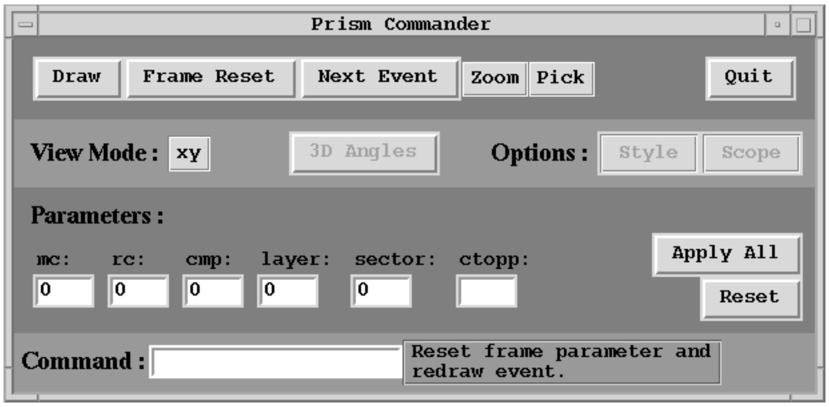
prism - the HERA-B Event Display and its Tcl/Tk User Interface

Version 4.9

Rainer Mankel
Humboldt Universität zu Berlin

May 30, 1997



Contents

Chapter 1

Introduction

1.1 Function

prism is a 2D event display package which started development in 1993. *prism* is designed to visualize the data entities of the HERA-B reconstruction environment *Arte* [1]. This means that objects can be displayed which are defined in the shape of *Arte* tables. The package is intended to help the user in

1. understanding the physics of HERA-B events
2. understanding the geometry of the HERA-B detector
3. controlling the performance of reconstruction tools
4. assist hardware & software development

The scope of displayable objects covers the detector itself, the signals measured in an event and the physical particles reconstructed from this information. For Monte Carlo generated events [2], also the Monte Carlo particle information is available. Visualization is mainly achieved by drawing, but in addition tools are provided which directly print information from the various *Arte* tables, which is very useful if you just want to find out what happens inside *Arte* .

1.2 Hardware and software environment

The *prism* module is embedded into the *Arte* program frame. *prism* uses the high level graphics interface HIGZ [3] from the CERN library as an interface to the basic graphic packages, usually GKS and X11. The user interface is handled with the KUIP package [4] from the CERN library. This type of interface which is well known from PAW provides builtin command abbreviation, help facility and macro interpretation, which makes the package quite user friendly. *prism* is

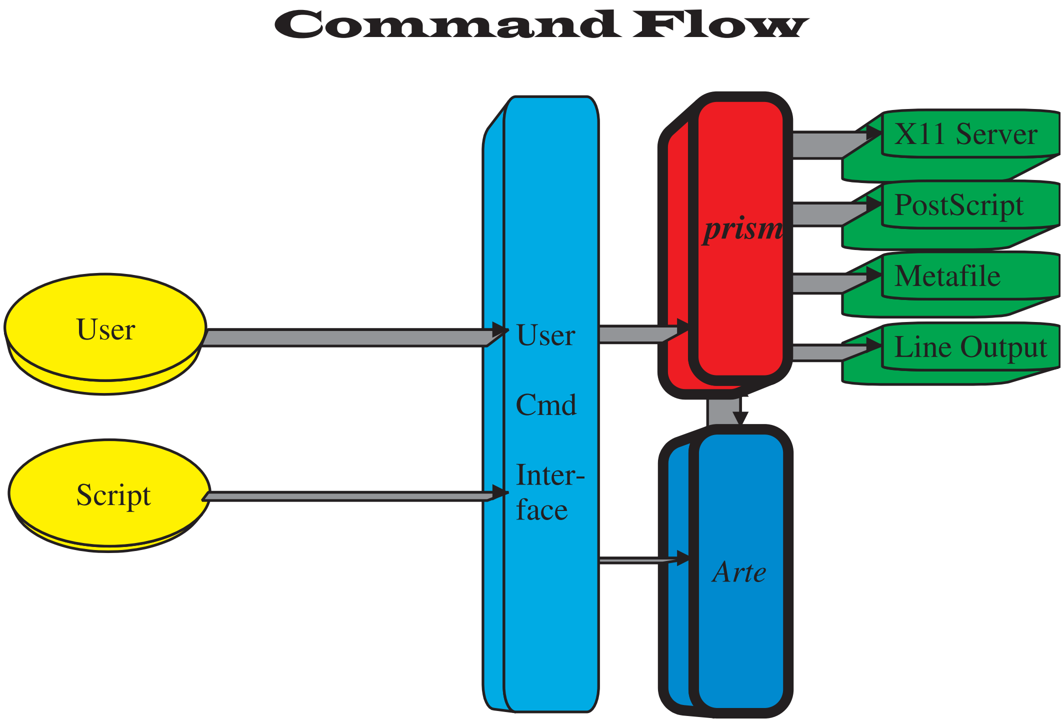


Figure 1.1: Event display command flow

activated through commands issued directly from the *Arte* command prompt – *prism* and *Arte* commands can be mixed in arbitrary sequence.

The way *prism* is embedded into the *Arte* framework is displayed in fig. 1.1.

Though *prism* can be completely controlled by commands, its most efficient use is through the *prism Commander*, which is a graphical user interface based on *Tcl/Tk*. See the dedicated chapter below for details on how to use it. Since the *prism Commander* runs as a separate process which communicates with the *Arte* module, the user can choose at run-time between the command mode and the graphical interface mode.

The *prism Commander* uses Tcl/Tk with at least Tk version 4.0. The Tcl script is executed under the *wish* shell which is assumed reside in the directory `/usr/local/bin`.

prism assumes as display station at least a grayscale monochrome X display. If a non-grayscale screen is used, a colour mapping to black and white is performed which however loses much of the differentiation between different object types. For full functionality, a colour X display is strongly recommended.

The present code has been run on Silicon Graphics 4D/Challenge, Hewlett Packard 9000/735, IBM RS/6000 and IBM SP/2 platforms without necessitating machine specific modifications. Implementations under Linux and Alpha/OSF are under trial as this document is written.

Chapter 2

The *prism* Command Mode

2.1 Initialization

The *prism* event display may (but need not) be initialized by the command

```
prism [metafile] [metatype] [wktype]
```

where `[metafile]` is the name of the file where PostScript pictures are to be stored. The name defaults to `events.ps`. The other two parameters can usually be omitted[1]. On initialization, the display window opens and the *prism* logo appears. The PostScript output file is created.

If the explicit initialization is omitted, *prism* initializes automatically when the first display command is issued.

