

Status Of the Event Display

Dmitri Litvintsev

ITEP, Moscow

December 2, 1998

CDF offline meeting



○

Introduction

- **Three kinds of objects:**

- **Real Objects, Graphical Objects and Views**

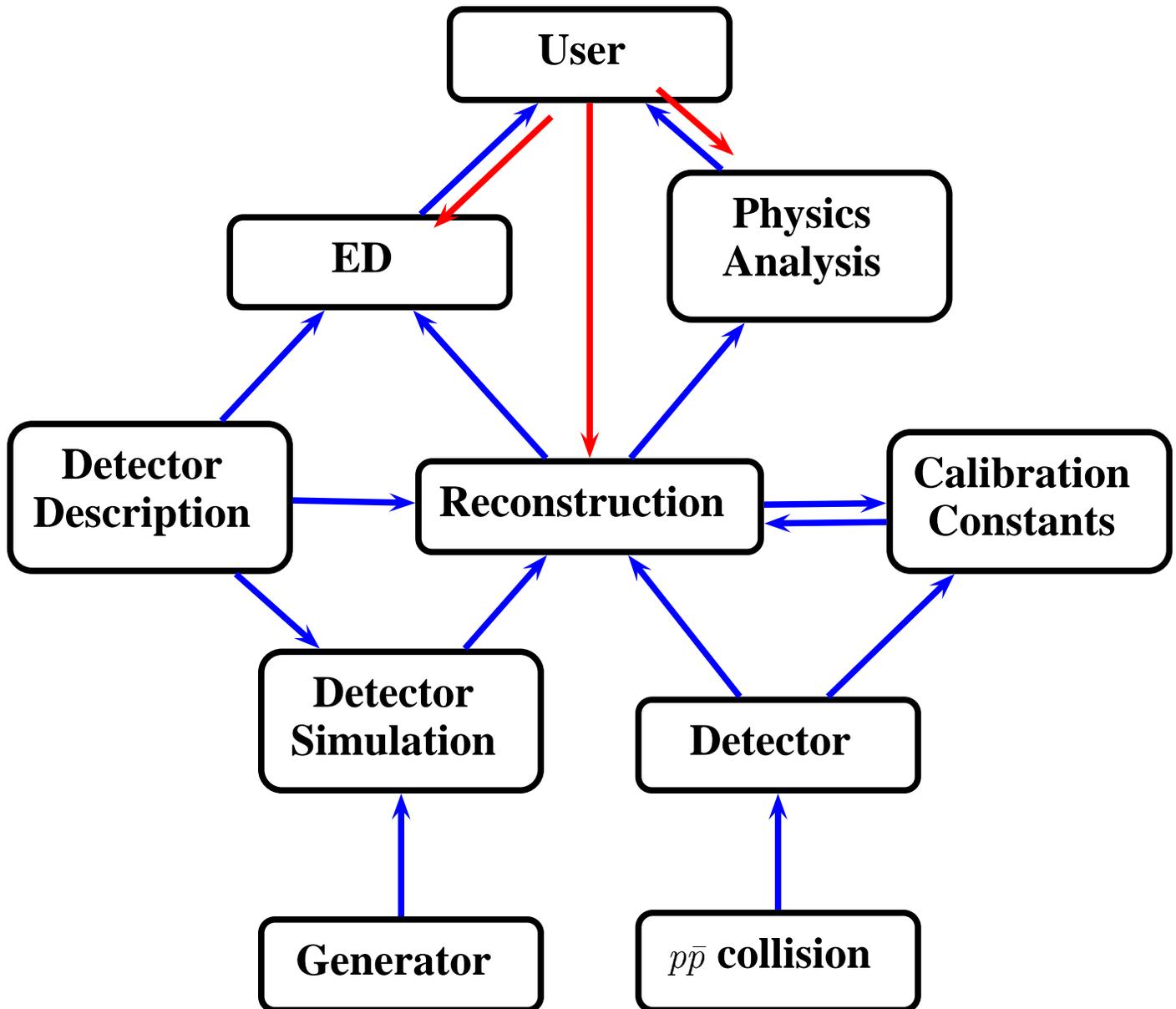
- **Real objects in the program are mapped to their graphical representations – Graphical Objects.**
 - **Views are different ways of visualization of sets of visual objects.**

- **Operations:**

- **Operations on Graphical Objects – change the visual properties of the graphics objects, the real objects are not changed. Operations can be performed on Graphical Objects or Views.**
 - **Operations on real objects can change the state of these objects.**



Introduction





Introduction

Functionality

- Access the data structures containing the data associated with the event and geometrical specification of the detector. (raw data, reconstructed data, physics data, simulated data etc.)
- The manipulation of the above data objects (mapping) suitable for translation into graphics primitives and subsequent presentation as displayed object.

Global Interactivity

Local Interactivity

- The presentation and manipulation of the graphics data in the display.
- Provision of comprehensive user interface to control the various options and the implementations of the above functions



○

OpenInventor

OI Status

OI is a Object Oriented toolkit for building 3-D applications.
OI is a library of objects and methods used to create interactive 3-D graphics applications.

- **History 1994 (?)**

- **Users**

- Vertual reality simulations
- Computer Animations
- Supported by GEANT4 project
- several experiments evaluating OpenInventor as a tool for visualization system building **ATLAS, CDF, CLEOIII, D0**
- Hepvis group (G.Alverson, A.Boehnlein, J. Boudreau, X. Fan, I. Gaponenko, J.Kallenbach, L.Taylor)

- **Availability**

- Proprietary product of Silicon Graphics Inc.
- Supported platforms SGI, Win NT, Linux promised
- **OI + OpenGL Unix (KAI)**
delevoper license: **\$6,950 + 16 % (MU)**
runtime license: **\$950 + \$152 (MU)**



○

OpenInventor

OI Status

- **OI only Unix (KAI)**
delevoper license: **\$3,950 + \$632 (MU)**
runtime license: **\$750 + \$120 (MU)**
- **native SGI Compiler**
delevoper license: **\$2,000**
runtime license: **free**
- **PC running Linux (KAI)**
delevoper license: **\$1,495 + 16% (MU)**
runtime license: **\$400 (MU)**
- **”side-wide” license is still a question**

Pricing information could be obsolete !



○

OpenInventor

OI Highlights

- **Library of objects the user can use and modify according to his needs**
- **Database primitives including shape, property, group and engine objects**
- **Interactive manipulators – handlebos and trackball**
- **material editor, directional light editor, examiner viewer**
- **Outputs VRML file, browseable over web**
- **Only 3-D representation of objects, OpenGL used as a low level graphics engine. 2-D image – projection of 3-D scene.**
- **rasterPostScript output**



ROOT

ROOT Status

ROOT is a set of over 300 C++ classes and an interactive C++ interpreter. It provides a OO framework for data analysis from event generation to graphical presentation.

• History

- Project started January 1995, original author Rene Brun (author and coauthor of extensively used PAW and GEANT packages)
- version 0.5 (demonstration prototype) in 11.95
- Latest version V2.00/13

• Users

- Started at NA49 (CERN)
- Adopted by Alice, Phobos...
- Several experiments evaluating ROOT as a base framework Minos, Star, Phenix, Brahm, HeraB

• Availability

- Shareware, i.e. priceless...
- Automated from Web side: <http://root.cern.ch>
- Support for all Unix platforms (native compilers + gcc on most) SGI, Linux, OSF, Solaris, Win 95/NT ...
- Documentation available



ROOT

ROOT Highlights (relevant to ED)

- C++ **CINT** interpretator
 - Automatic GUI and I/O code generation
 - Graphics:
 - **2-D** graphics (low-level and high level classes). All low level graphics primitives (lines, circles, boxes etc.) High level functions to draw non-equidistant points.
 - support for **vector** and **raster** Postscript
 - Canvas can be saved into several formats PS, EPS, GIF, into **ROOT** file for processing in a new session or into C++ macro file.
 - **3-D** graphics (**X3D** and **OpenGL**). X3D runs on Unix. Very fast capable of drawing complex 3-D pictures over the network.
X3D supports:
 - * **wireframe**
 - * **hidden line and hidden surface algorithms**
- OpenGL runs on Unix and Windows 95/NT systems. Implementation based on public product **MesaGL**
- **Editor** allows to add text and primitives to Canvas.



