates
svt_kine       : Reconstructed kinematic for selected candidates
```

4.11.5 MID file:

```
module_table svt_tracking_kine svt_vtx        read  1
module_table svt_tracking_kine svt_track      read  2
module_table svt_tracking_kine svt_kine       write 3
```

4.12 SVT_TRACKING_EVAL
4.12.1 Package: STE
4.12.2 Description:
This routine evaluates the tracking reconstruction performance of the STK package. It proceeds in two steps: 1- In a first steps, the routine search for the percentage of well reconstructed tracks by looking at the number of points on each reconstructed tracks that actually belong to the monte carlo track. It evaluates the number of well reconstructed tracks relative to the number of "real" tracks. etc etc 2- In a second step, the routine evaluates the momentum resolution of the tracks reconstructed with a minimum success.
4.12.3 Entity Relationship Diagram:
4.12.4 Input/Ouput Tables:

```
mvert           : GEANT Vertex table
svt_vtx         : Smeared vertex
svt_vtx_direct  : Smearing Parameters
mpart           : GEANT Particle declarations
mkine           : GEANT Kine Bank
mvert           : GEANT Vertex table
mhits_svt       : GEANT output SVT Hits
svt_spt         : SVT Reconstructed Space Points
svt_vtx         : Smeared vertex
svt_dtrack      : List of MC track space points
svt_mctrack     : Helix parameters of fitted MC tracks
svt_mckine      : Reconstructed kinematic for MC tracks
svt_ctrack      : List of track candidates
svt_track       : Helix parameters of fitted candidates
svt_kine        : Reconstructed kinematic for selected candidates
svt_teval       : Evaluation Table - Resolution
svt_teff        : Evaluation Table - Efficiency
```

4.12.5 MID file:

```
module svt_tracking_eval   svt_tracking_eval
#
module_table svt_tracking_eval     mpart       read   1
module_table svt_tracking_eval     mkine       read   2
module_table svt_tracking_eval     mvert       read   3
module_table svt_tracking_eval     mhits_svt   read   4
module_table svt_tracking_eval     svt_spt     read   5
module_table svt_tracking_eval     svt_vtx     read   6
module_table svt_tracking_eval     svt_dtrack  read   7
module_table svt_tracking_eval     svt_mctrack read   8
module_table svt_tracking_eval     svt_mckine  read   9
module_table svt_tracking_eval     svt_ctrack  read  10
module_table svt_tracking_eval     svt_track   read  11
module_table svt_tracking_eval     svt_kine    read  12
module_table svt_tracking_eval     svt_teval   write 13
module_table svt_tracking_eval     svt_teff    write 14
```

4.13 SVT_GEOM_DEFINE

4.13.1 Package: SVG

4.13.2 Description:

This module defines the svt geometry for an arbitrary number of layers, ladder/layers, wafer/ladder. The thickness of the detector and frame/misc material is also ajustable. The generation of the geometry (SVT_GEOM table) is driven by the configuration table SVT_CONFIG and the subsidiary wafer shape table SVT_SHAPE.

4.12.3 Entity Relationship Diagram:

4.12.4 Input/Ouput Tables:

```
svt_config  :  SVT Configuration Table
svt_shape   :  SVT Wafer Shape and Dimension Definition
svt_geom    :  SVT Wafer Geometry: Positions and Orientation
```

4.12.5 MID file:

```
#
table_lib svg
#
module svt_geom_define svt_geom_define
#
module_table svt_geom_define svt_config   read 1
module_table svt_geom_define svt_shape    read 2
module_table svt_geom_define svt_geom    write 3
```

6. Acknowledgement

I would like to thank R. Bellwied, J. Bielecki, S. Margetis, S. Paganis, D. Prindle, G. Roger and K. Wilson for numerous discussions, their help and suggestions.

7. References

1. D.J. Prindle, "SVT Event Reconstruction Algorithms", STAR World Wide Web Documentation.
2. P. Jones,I. Sakreda, STAR Note 56, "The STAR FaSt TPC Simulation (A User's Guide)", also look for documentation under STAR World Wide Web.
3. J. Bielecki, STAR Note 31, M.Sc. Thesis, Wayne State University, "A Helix Track Fitting Program for the Silicon Vertex Tracker (SVT) in the STAR Experiment"
4. E. Paganis, S. Margetis, STAR Note 126, "dE/dx Information with the SVT"
5. K. Wilson, STAR World Wide Web Documentation, "Resolution Simulator".