

Progress on the RAVE / VERTIGO Vertex Reconstruction Toolkit

**Winfried A. Mitaroff,
Wolfgang Waltenberger
and Fabian Moser**

Institute of High Energy Physics
Austrian Academy of Sciences
Nikolsdorfer Gasse 18
A-1050 Vienna, Austria, Europe



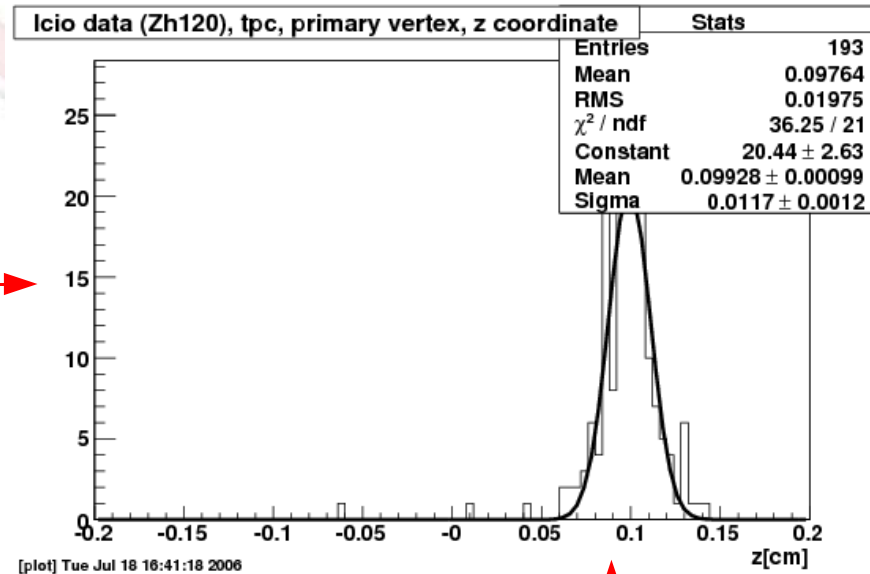
A detector-independent toolkit (RAVE) is being developed for the reconstruction of the common interaction vertex or vertices from a set of fitted tracks, along with a supplementary framework (VERTIGO) for testing, analyzing, and debugging. Main design goals are ease of use, high integrability into existing software, extensibility, and openness. The toolkit is coded in C++, with interfaces for Java and Python. A beta release is available.



Test of RAVE embedded in Marlin

Rave as an ILC "MarlinProcessor", fitting primary vertices from TPC Tracks.