ROOT

ROOT Highlights (Cond) (relevant to ED)

- **1-D 2-D 3-D histogramming. Profile histograms and projections. User defined objects can be added to a list in a given histogram.**
- **set of container classes. TClonesArray provides support of arrays of objects typically created in the reconstruction and simulation**
- **Minimization based on Minuit**
- **Client/Server, Shared memory and Threads support. Crucial for on-line use.**
- **Interface to OS**
- **General Utilities (linalg, transforms, Random numbers)**



○

Requirements to ED

Functional Requirements

Views

- Views to be available
 - **x-y**
 - **r- ϕ**
 - **r-z**
 - **Fish-eye**
 - **Histogramming**
 - **Wireframe 3D**
 - **Solid 3D**
 - **2D slice of 3D**
 - **Hierarchy**
- Textual information
- Several views simultaneously



○

Requirements to ED

Functional Requirements

Objects

- **Following objects should be viewable (not necessarily all listed views available for a given object)**
- **Detector components**
- **Active detector parts**
- **Passive detector parts**
- **Event data**
 - **Hits**
 - **Tracks**
 - **Muon stubs**
 - **Calorimeter Clusters**
 - **Jets**
 - **particles**
 - **MC truth**
- **Errors (uncertainties) visualization**



○

Requirements to ED

Functional Requirements

Visual Operations

- **Operations:**
 - **Rotation**
 - **Zooming**
 - **Zooming in one direction**
 - **Scaling**
 - **Scaling in one direction**
- **change of viewpoints**
- **hiding/unhiding selected objects**
- **sub-view creation**
- **operations on created groups of object**
- **visual properties of objects definition (colors, line patterns, ...)**



○

Requirements to ED

Functional Requirements

Access to Data

- Access to the Raw data, simulated data and reconstructed data
- Browsing and searching events

Operations on Data

- Get/set object values
- Call selected real object member functions
- Start/stop/run/rerun job from the ED
- Redoing reconstruction of selected events (parts of events) after browsing
- Selection according to data source, physics channel, sub-detector, ... should be available
- Interface to the histogramming package
- Impose cuts from the ED and use them to select events



○

Requirements to ED

Functional Requirements

Setup

- It should be possible to read/save/change setup
- Standard (default) setup should be available
- It should be possible to read/save selected views
- Output in the following formats:
 - Vectorized PostScript (PS/EPS)
 - JPEG/GIF/TIFF/BMP ...
 - VRML
- It should be possible to print (selected regions of) views
- Slow operations should be iterative. (The most important parts of the view should be displayed first)
- Slow operations should be interruptable
- On-line and off-line help should be available
- Command line interface should be available. Should be possible to enter ED commands from the command prompt
- It should be possible to create macro-scripts
- It should be possible to record and replay any sequence of operations



○

Requirements to ED

Operational Requirements

- ED should be a module
- ED should be able to interact with running reconstruction program. (It should be possible to control running program and behave like a simple GUI and running program should be able to update ED automatically)
- It should be possible to reprocess an event after interacting with it via ED
- Graphics Editor should be available allowing for simplification of data presentation

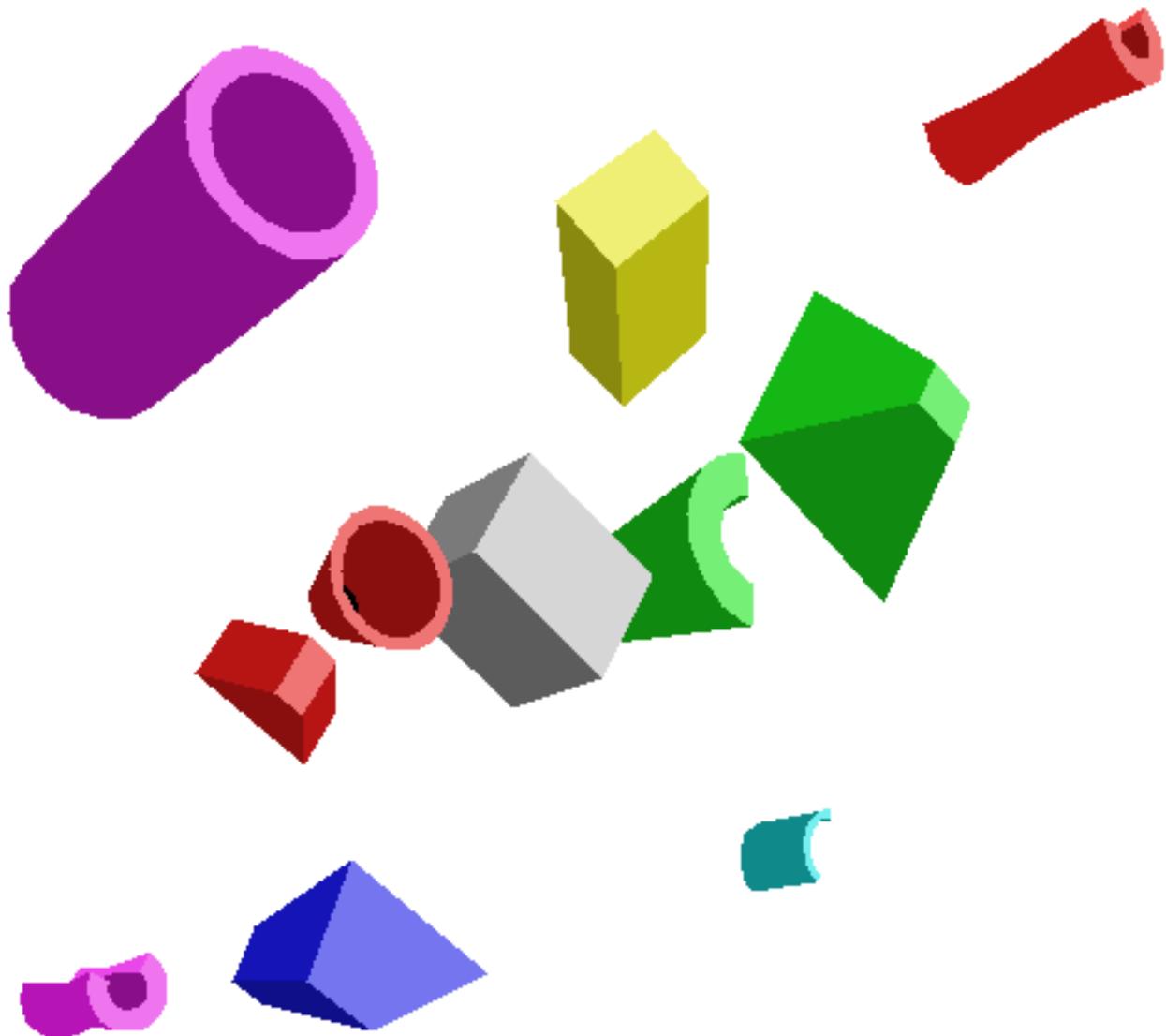
System Requirements

- ED should be able to run remotely over the network.
- ED should run on all used platforms and OS (currently SGI, Linux, OSF)
- ED should run with reasonable speed on any supported platform w/o special graphics hardware
- ED should be able to use special graphics hardware if exists on used machine.
- It should be possible to use ED in on-line environment