2.2 Getting started

2.2.1 The generic display command

Most of *prism* 's commands belong to the `display` command menu. The command

```
display/draw
```

(re)draws the detector or event with the current display attributes. If no Monte Carlo tracks or other event objects are activated, the result is a detector view as in fig. 2.1. Note that every command can be abbreviated as far as uniqueness is preserved, thus

```
d/dra
```

or even

[1] See the HIGZ manual [3] for their explanation

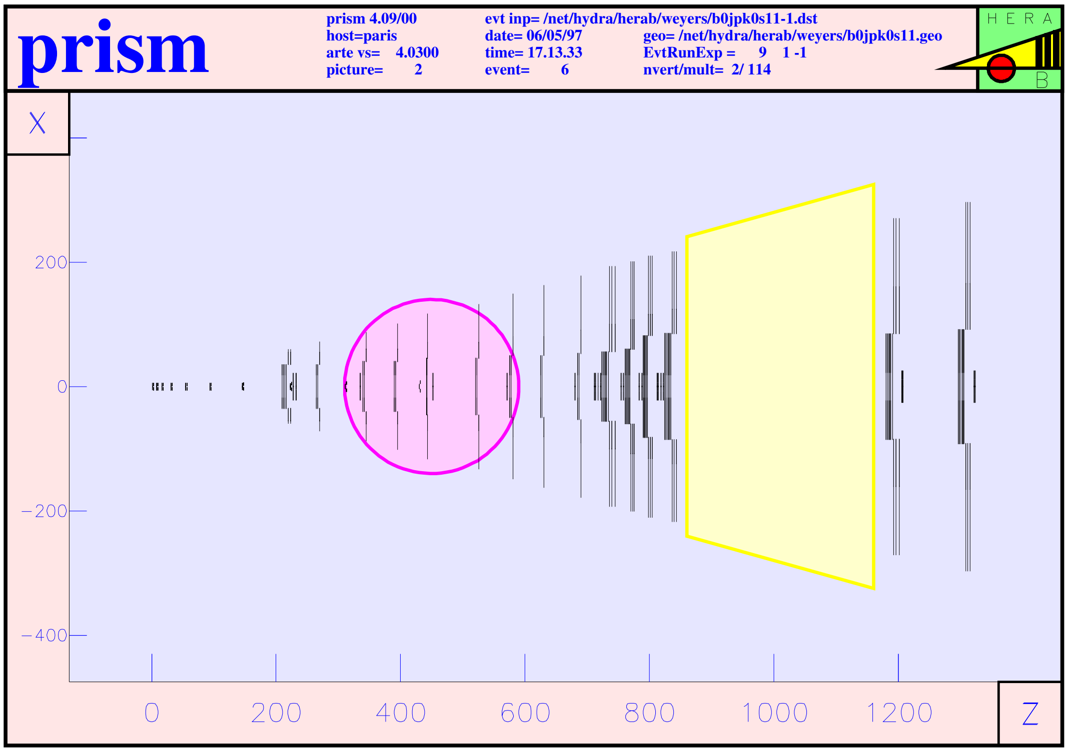


Figure 2.1: Event display showing detector in xz view

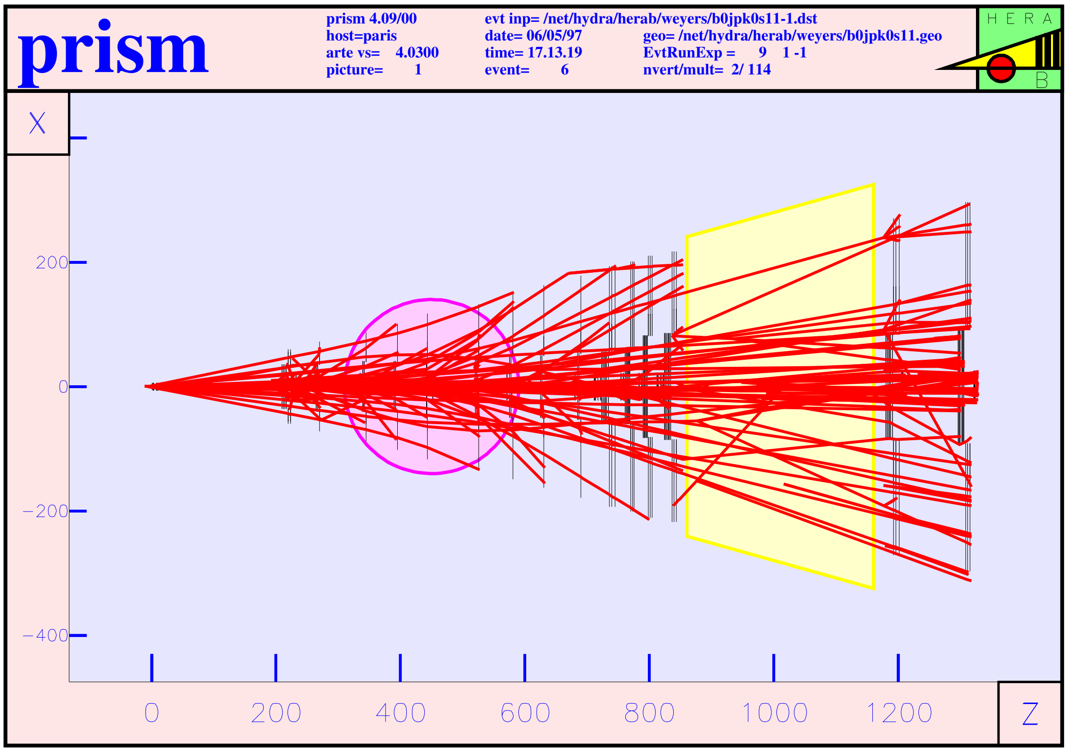


Figure 2.2: Event display showing detector with Monte Carlo tracks in xz view

```
draw
```

will have the same effect (this may change in future if conflicting command names should be introduced elsewhere in the HERA-B software chain). Some commands which change the appearance of the display issue an automatic redraw of the picture.

2.2.2 Switching the display of Monte Carlo particles

If the dataset you are using contains simulated events, the Monte Carlo tracks can be activated or deactivated by setting the corresponding style option. With the command

```
display/style mc=y
```

(the parameter may be **y**/**n** for yes/no or **t**/**f** for true/false), we arrive at the display in fig. 2.2 which shows Monte Carlo tracks as red lines.[2]

[2]Since *prism* is at present mostly applied to simulated data, the Monte Carlo track option is on by default.

Field	Content
evt inp	name of event input file
geo	name of geometry file
prism	*prism* version number
arte vs	*Arte* version number
host	node name of workstation
date	date at which picture is drawn
time	time at which picture is drawn
EvtRunExp	event identification (EventRunExperiment)
event	current number of event in session
picture	sequence number of picture on metafiles
nvert	number of primary vertices (=number of superimposed events) (MC)
mult	number of primary particles (MC)

Table 2.1: Explanation of header fields

2.2.3 The display header

The header shows information useful for interpreting and archiving event displays (see e.g. fig. 2.2). The header contains the current *prism* and *Arte* version numbers, the consecutive number of the picture in this *prism* session, the event input and geometry file names, the current time and date and the event identification (EventRunExperiment). The numbers labelled **nvert/mult** are the number of primary vertices[3] and the primary multiplicity, i.e. the number of particles emerging directly from these primary vertices, in case of simulated data.

The header can be switched off via

```
display/style header=n
```

The meaning of the header fields is summarized in table 2.1. The header sector is likely to change according to varying demands.

2.2.4 Displaying the B decay

It is common for fullscale HERA-B events to have so many tracks that it is difficult to get a clear view of the detector below. The following commands can be used to restrict the display to the particles from the B^0 decay:

```
display/param ctopp=B.
display/style chain=y type=y
```

where we have also activated the *type* option to display a particle type label next to the end of a track. The result is shown in fig. 2.3.