z coordinate
of primary
vertices

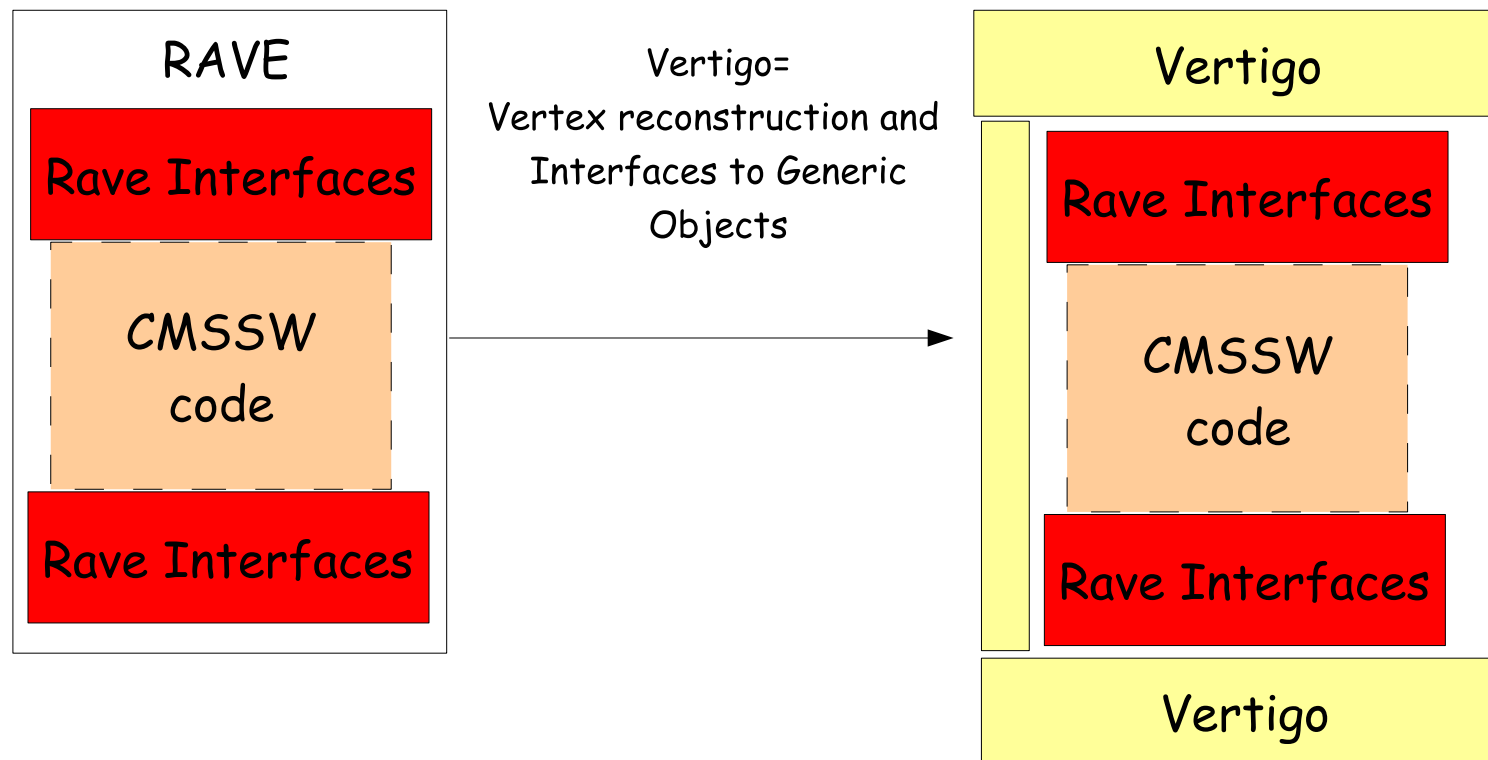


1 mm shift in z - a bug in the ILC
track reconstruction found by rave!

Reconstructed z coordinate of adaptively fitted primary vertices generated at (0, 0, 0).
The shift of 1 mm hinted to a Marlin bug, meanwhile corrected by Steve Aplin.



Step 3: creation of the stand-alone framework

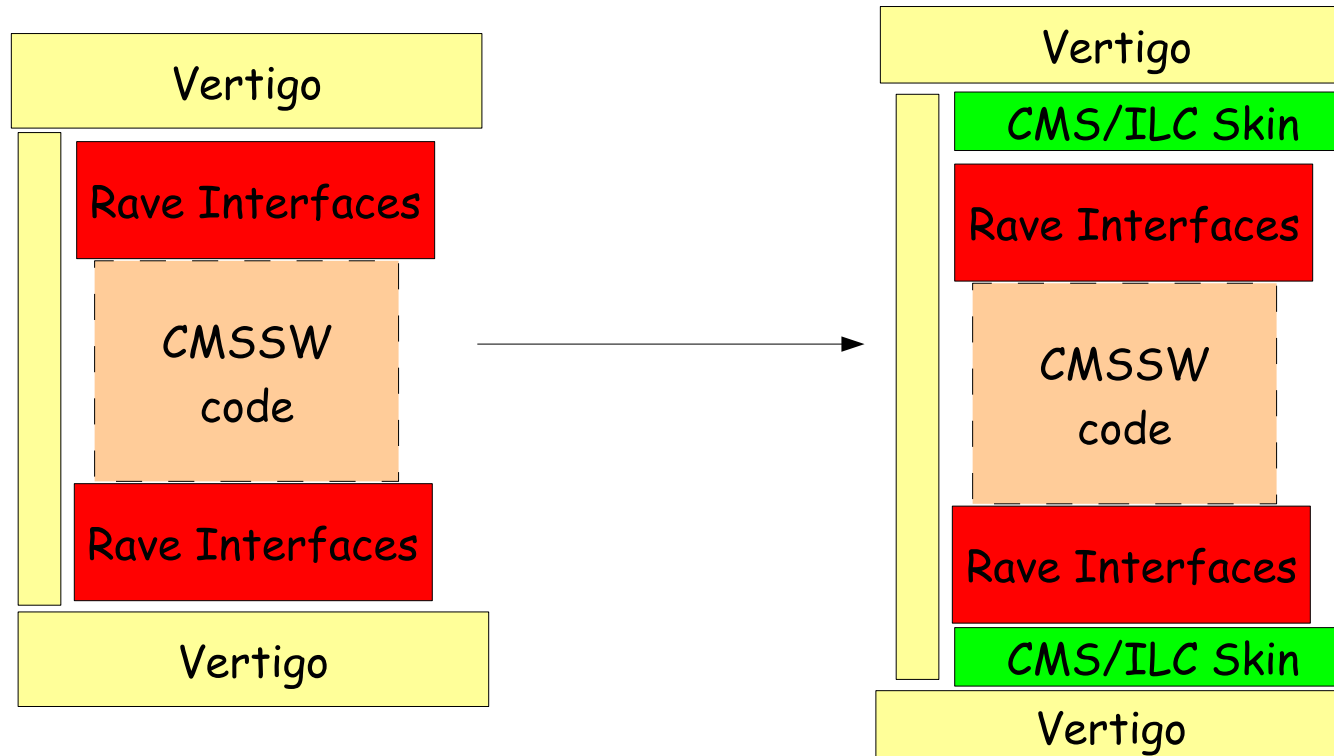


Status:

- Finished; served as a “test bed” as long as the CMSSW framework was not fully operational.



Step 4: VERTIGO emulating detector setups



Status:

- Full CMS skin (“native” , i.e. no parameter conversions needed): tested with input from the Data Seeder;
- Simple LDC skin ($B_z = 4T$, no material effects): tested with input from the LCIO Event Generator (switch for standard “L3” or temporary “BRAHMS” track fit parameter representations, respectively);
- Simple SiD skin (so far identical to LDC, except $B_z = 5T$).