ROOT

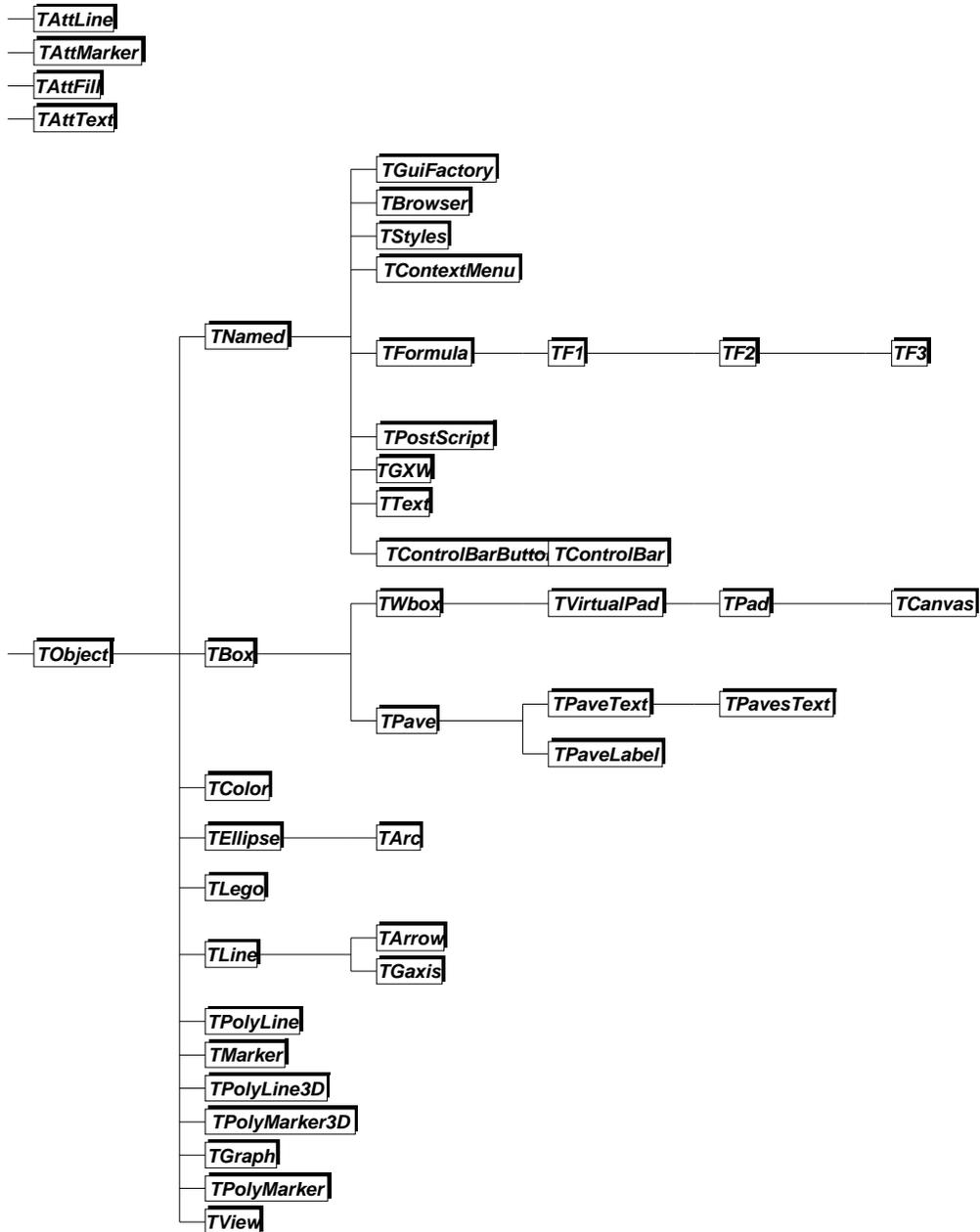
ROOT 3-D basic primitives





ROOT

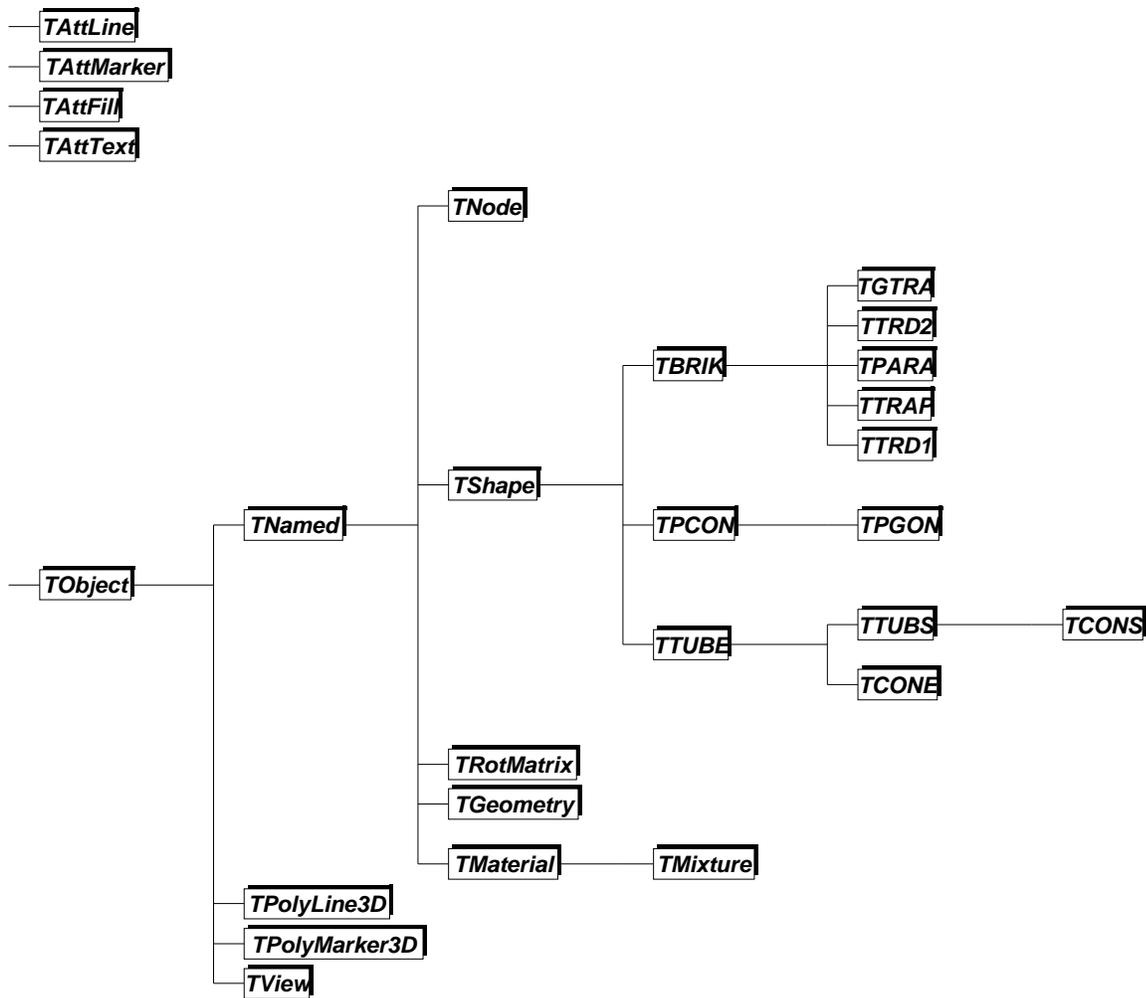
2-D Graphics Classes





ROOT

3-D Graphics Classes





○

ED Prototyping

Existing Prototypes

- **OpenInventor** based ED prototype for SVX by Joe Boudreau (standalone)
- **OpenGL** based ED prototype for COT by G.Gerard and Pasha Murat (Module integrated into Framework, accessible via talk to)
- **ROOT** based ED prototype for SVXII/ISL by Uni. of Karlsruhe: (Michael Feindt, Kurt Rinnert and Patrick Schemitz) (Module integrated into Framework, accessible via talk to)
- **ROOT** based ED prototype for COT by Pasha Murat (Module integrated into Framework, accessible via talk to)



○

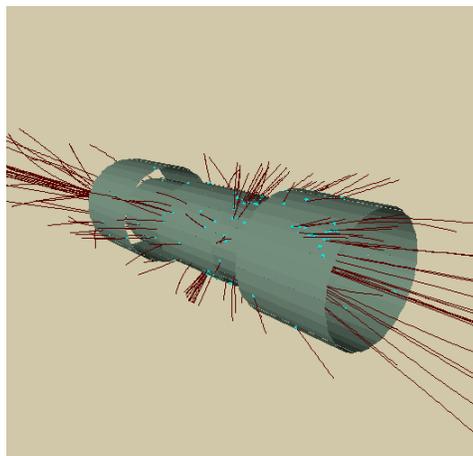
ED Prototyping



UNIVERSITY OF KARLSRUHE

Event Display for ISL/SVXII

- AC++ module, configurable via talk-to
- 3D model of ISL and SVXII
- Based on the ROOT framewor
- Zoomable 2D and 3D views using ROOT graphics engine, and OpenGL
- Runs on IRIX and RH Linux