[3] i.e. those having no mother particle

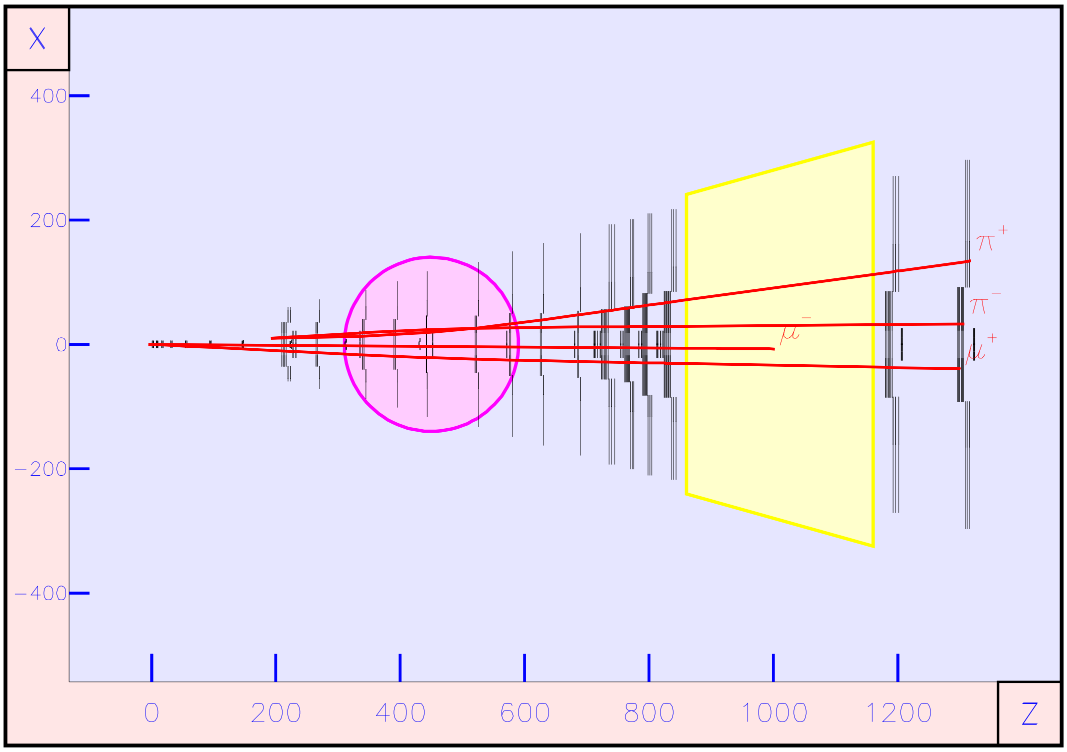


Figure 2.3: Event display showing the tracks from the golden B decay

2.2.5 Zooming in and out

The current display borders can be modified with the **frame** command. If no numbers are given

```
display/frame
```

the user is prompted with a rubberband to specify the desired borders directly on the display window. If the display window is not activated on your display or covered by another window, activate it into the foreground by pointing the mouse to its top bar and clicking the left mouse button.[4] (If you were to click into the display window itself this would be taken as the first frame definition point.)

You can alternatively enter the frame borders explicitly, e.g.

```
display/frame 200. 700. -200. 200. -a
```

if given in view 1 limits the display to the magnet area, since the first two numbers specify the horizontal range (z) in cm. Note that the second pair of numbers (here representing x) is only regarded literally since option `-a` is given, otherwise the vertical coordinates would be adjusted such that the natural aspect ratio is conserved.

The command

```
display/frame -r
```

resets the borders to default, which is calculated automatically from the current scope of detector objects to be displayed. It is also possible to zoom in or out by a fixed ratio $2\times$ (option `-i`) of 1/2 (option `-o`).

2.3 Viewing modes

The observer's direction of view can be set with the view command. The available views are

0 = front view (xy)
1 = top view (xz,default)
2 = side view (yz)
3 = arbitrary phi/theta view direction (see "angle" command)
4 = isometric
5 = dimetric
6 = trimetric
7 = along +5deg stereo wires (uz)
8 = along -5deg stereo wires (vz)

[4] Assuming the `mwm` window manager

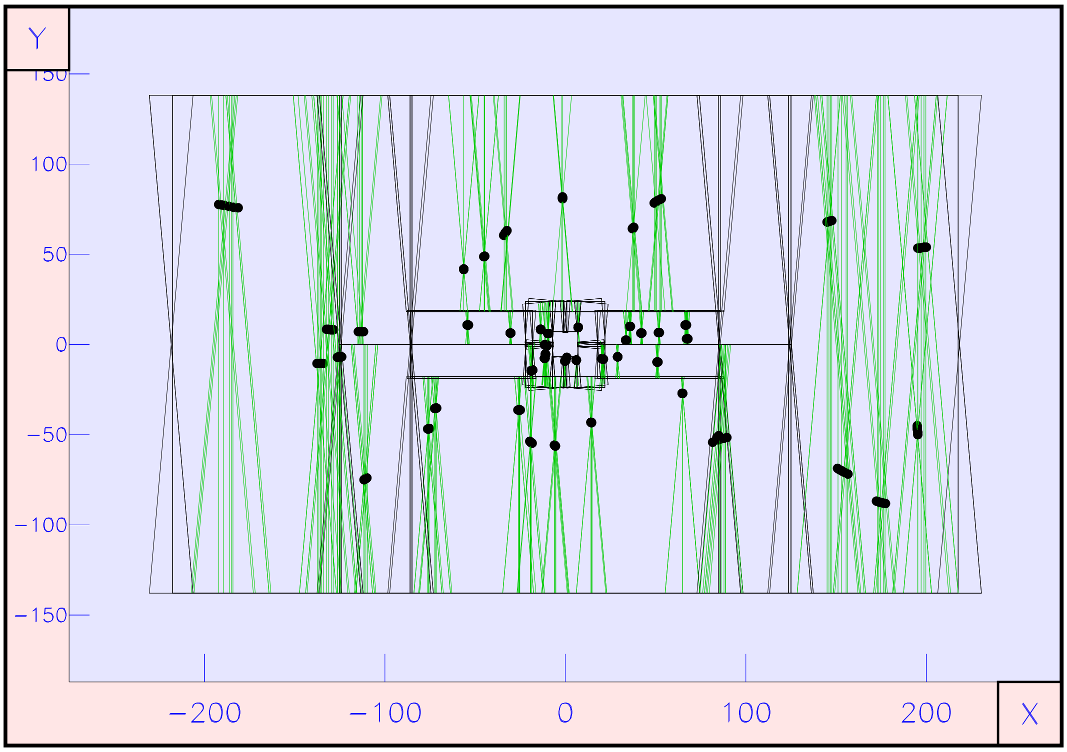


Figure 2.4: Example of event display with view 0. Also the stereo planes are shown in this picture. Both **wires** and **mimps** are active.