Examples of RAVE & VERTIGO APIs

Calling RAVE from a C++ environment:

Easiest use case:

```
rave::Factory factory (MyMagneticField(), MyPropagator());
```

```
std::vector < rave::Vertex > vertices = factory.create( tracks,  
"method" );
```

User must implement interfaces

Factory creates vertices from tracks, using method "method"

A Python interface exists for VERTIGO:

```
import vertigo
```

```
eventfactory=vertigo.EventFactory("lcio:tracks.slcio" )
```

```
ravefactory=vertigo.RaveFactory()
```

```
visualiser=vertigo.Visualiser()
```

instantiate, data source
is an lcio file

```
for event in eventfactory:
```

```
vertices=ravefactory.create ( event.tracks(), "avf" )
```

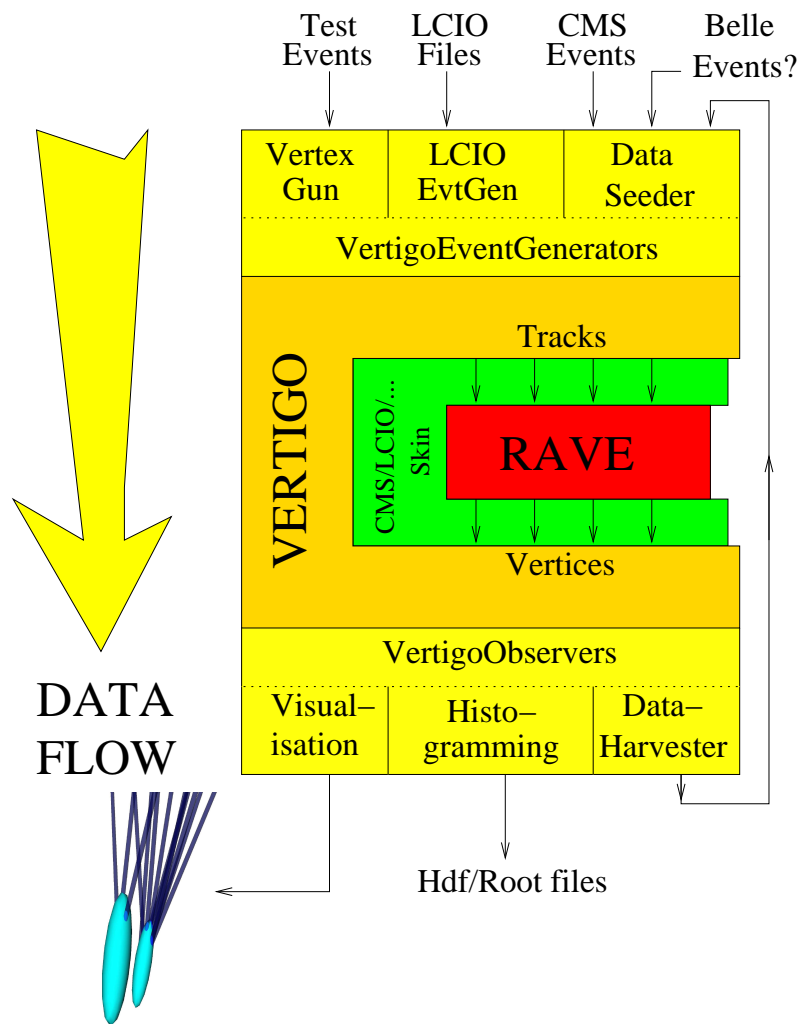
```
event.add ( vertices )
```

```
visualiser ( event )
```

fit tracks with adaptive "avf"
method, add vertices to event,
and visualise.