○

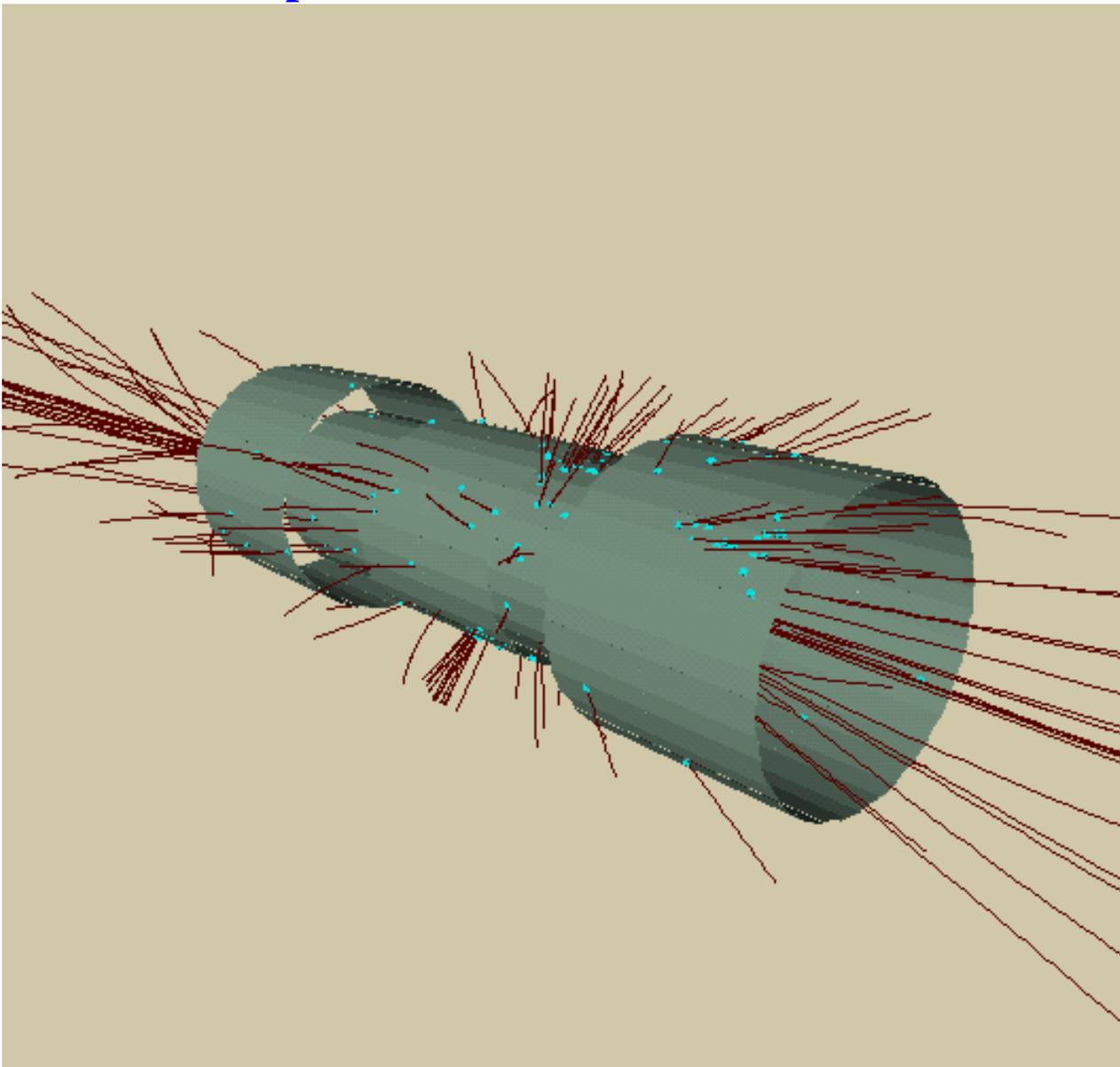
ED Prototyping

ROOT Based ED for ISL/SVXII



UNIVERSITY OF KARLSRUHE

OpenGL View of SVX Detector





○

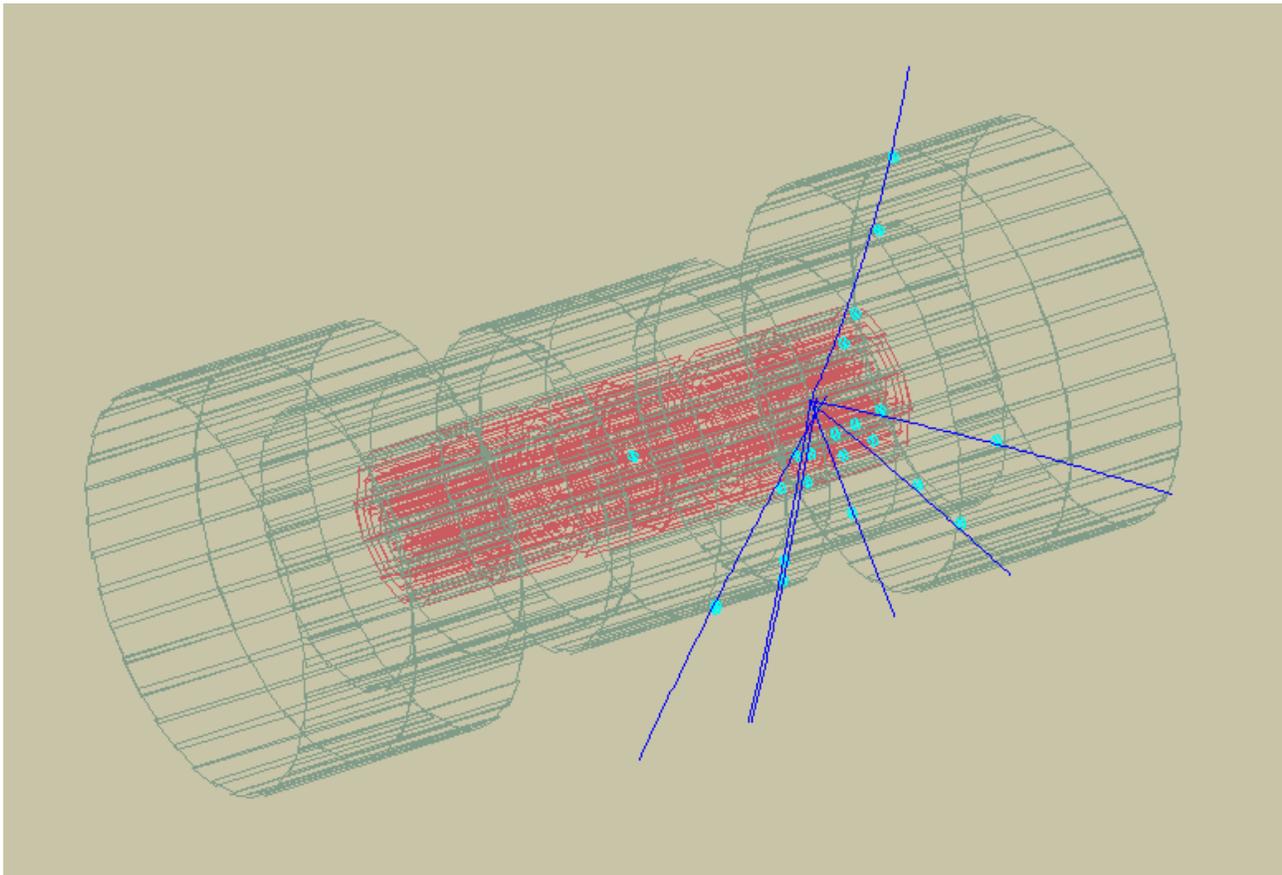
ED Prototyping

ROOT Based ED for ISL/SVXII



UNIVERSITY OF KARLSRUHE

3-D Wireframe View of SVX Detector





○

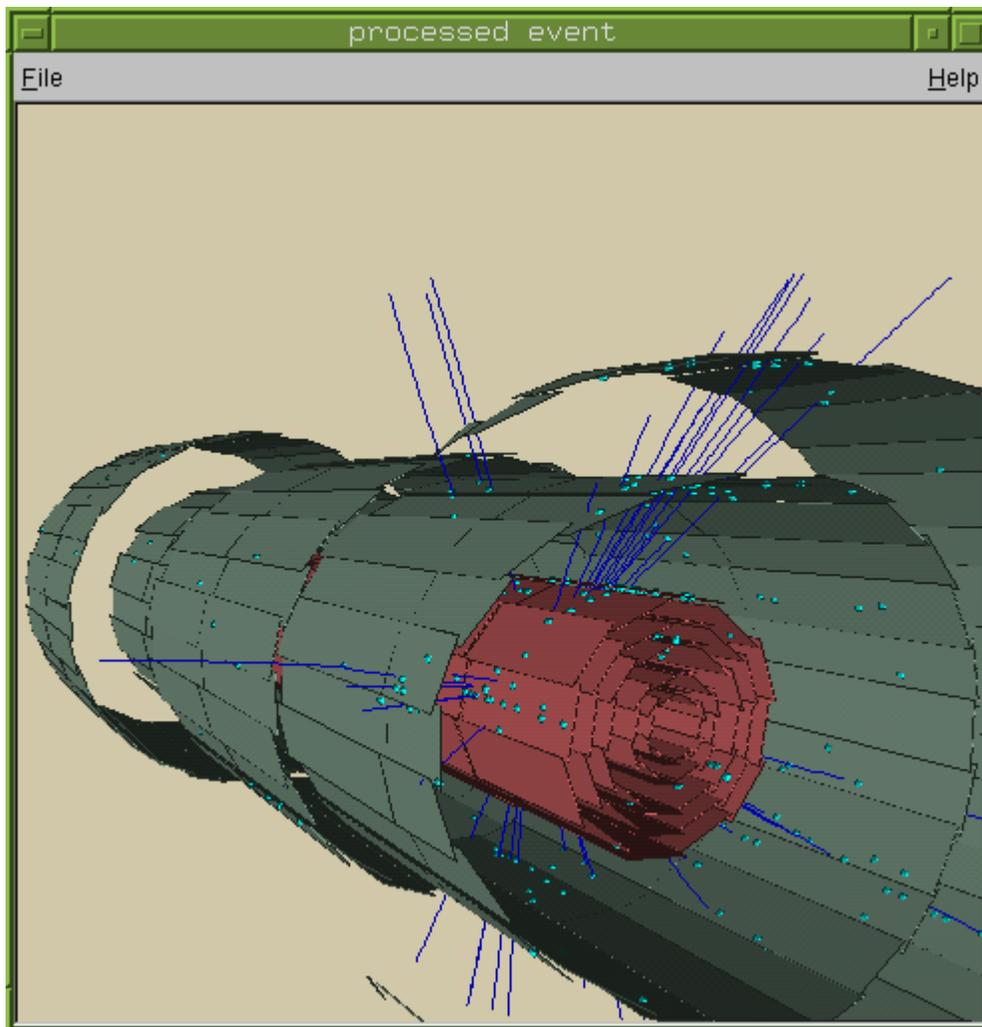
ED Prototyping

ROOT Based ED for ISL/SVXII



UNIVERSITY OF KARLSRUHE

zooming on 3-D views





○

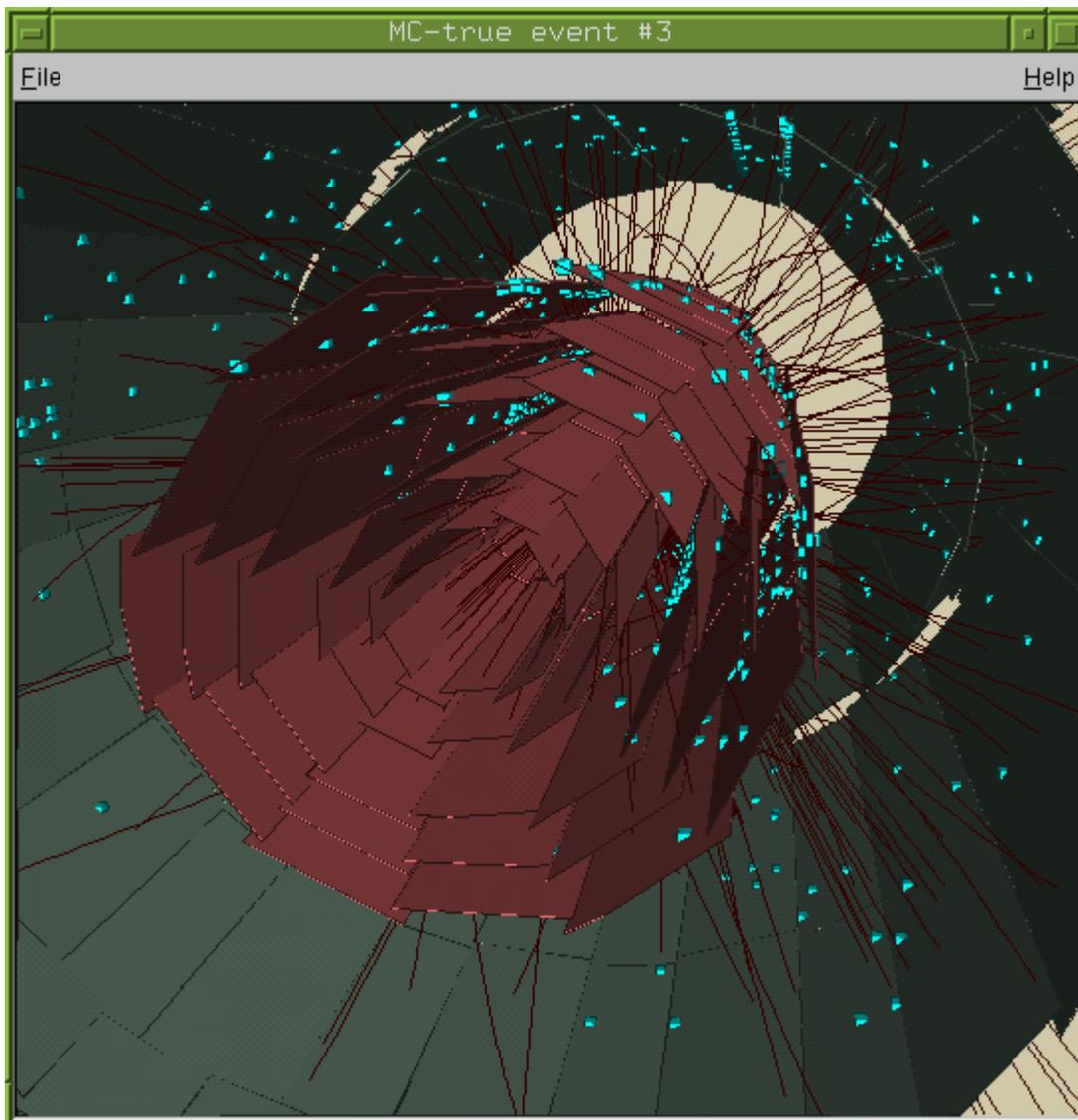
ED Prototyping

ROOT Based ED for ISL/SVXII



UNIVERSITY OF KARLSRUHE

zooming on 3-D views





○

ED Prototyping

ROOT Based ED for ISL/SVXII



UNIVERSITY OF KARLSRUHE

The screenshot displays the ROOT Event Display (ED) interface. The main window, titled "MC-true event", shows a complex event visualization with numerous red tracks radiating from