9 = "cone": special non-linear scale view (see below)
10 = "log-cone": "

An example of the xy view is shown in fig. 2.4. This display was generated with the commands

```
display/view 0
display/scope PC04 MS13
display/style st=y wire=y mimp=y mc=n
display/frame -r
display/draw
```

This display also shows the drawing of Monte Carlo **im**pact **p**oints (MIMPs) – i.e. those points where the Geant simulation has traced intersections between charged particle trajectories and sensitive detector volumes – as filled circles in black, which has been activated with the **mimp** option of the **display/style** command. Note how the **display/frame -r** directive is used to automatically adjust the frame after the scope has been changed. The command sequence

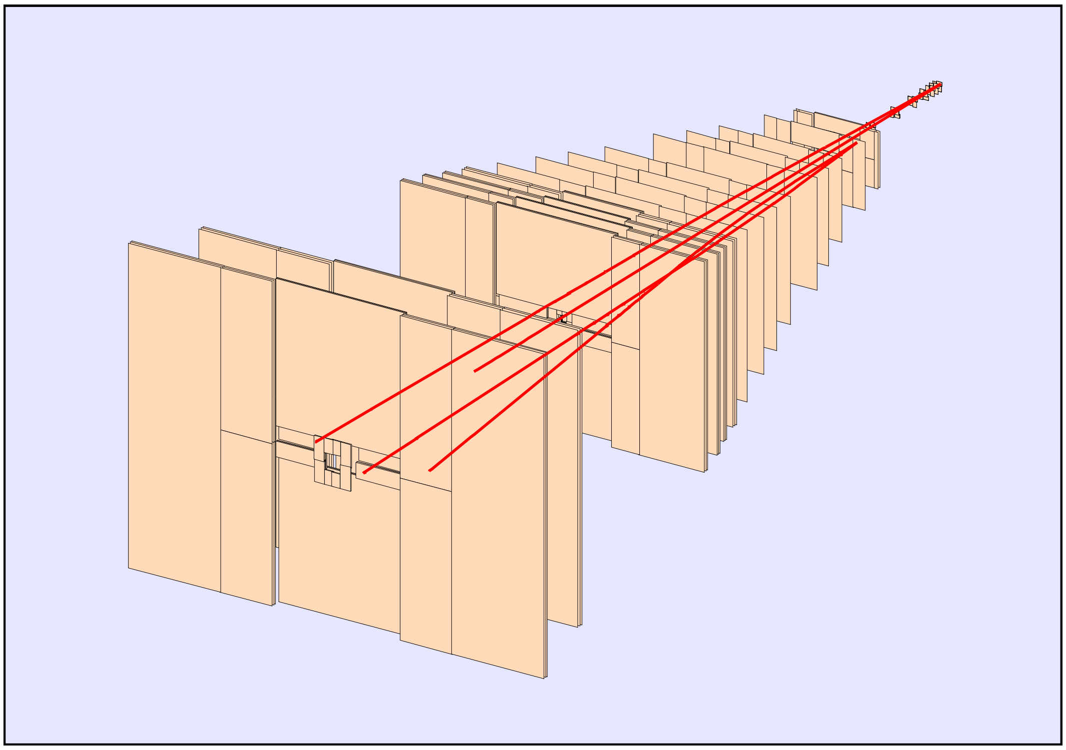


Figure 2.5: Example of event display with trimetric view 6

```
display/view 6
display/style opaque=y
display/draw
```

gives a pseudo-3D view of the detector, since style option **opaque** fills the area of each plane to be drawn, resulting in a simple hidden lines removal (fig. 2.5).

The viewing angles (in degrees) for view 3 are set with the **display/angles** command. The command sequence

```
display/angles phi=0 theta=25
display/view 3
display/detector -mxh
```

produces the display in fig. 2.6.

The views 9 and 10 are experimental nonlinear representations of the events. View 9 transforms the cone imposed by the Lorentz boost to a rectangle with the result that tracks coming from the target become approximately parallel. View 10 attempts, in addition, to boost the vertical scale at small radii in such a way that the track density becomes roughly flat, resulting in a blowup of the whole inner tracking system. This view is intended to make outer, inner and vertex tracker properly visible at the same time and allow to view a track "in full",