VERTIGO analysis, debugging and I/O tools

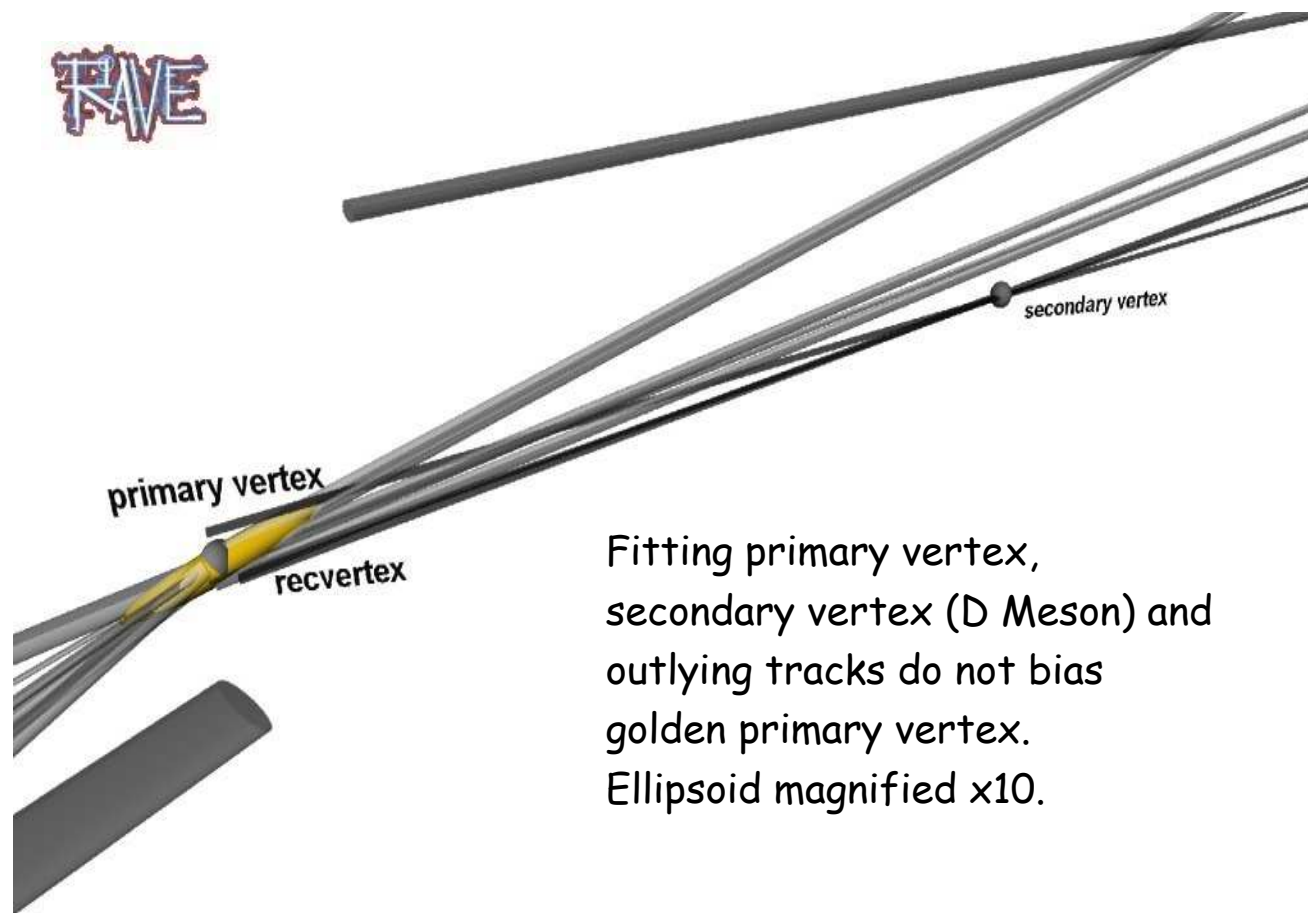


Interfaces & tools:

- Comparable with vertexing in ORCA / CMSSW;
- Input = "Event Generators":
 - Vertex Gun (generating artificial events),
 - LCIO Event generator (for LCIO data),
 - Data Seeder (flexible general input format);
- Output = "Observers":
 - Visualization (VIS),
 - Histogramming,
 - Data Harvester (complement to the Seeder).



Show adaptive vertex fitter (example 1)



Fitting primary vertex,
secondary vertex (D Meson) and
outlying tracks do not bias
golden primary vertex.
Ellipsoid magnified x10.

ORCA 8.2.0 $c\bar{c}$ event, the primary vertex (gold) fitted adaptively.



Outlook to the future

Near future:

- Embedding RAVE into the new Marlin version 00-09-05 (standard “L3” track fit parameters and error matrix);
- Refinement of the VERTIGO skins for the LDC and SiD detectors;
- A simple VERTIGO skin for the BELLE detector (input of “Panther tables” as Ntuples via the Data Seeder);

Mid-term future:

- Embedding RAVE into the org.lcsim (Java based) reconstruction software by means of a C++ wrapper (SWIG);
- Embedding RAVE into the BASF (C++ based) reconstruction software of the BELLE detector.

Long-term future:

- Re-coding of RAVE in Java 5.0, if the C++ wrapper solution turns out to be inefficient; manpower problem ?

Maintenance & development:

- The HEPHY Institute of Vienna is committed to the maintenance, documentation and distribution of the RAVE and VERTIGO packages;
- The RAVE toolkit will permanently be augmented by new high-quality algorithms; a first candidate being ZvTop by David Jackson, Ben Jeffery, Jan Strube et al.;
- Other HEP experiments using a tracking detector are welcome to contribute with a VERTIGO skin of their own, or alternatively to embed RAVE into their reconstruction software;



Contributors and References

The Vienna team:

- [Wolfgang Waltenberger](#) (the “mastermind”): core algorithms, tools, CMS interface, Java integration;
- [Winfried Mitaroff](#): BELLE and ILC (JAS3/org.lcsim) interfaces, Java integration;
- [Fabian Moser](#) (undergraduate student, until Aug. 2006): core algorithms, tools, ILC (Marlin.Reco) interface;
- [Gerald Richter](#) (graduate student): tools, Python support.

External contributors:

- The CMS “vertexing circle” community (Brussels, CERN, Lyon, Zurich, . . .);
- welcome: the ILC “ZvTop vertexing” communities (Ben Jeffery, Jan Strube) 😊

Spreading the word:

- [Presentations 2004](#): 7th Int. LCWS (April, Paris), CHEP '04 (Sept., Interlaken) \implies proceedings available:
<http://www-flc.desy.de/lcnotes/notes/LC-T00L-2004-017.pdf>
<http://doc.cern.ch/archive/cernrep/2005/2005-002/p272.pdf>
- Development was stalled in 2005, mainly because of Wolfgang’s one-year civil service, and Winfried having been busy with organizing the 3rd ECFA-ILC Workshop (November, Vienna);
- [Presentations 2006](#): ILC Software Workshop (April, Cambridge), SiLC Collaboration Meeting (June, Liverpool), Wolfgang’s visit at SLAC (Oct.), IEEE Nuclear Science Symposium (Nov., San Diego).

Where to look:

- <http://forum.linearcollider.org/> \implies Analysis Tools \implies RAVE/VERTIGO thread.