a central point. A prominent green track is visible. The interface includes a menu bar with options: File, Edit, View, Options, Inspect, Classes, and Help. On the right side, there is a panel titled "KEventCanvas::event" containing a list of interactive options:

- ZoomIn
- ZoomOut**
- MoveRight
- MoveLeft
- MoveUp
- MoveDown
- SetZoomStep
- SetMoveStep
- ResetView
- SetNextEvent
- ✓ SetEditable
 - Divide
 - UseCurrentStyle
 - Range
 - SaveAs
 - SetBorderMode
 - SetBorderSize
 - SetGridx
 - SetGridy
 - SetLogx
 - SetLogy
 - SetLogz
 - SetName
 - SetTickx
 - SetTicky
 - x3d
- Delete
- DrawClass
- DrawClone
- Dump
- Inspect
- SetLineAttributes
- SetFillAttributes



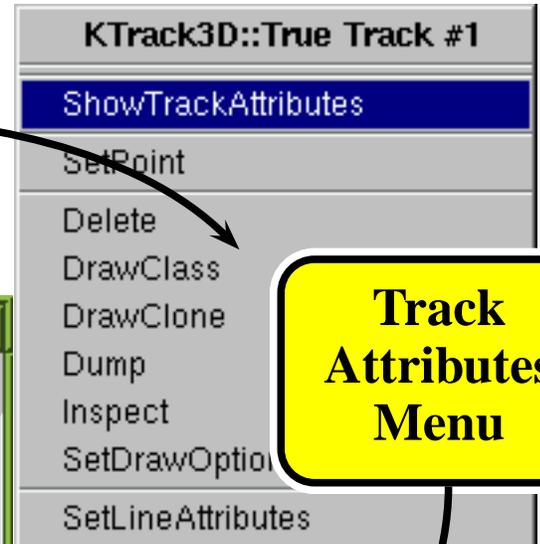
ED Prototyping

ROOT Based ED for ISL/SVXII

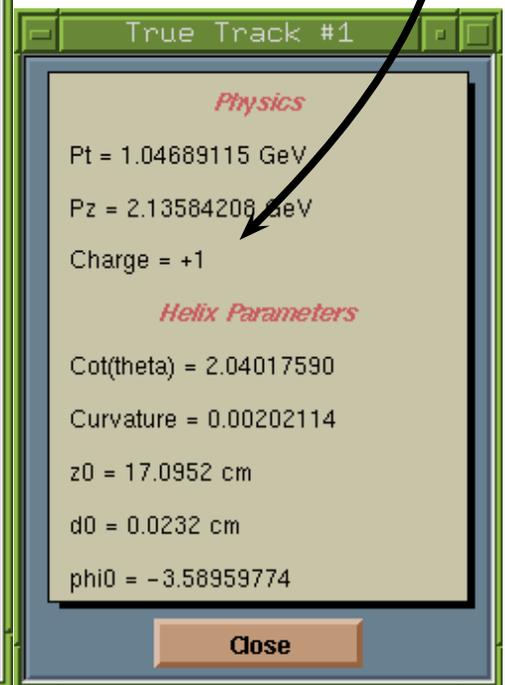
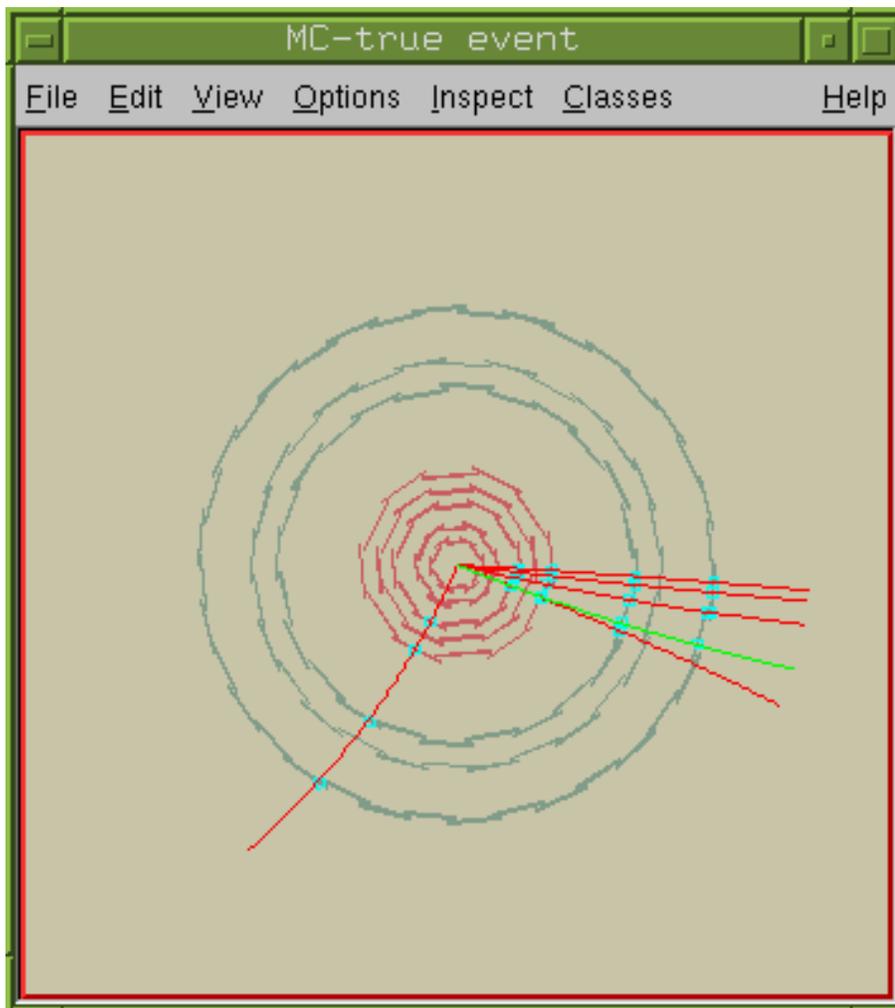