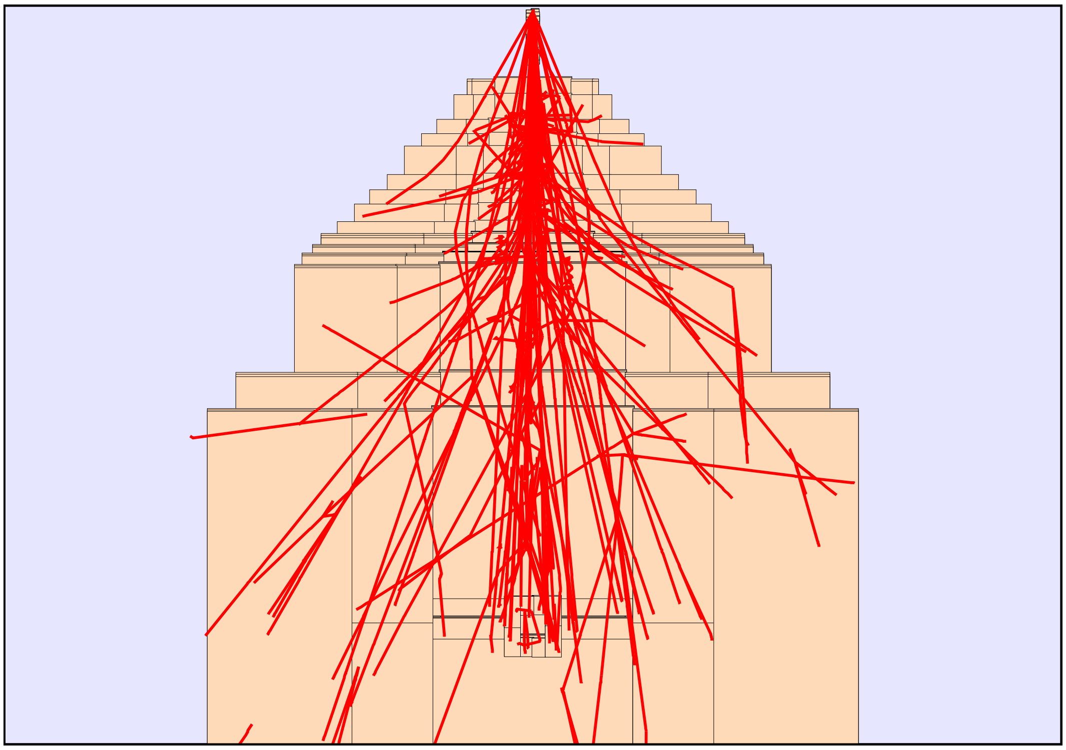


Figure 2.6: Example of event display with view 3

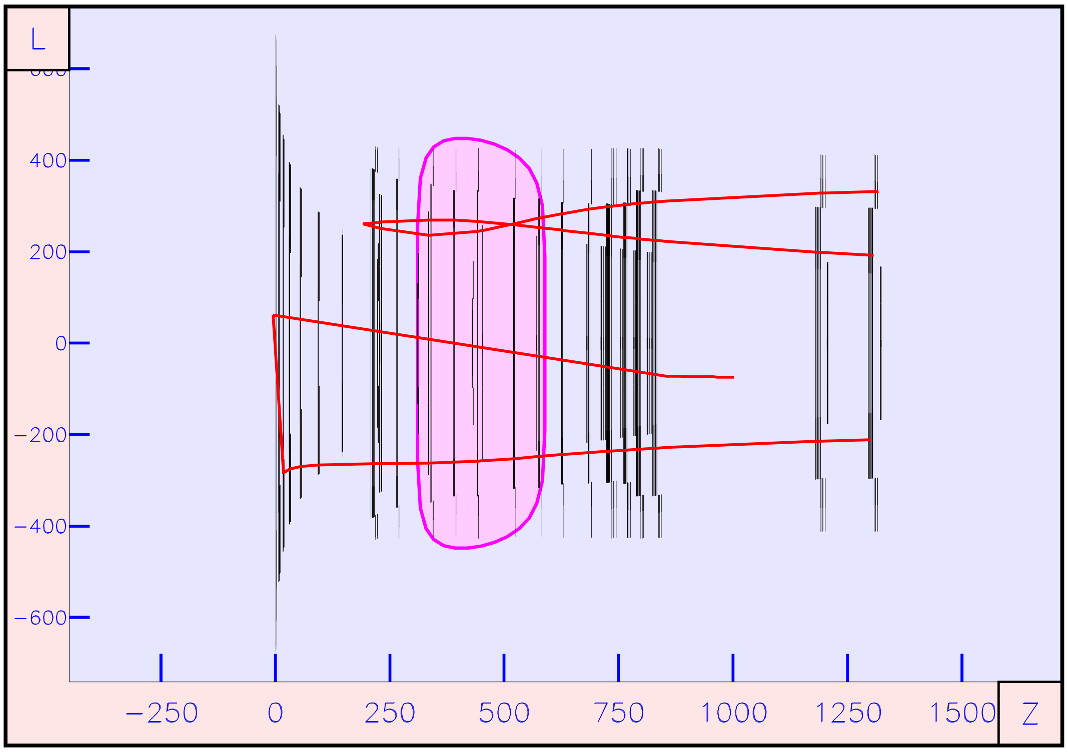


Figure 2.7: Example of event display with log-cone view 10

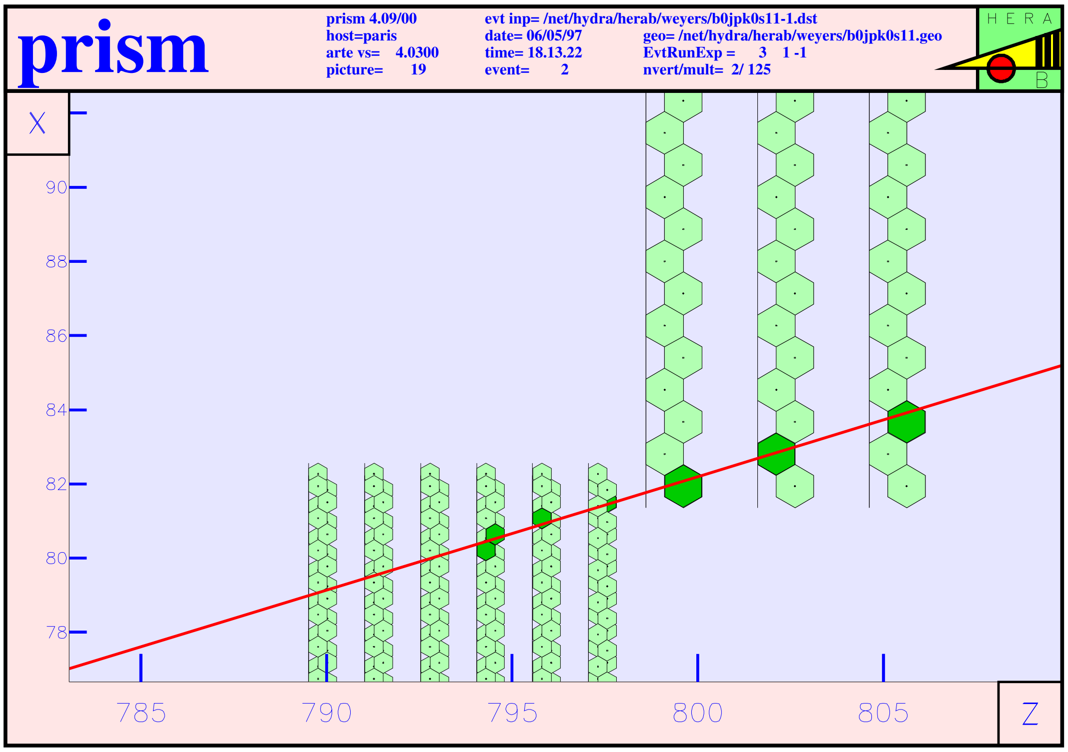


Figure 2.8: Display of the cell structure in the outer tracker

which is important for track segment matching studies. Fig. 2.7 was generated in view 10.

2.4 Style options

2.4.1 Detector geometry

Per default, each detector described in the GEDE or GCAL tables is displayed as a polygon or a straight line, depending on the projection. This feature is governed by the **plan=** style parameter. To limit the complexity of the display, the stereo layers of the main tracker are drawn only if explicitly requested with the corresponding style option:

```
display/style stereo=y
```

This rule does not apply to *all-stereo* systems like the Silicon vertex detector.

It is also possible – though more time consuming – to show the detailed structure of all detector cells by activating the corresponding style option:

```
display/style cell=y
```

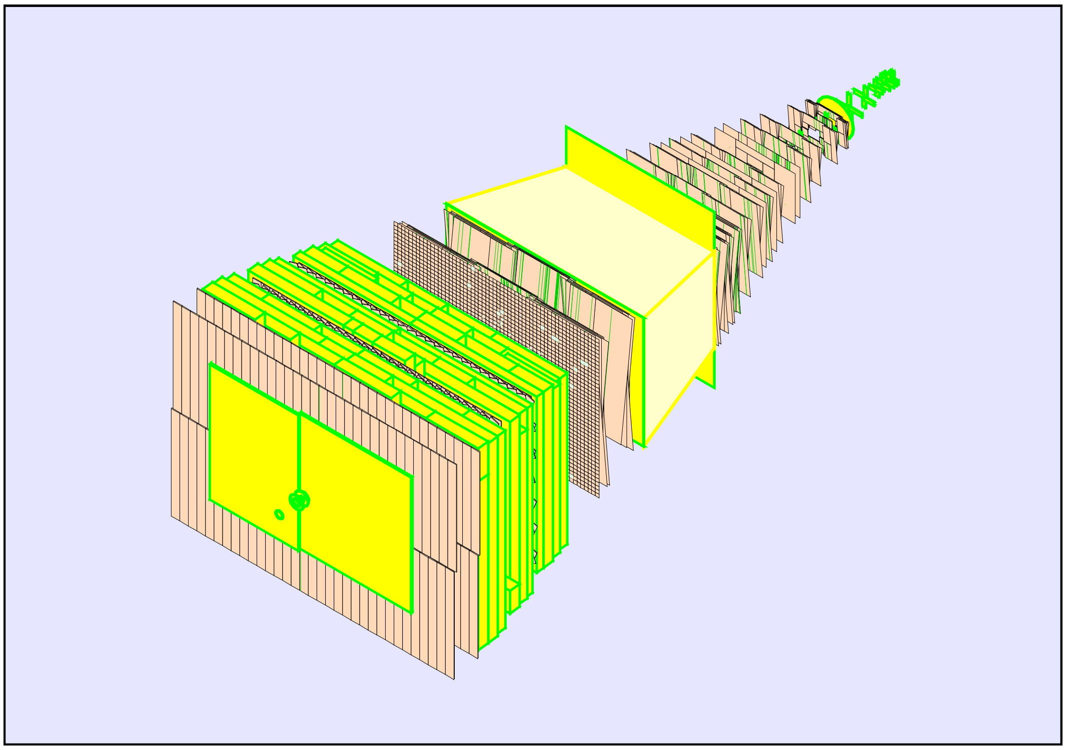


Figure 2.9: Display of the HERA-B detector including the walls implemented in the current geometry description.