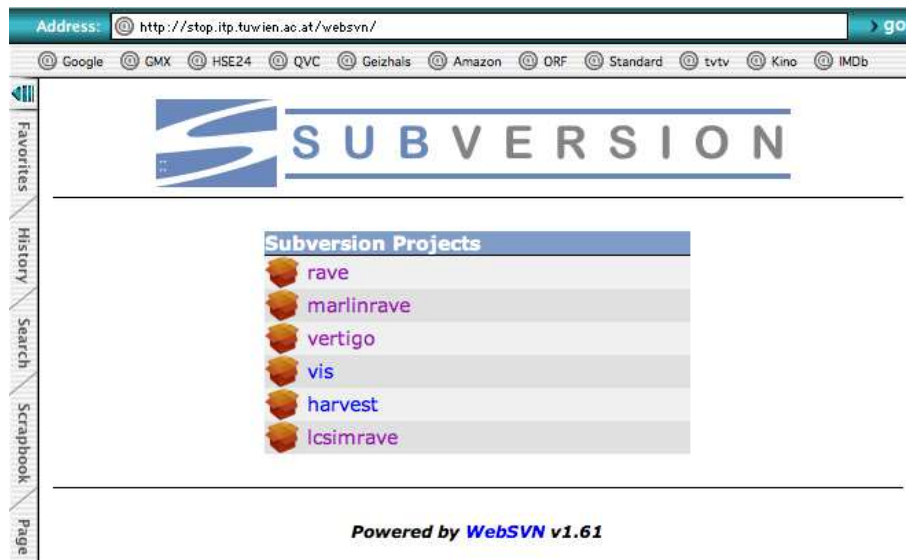


Source Codes and Docus

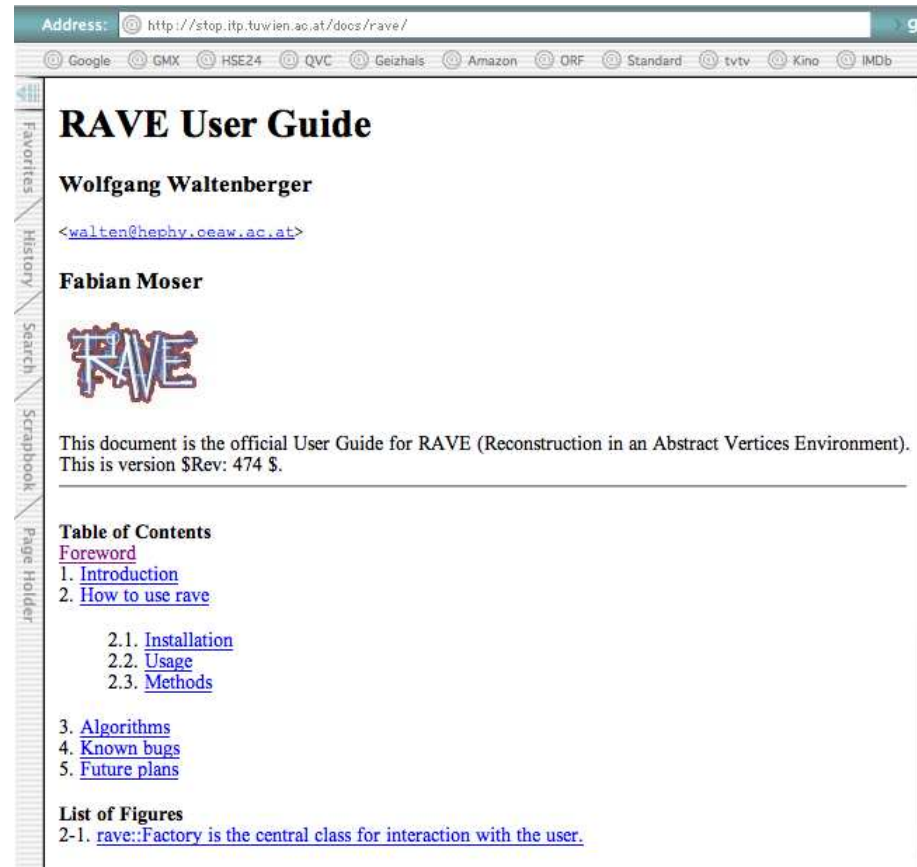
The Vienna subversion repository:

<http://stop.itp.tuwien.ac.at/websvn/>

<http://stop.itp.tuwien.ac.at/docs/rave/>



A screenshot of a web browser showing the Subversion website. The address bar contains <http://stop.itp.tuwien.ac.at/websvn/>. The page features a large 'SUBVERSION' logo with a stylized 'S' icon. Below the logo is a table titled 'Subversion Projects' listing several projects with orange folder icons: rave, marlinrave, vertigo, vis, harvest, and lcsimrave. At the bottom of the page, it says 'Powered by WebSVN v1.61'. The browser's sidebar shows 'Favorites', 'History', 'Search', 'Scrapbook', and 'Page Holder'.