UNIVERSITY OF KARLSRUHE

Context Menu of a given Class



Track Attributes Menu



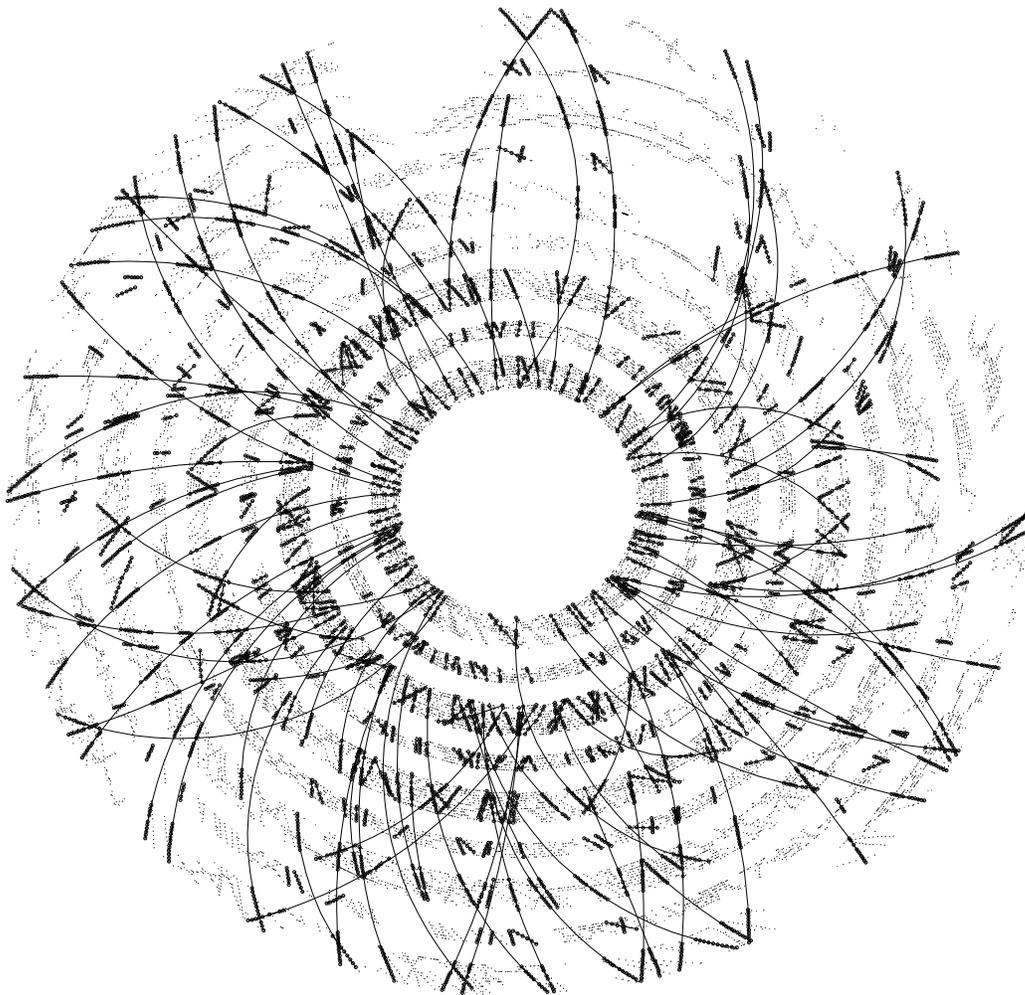


○

ED Prototyping

ROOT Based ED for COT

2-D View of COT Detector



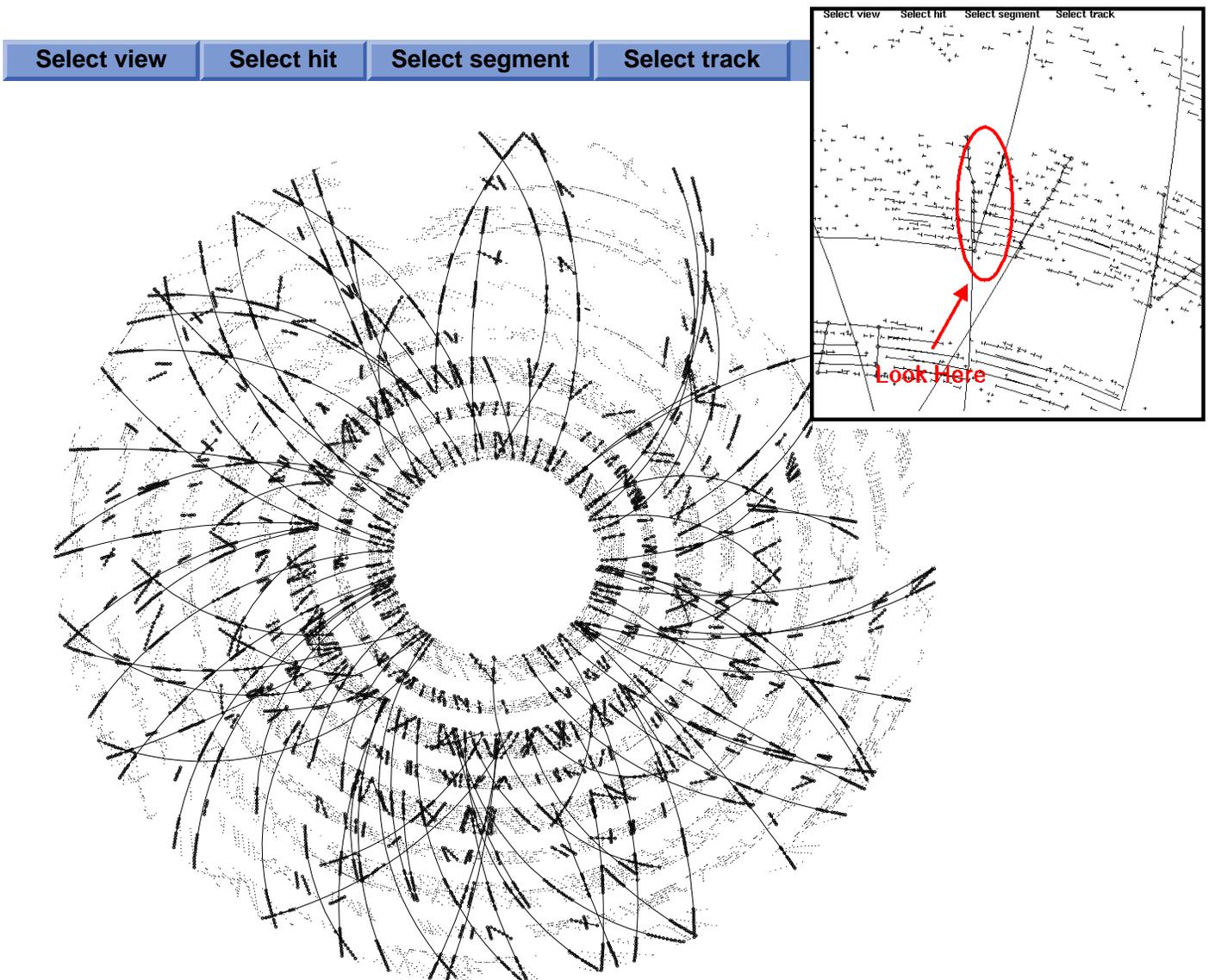


○

ED Prototyping

ROOT Based ED for COT (P.Murat)

Manipulations of 2-D View of COT





ED Prototyping

ROOT Based ED for COT (P.Murat)

Session Display

The image shows a ROOT-based event display (ED) for the COT detector. It consists of several windows:

- COT display (Left):** Shows a top-down view of the detector's concentric layers with tracks overlaid. A red label 'snapshot' is in the top left corner.
- COT display (Right):** Shows a perspective view of the detector layers and tracks.
- Command Prompt Window (Bottom):** A terminal window showing the execution of the 'xterm' command. It displays a table of track parameters and a list of segments. A yellow callout box labeled 'Command Prompt Window' points to this window.

Table 1: Track Parameters

Track	McNumber	CotTrack	SvxTrack	Vertex	chi**2(vz)
0x12f901a0	5	5	3	2	1
0x12f3e070	6	11	4	7	2
0x12f0c3d8	8	12	5	7	2