This feature is so far only honoured by the honeycomb drift chambers. The routine performing this work was provided by Alexander Lanyov (Dubna), at present it operates for wires parallel to the view direction. A result can be viewed in fig. 2.8.

2.4.2 Walls

The term *walls* refers to objects of insensitive material. Examples are the end plate of the vertex detector vessel, or the muon absorber shield. The display of these walls can be activated with the switch

```
display/style walls=y
```

A picture of the HERA-B detector with walls displayed is shown is fig. 2.9.

2.4.3 Detector hits

There are two options governing the way detector hits are displayed. The command

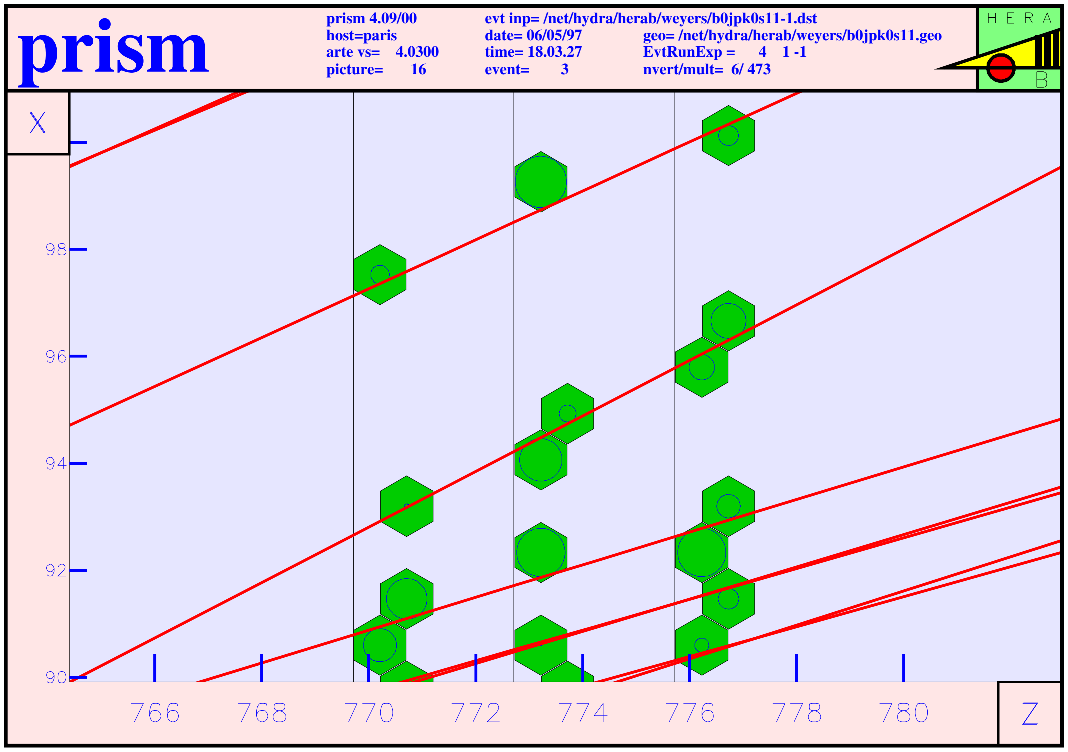


Figure 2.10: Closeup of the outer tracker with both **wire** and **quant** options activated.

```
display/style wire=y
```

switches display of hits (HITB or HITC) on. An additional option

```
display/style quant=y
```

determines if a quantitative measure associated with the hit should displayed. For drift chamber devices, this quantity is the measured drift-distance. An example of an outer tracker display with both `wire` and `quant` options on is shown in fig. 2.10.

The way drift-distance isochrones are displayed depends on the view relative to the wire direction. In the xz view (view 1), the zero degree layers isochrones of the honeycomb tracker are displayed as circles (fig. 2.11 top). In the yz view (view 2) the stereo layers isochrones are displayed as ellipses (fig. 2.11 bottom). In the uz and vz views (view 7 and 8), the isochrones of the corresponding stereo view are displayed as circles.

2.4.4 The target

The target ribbons can be displayed if their geometry is present in the GTAR table, by activating the style option

```
display/style target=y
```

Fig. 2.12 shows a typical target display. Note that the target is much thinner in x than in z.

2.4.5 Neutral particles

Normally, neutral Monte Carlo particles are not displayed to keep the complexity of the image at a digestible level. Their rendering can be activated through the extra switch

```
display/style neut=y
```

Only a limited choice of neutral particles is displayed: photons, K_S, K_L, neutral B and D mesons, neutrons. Only those particles having begin *and* end vertices inside the *Arte* vertex table can be shown. Fig. 2.22 gives a typical picture.

2.4.6 Decay chains

The option `chain` in the `display/style` command directs *prism* to display Monte Carlo particles along with their full decay chains. This feature makes only sense when some subset of Monte Carlo particles has been selected (see *Parameters* section). An example was already given in fig. 2.3.