A screenshot of a web browser showing the RAVE User Guide. The address bar contains <http://stop.itp.tuwien.ac.at/docs/rave/>. The page title is 'RAVE User Guide' by Wolfgang Waltenberger, with contact information [<walten@hephy.ceaw.ac.at>](mailto:walten@hephy.ceaw.ac.at) and Fabian Moser. A stylized 'RAVE' logo is displayed. The text states: 'This document is the official User Guide for RAVE (Reconstruction in an Abstract Vertices Environment). This is version \$Rev: 474 \$.' Below this is a 'Table of Contents' with links for: Foreword, 1. Introduction, 2. How to use rave (with sub-links for 2.1. Installation, 2.2. Usage, 2.3. Methods), 3. Algorithms, 4. Known bugs, and 5. Future plans. A 'List of Figures' section includes a link for 2-1. [rave::Factory](#) is the central class for interaction with the user. The browser's sidebar shows 'Favorites', 'History', 'Search', 'Scrapbook', and 'Page Holder'.



The Vertexing Circle's credo

"The method of Least Squares is seen to be our best course when we have thrown overboard a certain portion of our data -- a sort of sacrifice which has often to be made by those who sail the stormy seas of Probability."

F. Y. Edgeworth, 1887

"Let us not throw away data all too hastily. Instead, let us weight and reweight the data, consider and reconsider alternative models. Only if we must, at the latest possible stage, shall we distinguish between 'in' and 'out', between signal and noise."

The CMS vertexing circle, 2004



Representative robust algorithms

- AdaptiveVertexFitter (AVF):

A robust generalization of the Kalman filter, iteratively downweighting the contribution of outlier tracks to the objective function (“soft assignment”). The extra weights w_i on the reduced residuals r_i are calculated by a Fermi function with cutoff parameter r_{cut} . In addition, a deterministic annealing schedule with decreasing “temperature” T is introduced in order to avoid falling into local minima:

$$w_i(r_i, T) = \frac{e^{-r_i^2/2T}}{e^{-r_i^2/2T} + e^{-r_{cut}^2/2T}} = \frac{1}{1 + e^{(r_i^2 - r_{cut}^2)/2T}}$$

Iterations (index k , omitted above) start with $w_{i,1} = 1$ (least-squares). For $k > 1$, the $w_{i,k} = w_i(r_{i,k-1}, T_k)$, with $T_k \leq T_{k-1}$ defined by the annealing schedule. For $T \rightarrow 0$, the Fermi function approximates the Heaviside function, and the assignment turns into a “hard” one ($w_i = 1$ or 0).

- MultiVertexFitter (MVF):

This robust estimator is a generalized AVF, simultaneously fitting n vertices by “soft assignment” of each track to more than one vertex. The extra weights w_{ij} on the reduced residuals r_{ij} w.r.t. vertex j are

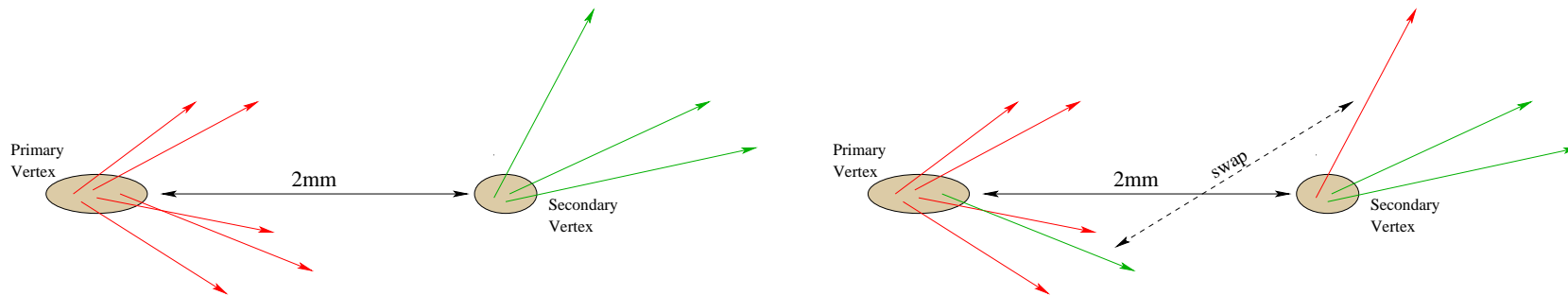
$$w_{ij}(r_{ij}, T) = \frac{e^{-r_{ij}^2/2T}}{\sum_{\ell=1}^n e^{-r_{i\ell}^2/2T} + e^{-r_{cut}^2/2T}}$$