Table 2: Segment Parameters

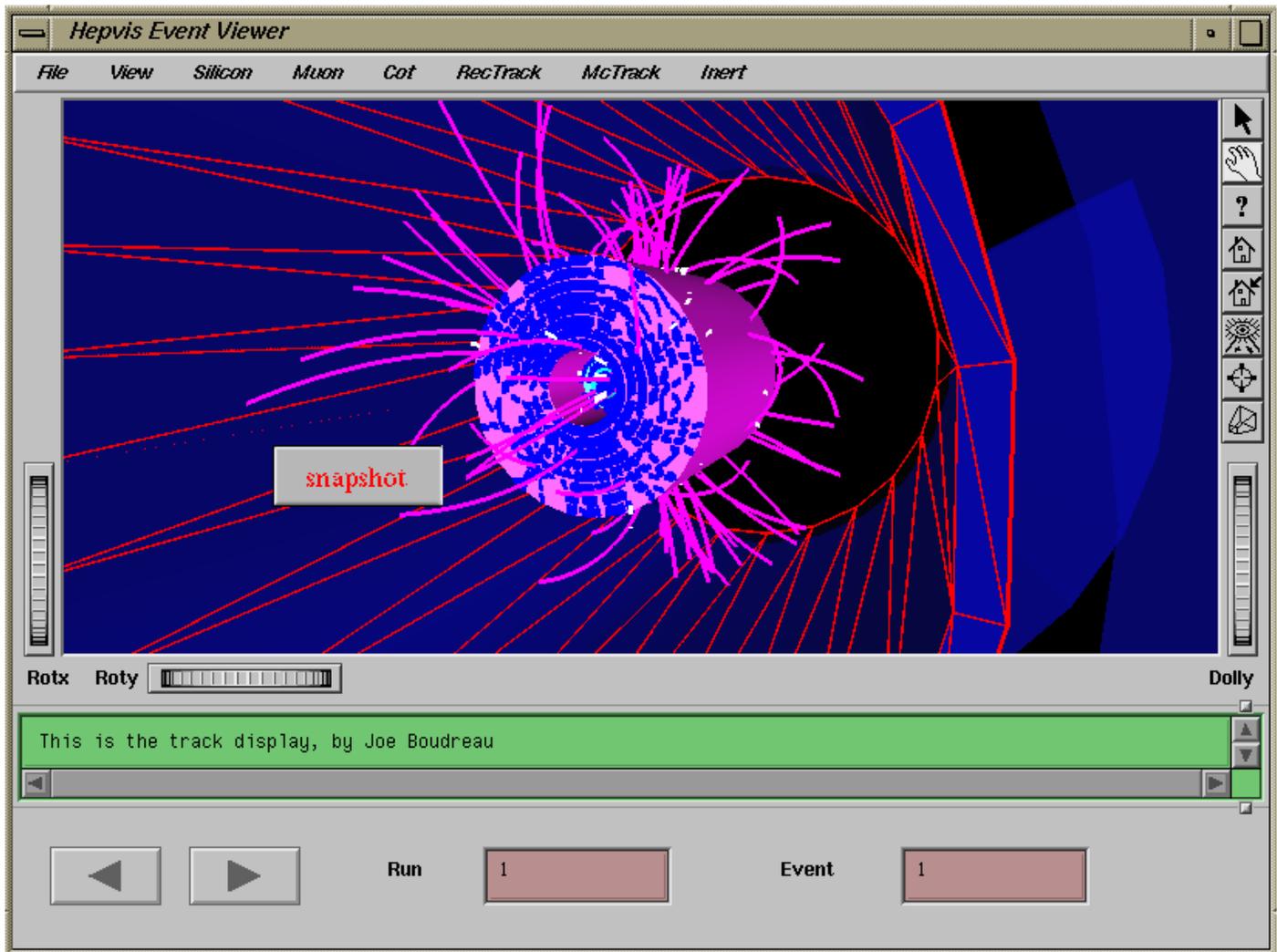
Segment	SL	NHits	NLeft	NRight	NCells	Chi2	Radius	Phi	DphiDr	DeltaDrift
0x12fd9100	0	6	4	2	2	2.60909	34.64	0.815188	-0.00298444	0.0534274
0x12ff0898	1	4	1	3	2	1.42071	46.3854	0.842225	0.000515009	-4.3689e-05
0x12f95e48	2	7	4	3	2	4.68799	58.2375	0.747635	-0.00302266	-0.0328022
0x12fd28c0	3	4	3	1	2	2.94866	69.88	0.668431	0.00616743	0.0426563
0x12f5edc0	4	9	7	2	2	2.03947	81.6539	0.67917	-0.00285088	-0.0131215
0x12f901a0	5	5	3	2	1	0.836172	93.2484	0.680702	-0.00269548	0.00666191
0x12f3e070	6	11	4	7	2	1.89634	104.916	0.610913	-0.00295794	-0.019994
0x12f0c3d8	8	12	5	7	2	0.701479	128.101	0.542223	-0.00299667	-0.0167264