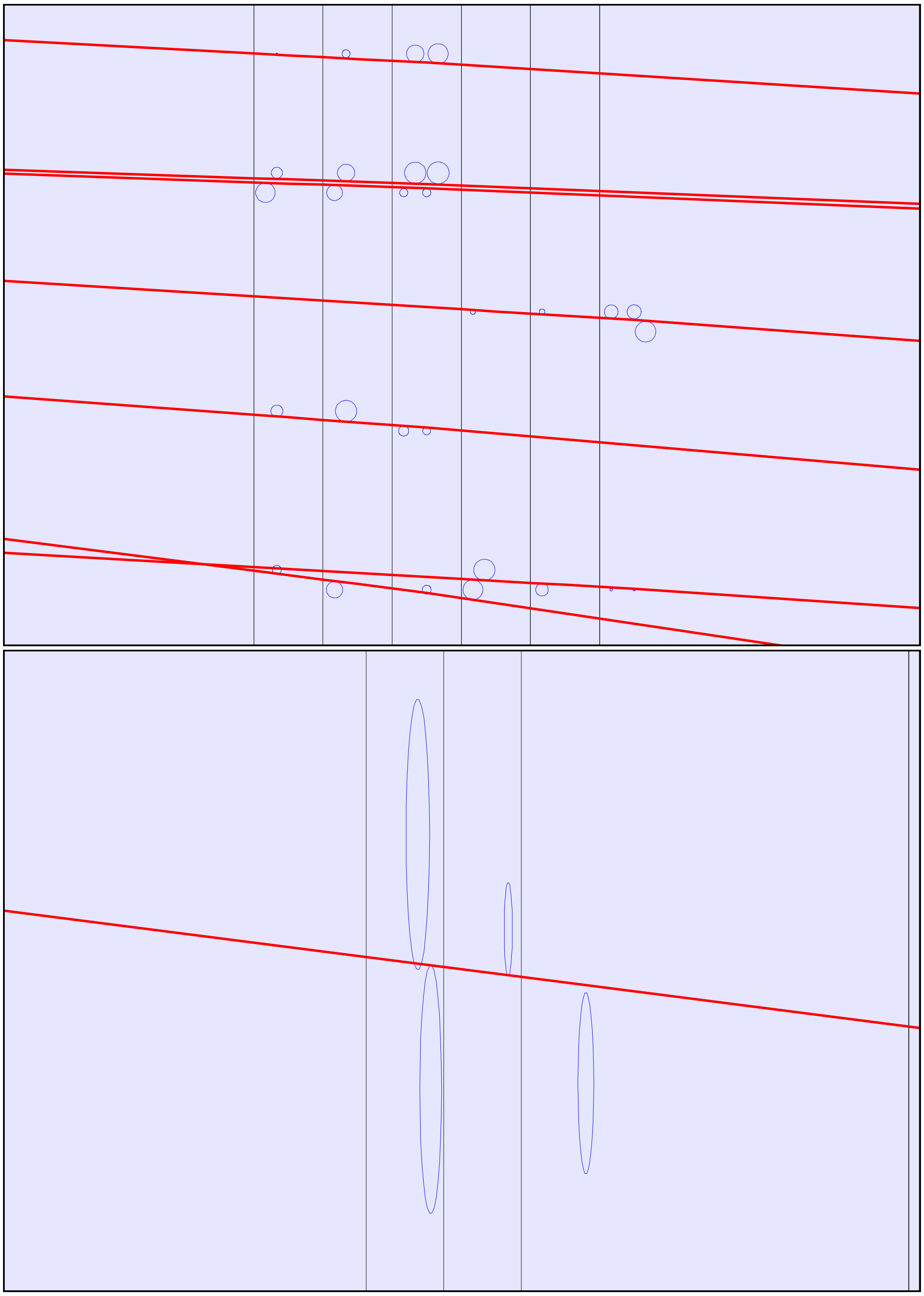


Figure 2.11: Isochrones of zero degree layers (top) and stereo layers (bottom)

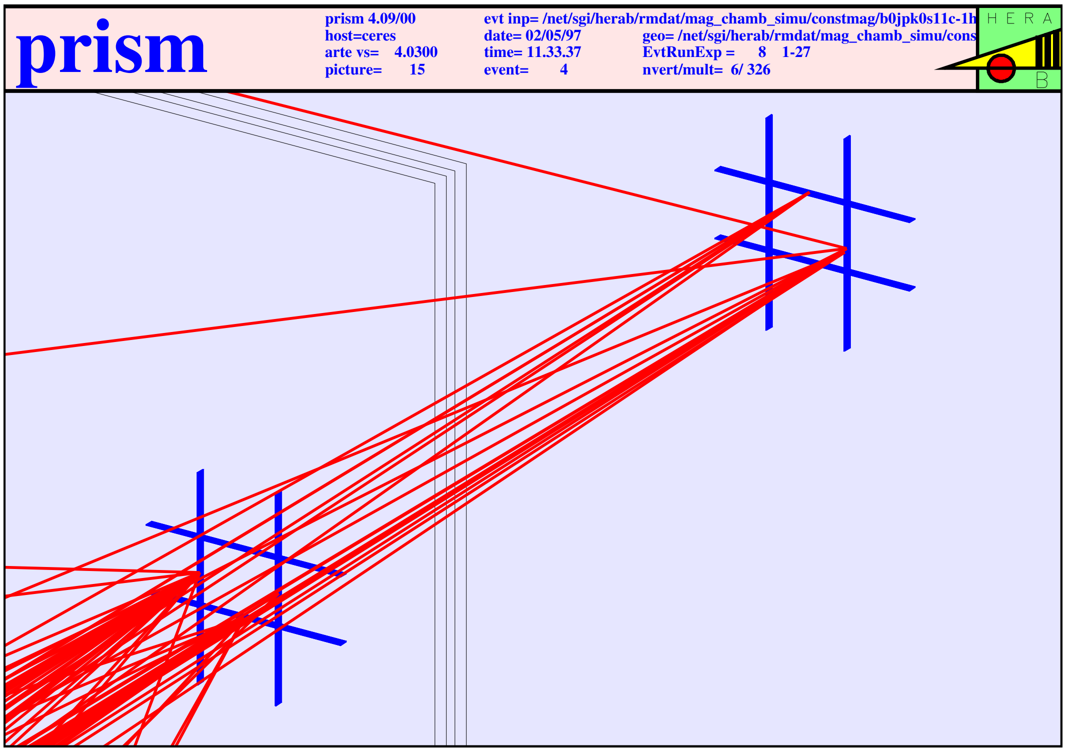


Figure 2.12: Display of the target ribbons with interaction vertices.

2.4.7 Reconstructed tracks and segments

If reconstructed tracks are available (and the RHIT table is filled), they can be display by setting the option

```
display/style rc=y
```

The additional option `cross` draws a cross marker at the position of each reconstructed hit. In the same way, reconstructed segments can be draw by activating the `RSEG` option.

2.4.8 Reconstructed vertices

If vertices have been reconstructed and placed into the RVER table, their covariance ellipsoid can be displayed by activating the `rver` option:

```
display/style rver=y
```

The routine performing this display was provided by Ivan Belyaev (ITEP Moscow).

2.4.9 Appearance of the display box

A number of options determine if a header (`header`), the coordinate system (`axis`), the display box itself (`frame`) or its background (`background`) are to be drawn.

Additional style options, many of them less interesting to the casual user, are listed in the command reference section of this manual.

2.5 Detector components

In this section we describe briefly the mechanism of selecting detector parts for display, followed by a more detailed description of the way specific detector components are displayed. All styles of rendering are completely open to suggestions and contributions from the responsible detector groups.