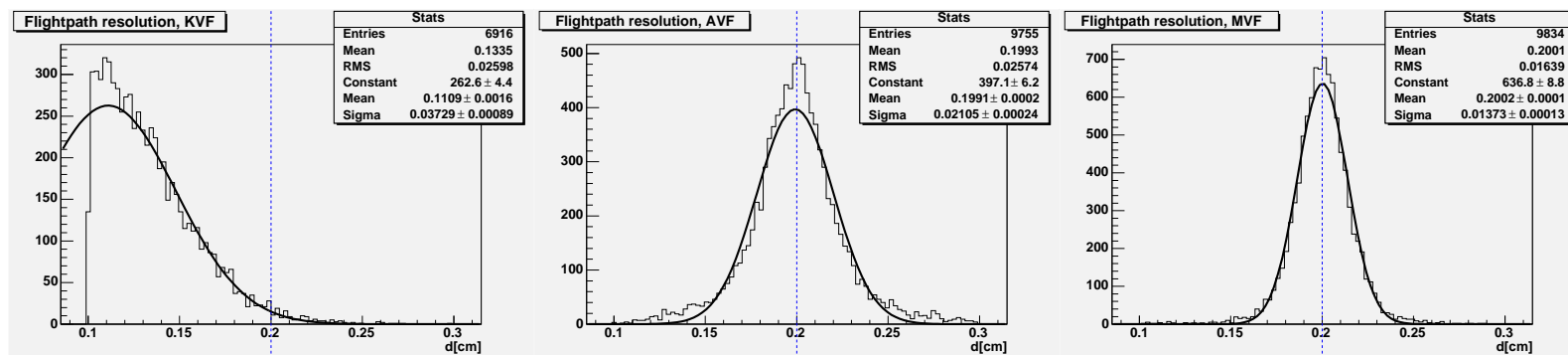
Performance of the MVF

Generated events: PV at (0, 0, 0) cm with 5 tracks, forming a jet of total momentum (0, 25, 25) GeV and opening angle 0.5 rad; one SV at (0, 0, 0.2) cm with 3 tracks, total jet momentum (-15, 0, 20) GeV and opening angle 0.5 rad. Track errors Gaussian and correctly described by the tracks' covariance matrices.

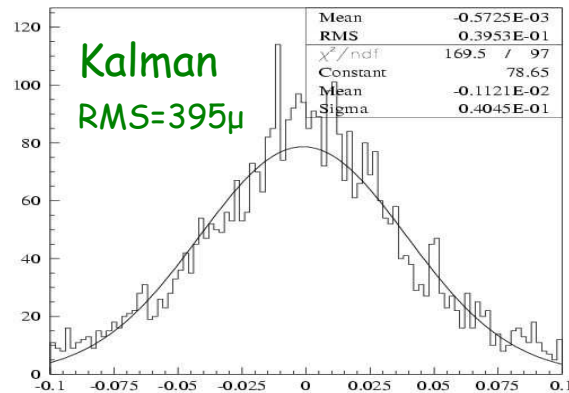
Thereafter, one PV track and one SV track were chosen to be “swapped”, i.e. assigned to the wrong vertex.



The two track bundles, each containing one wrongly assigned track, were submitted to a Kalman filter, the AVF and the MVF. Resolutions of the reconstructed decay lengths are shown below:

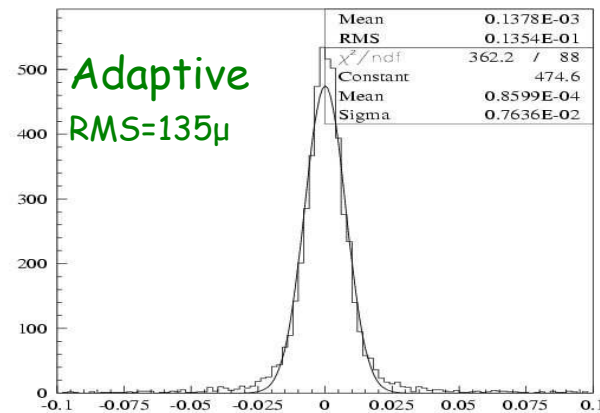
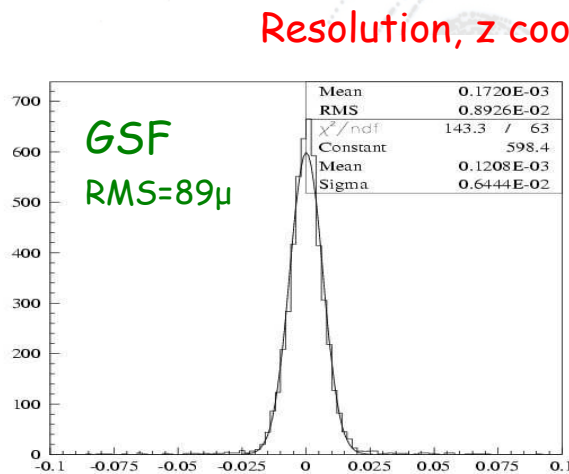


The Gaussian Sum Vertex Fitter (GSF)



GaussianSumVertexFitter(2)

Artificial data, 4 tracks from a primary vertex, 1 track from a secondary vertex (3mm away). Tracks smeared with two Gaussians, "tail Gaussian" has a weight of 10 %, and is 10 times wider. GSF "sees" the right mixture; KVF and AVF see only core component.

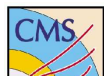


Wolfgang Waltenberger

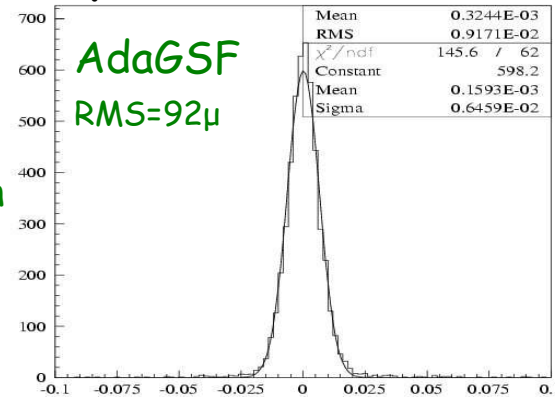
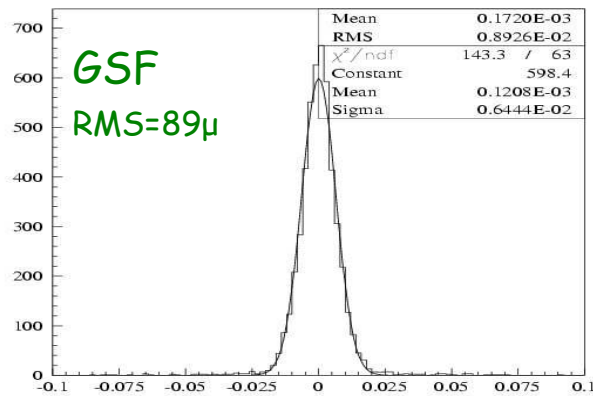
12