○

ED Prototyping

OI Based ED for SVX J.Boedrau Session Display

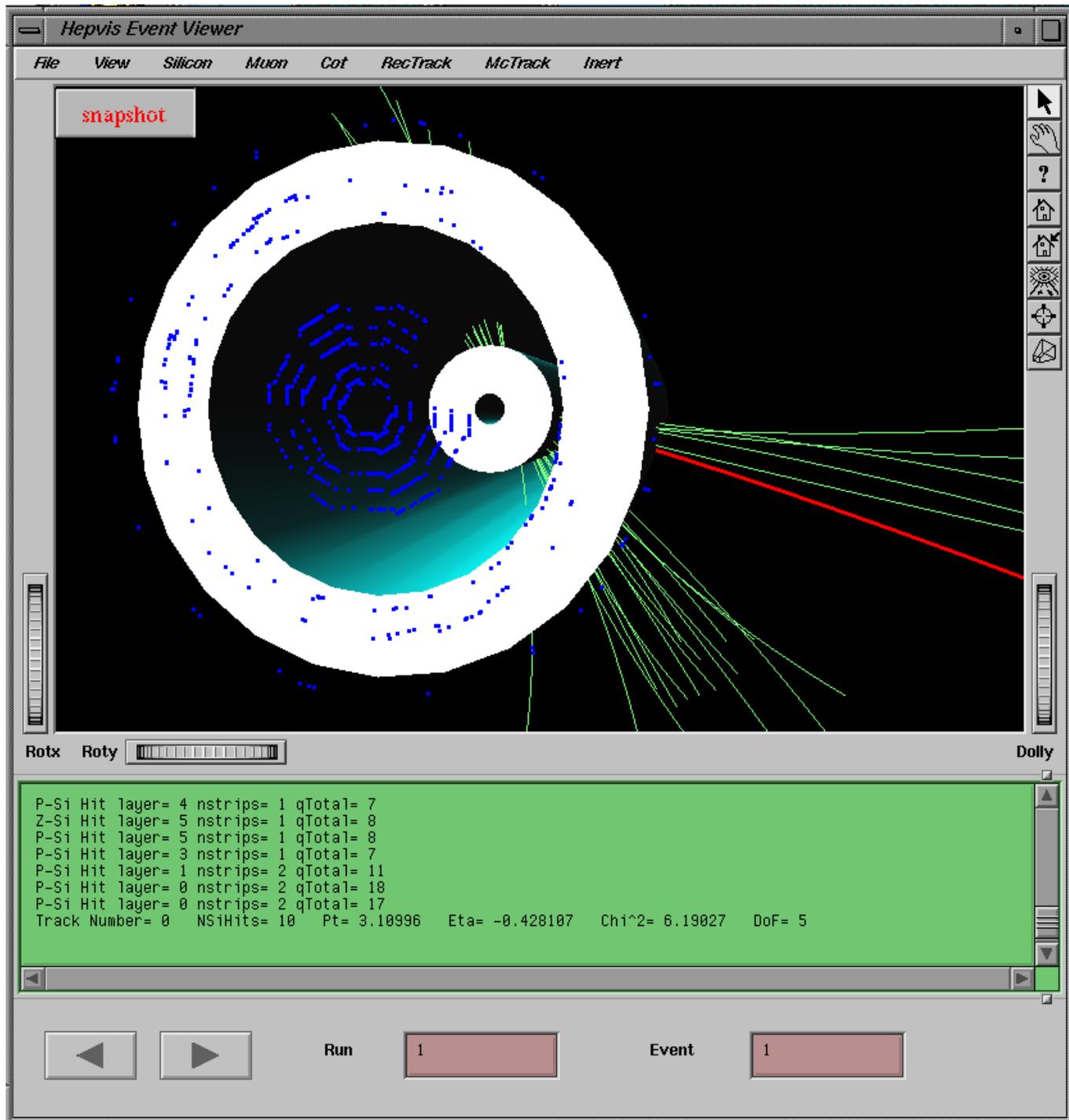




ED Prototyping

OI Based ED for SVX

J.Boedrau



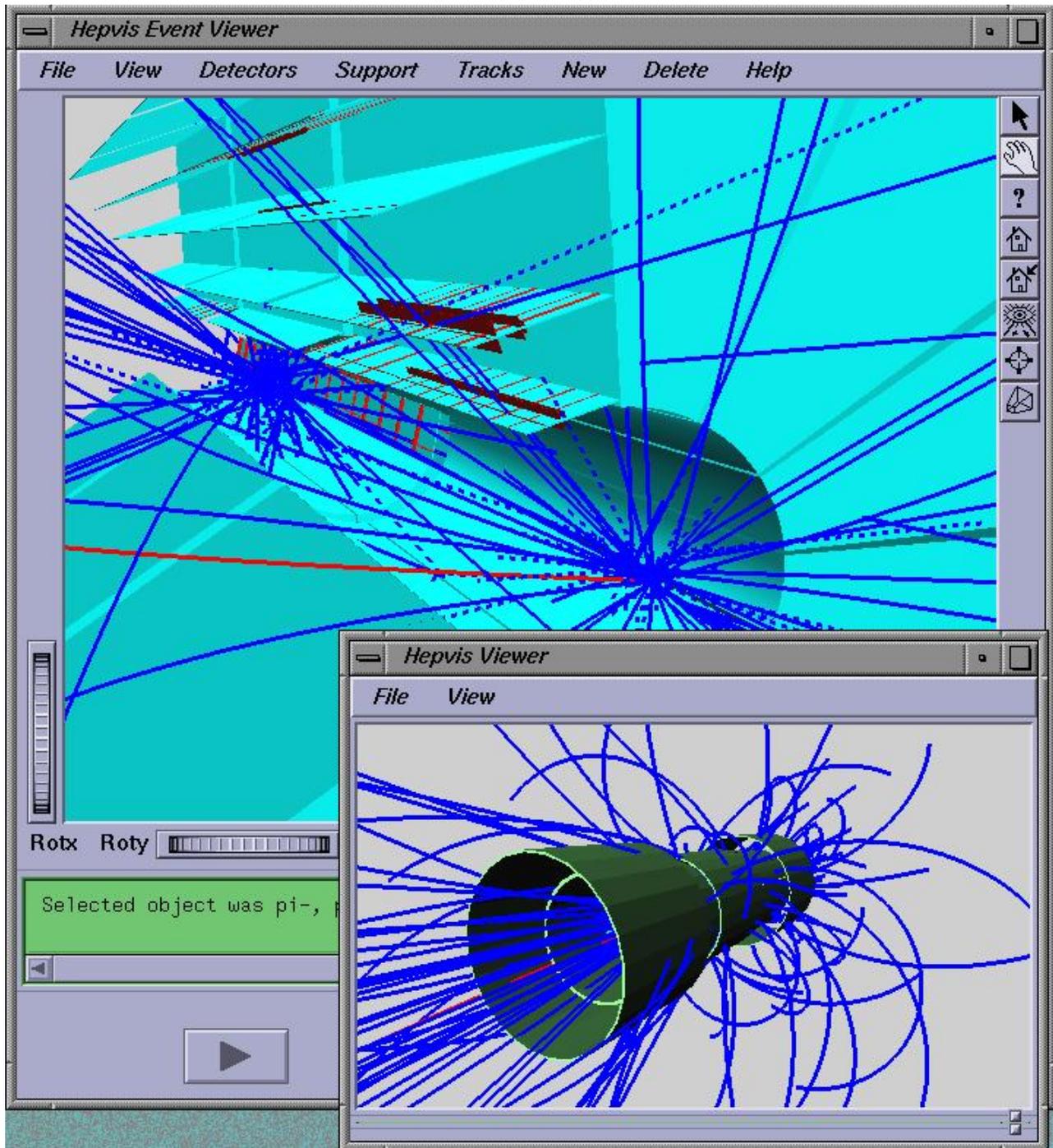


○

ED Prototyping

OI Based ED for SVX

J.Boedrau



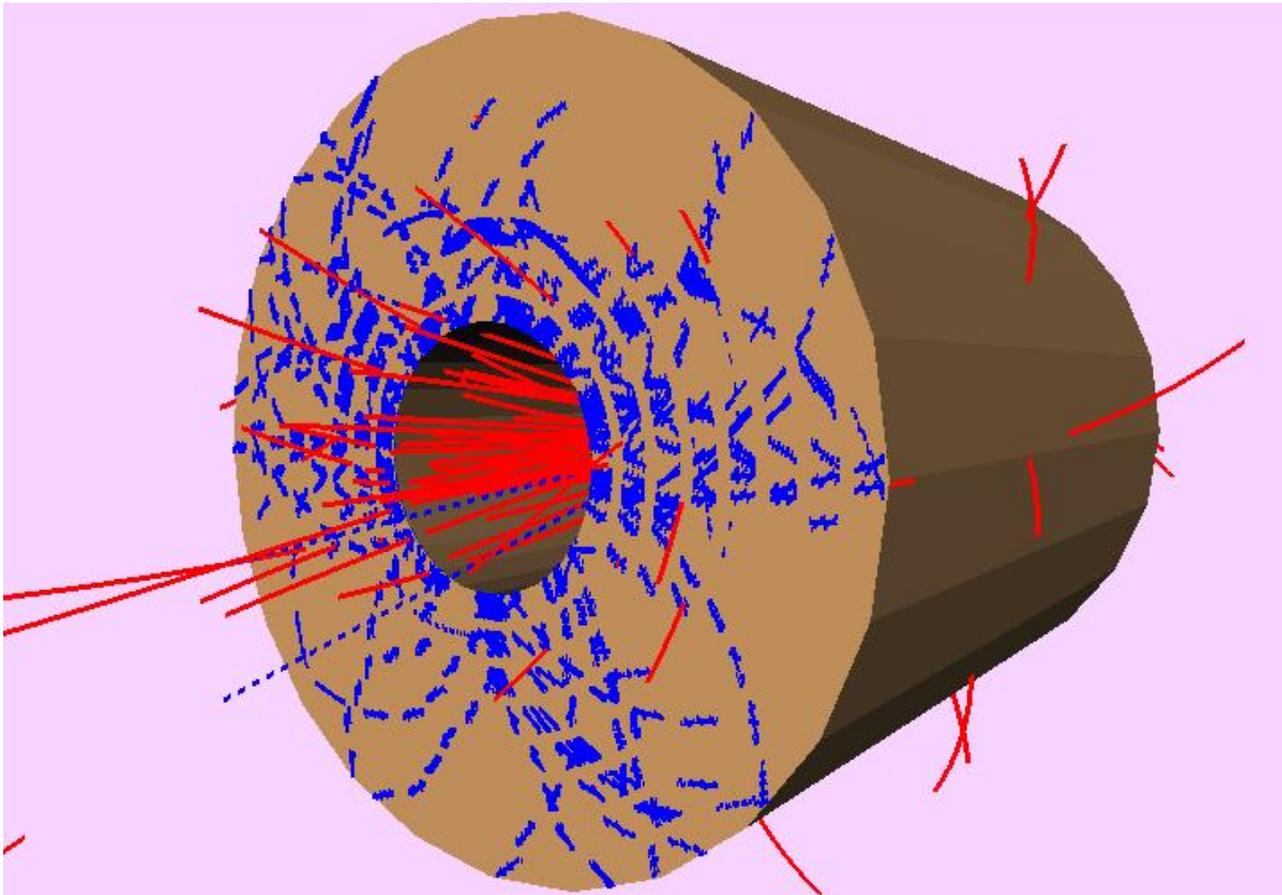


○

ED Prototyping

OI Based ED for SVX

J.Boedrau



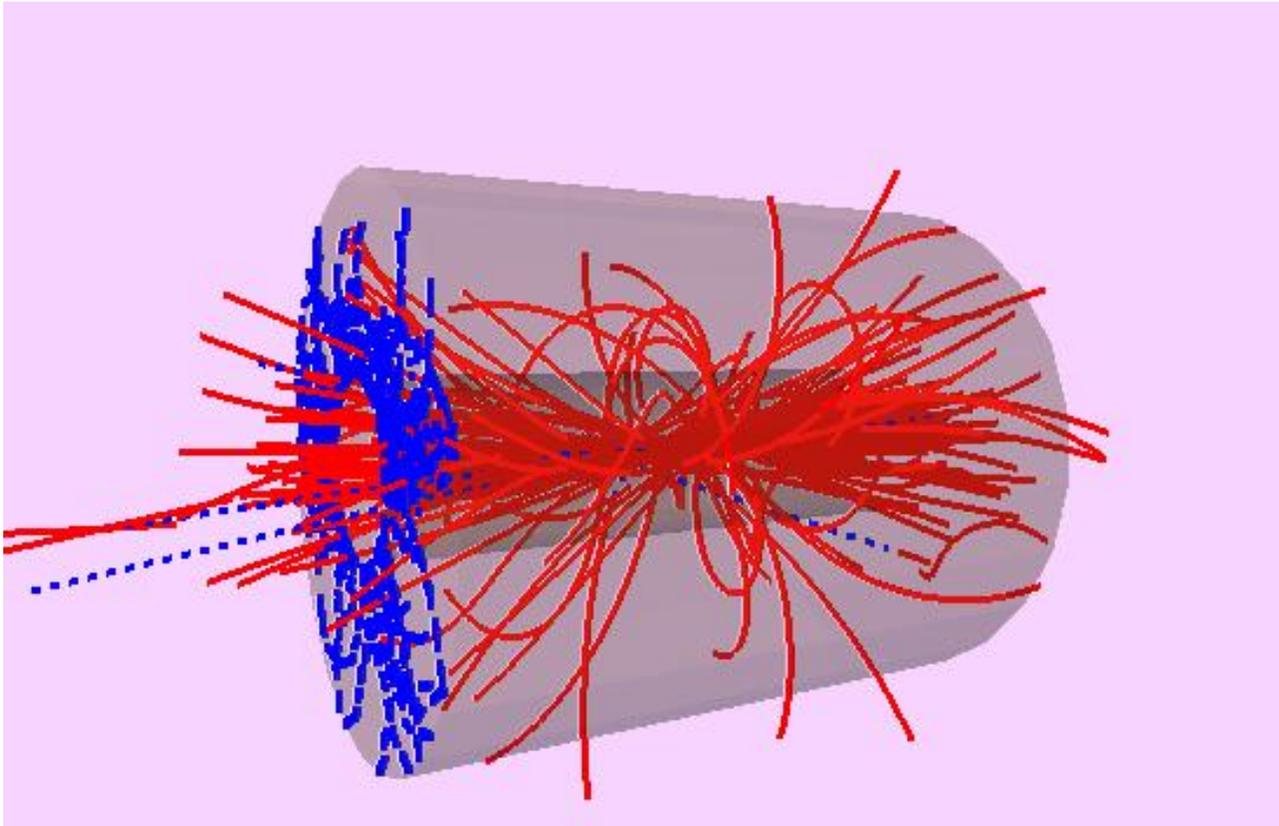


○

ED Prototyping

OI Based ED for SVX

J.Boedrau





○

ED Prototyping

OI Based ED for SVX

J.Boedrau