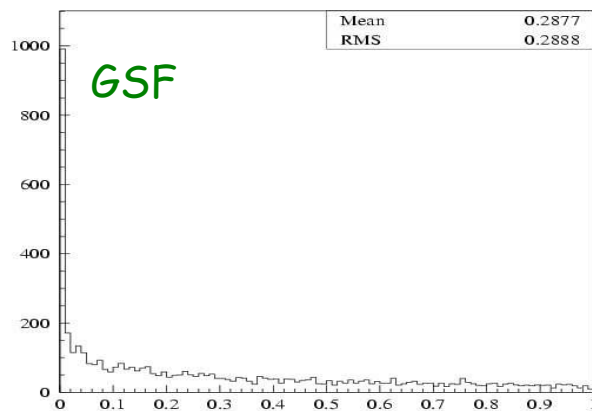
Combination of the AVF and the GSF



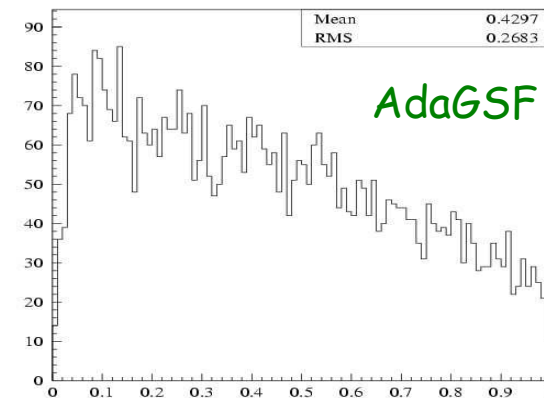
The **adaptive** method and the **Gaussian-Sum** technique can actually be **combined**! Proof of concept:



resolution
z coord

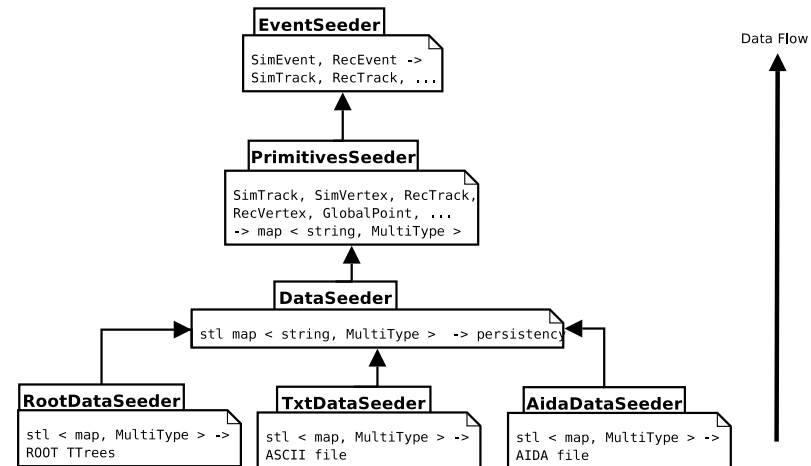
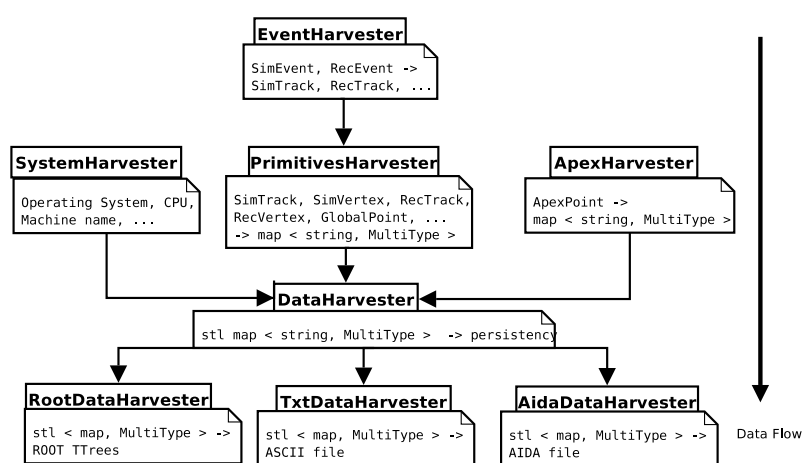


χ^2 -prob



VERTIGO Data Harvester & Seeder

- A persistency storage solution was originally realized on top of ROOT; it is currently being extended by more standard-compliant alternatives (AIDA and XML). DataSources include a “vertex gun”, LCIO, etc.
- All I/O is handled through a “data harvesting” concept (which may possibly be integrated as front/end in AIDA): object → STL map → ASCII/ROOT/AIDA file (Harvester) and vice versa (Seeder).
- The STL mapping is heterogeneous: it handles int/double/string objects as MultiType.



Visualisation tool – example 1