2.5.1 Setting the scope

The *scope* concept is used to tell *prism* which parts of the detector are to be displayed. *prism* converts scope descriptors into a list of active superlayers (GESLs). The argument of the `display/scope` command may be a list of superlayer names (the way they are stored in the GESL table), or of shorthand notations for groups of superlayers. For example,

```
display/scope ipat opat
```

SIVD	all silicons with z< 200cm
MAIN	all silicon, MSGC and HDC trackers with $z > 200$cm
FULL	all silicon, MSGC and HDC trackers
IMAIN	all inner main trackers
OMAIN	all outer main trackers
MAG	all magnet trackers
IMAG	all inner magnet trackers
OMAG	all outer magnet trackers
PAT	all pattern trackers
IPAT	all inner pattern trackers
OPAT	all outer pattern trackers
TRIG	all trigger trackers
ITRIG	all inner trigger trackers
OTRIG	all outer trigger trackers
ECAL	calorimeter
MUON	all muon chambers
PT	all high-pt trigger chambers

Table 2.2: Table of shorthand notations for operands of the `display/scope` command

activates the pattern tracker superlayers PC01–PC04 and MS10–MS13. A list of shorthand notations is given in table 2.2. If `display/scope` is issued without additional options, all superlayers are first deactivated and then those referenced in the list are activated. If the option `-i` is given ('on'), the referenced superlayers are activated in addition to those already active. If the option `-o` is given ('off'), the referenced superlayers are deactivated. Example: after activating the pattern tracker with the previous command, the command

```
display/scope pc02 pc03 ms11 ms12 -o
```

would deselect the two middle pattern tracker superlayers of inner and outer tracker, leaving only those superlayers used for first level triggering.

```
display/scope -l
```

produces just a list of all superlayers indicating which superlayers are active.

2.5.2 The Silicon Vertex Detector

A typical perspective view of the silicon vertex detector system is shown in fig. 2.13.

If the `walls` style option is active, the Beryllium support structures are visible rather than the Silicon wafers themselves. Fig. 2.14 shows an overview of the whole detector with the vertex detector in front and the muon chambers looming up at the rear.

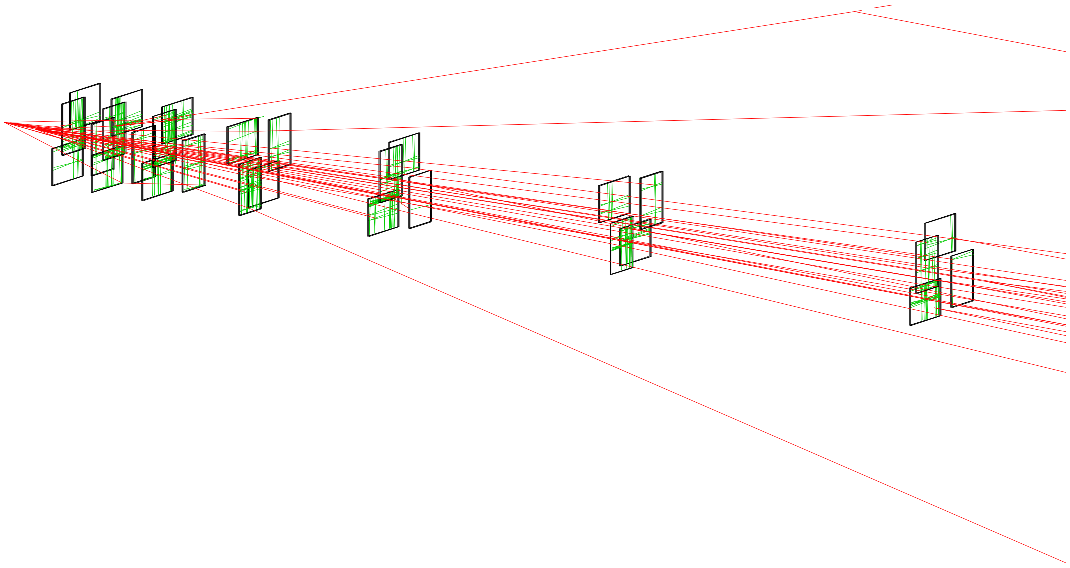


Figure 2.13: Perspective view of the Silicon micro vertex detector. This is an event with less-than average multiplicity.

2.5.3 The Outer Tracker

A perspective view of the outer tracker is shown in fig. 2.15. Display of wires, cells and drifttime isochrones are supported for the honeycomb chambers and have already shown in earlier sections where these features were explained (see figs. 2.4, 2.8, 2.10 and 2.11).

2.5.4 The Inner Tracker

A trimetric closeup from the inner tracker is displayed in fig. 2.16. The fine granularity and the high track density prevalent in this high resolution device present a particular challenge to a combined display together with other devices. The compromise chosen for *prism* works the following way: if the `wire` option is active, green circles with a size independent of the resolution and scale are displayed at the hit coordinate, these are clearly visible even in overview displays. If, on the other hand, the `quant` style option is activated, the inner tracker hits are shown by tiny blue rectangles, the width of which corresponds to the hit resolution (fig. 2.17).

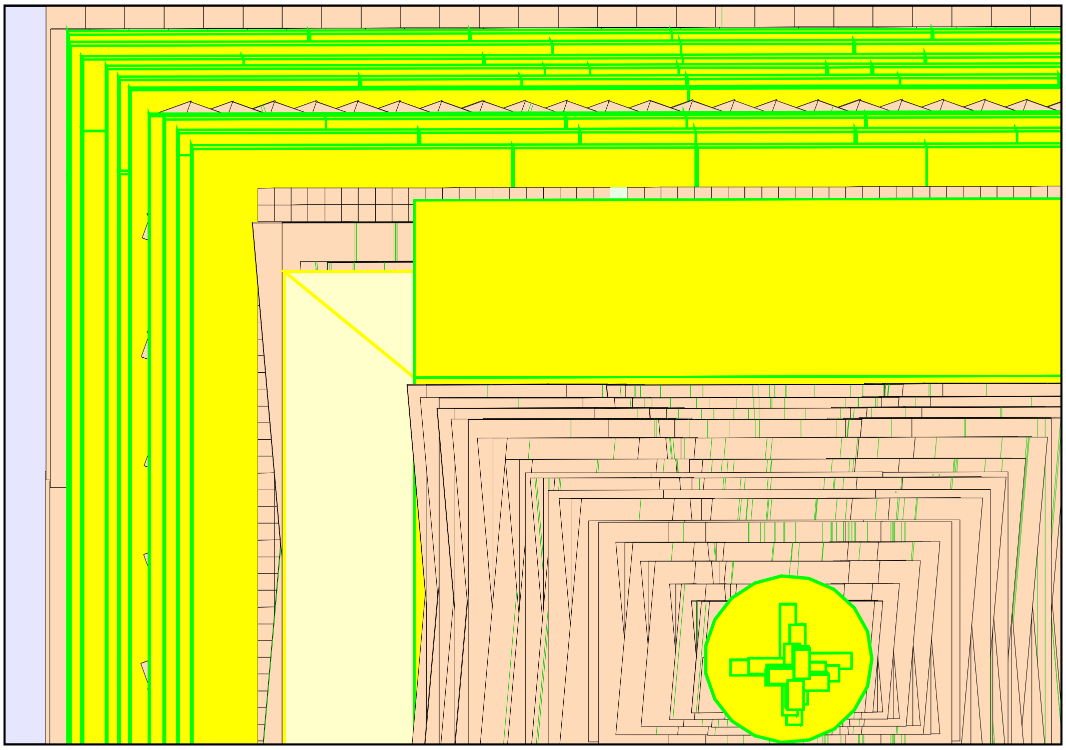


Figure 2.14: Perspective view of the whole detector with the `walls` option on. In the vertex detector region, the Beryllium supports of the Silicon planes and the exit window of the vertex vessel are clearly visible.

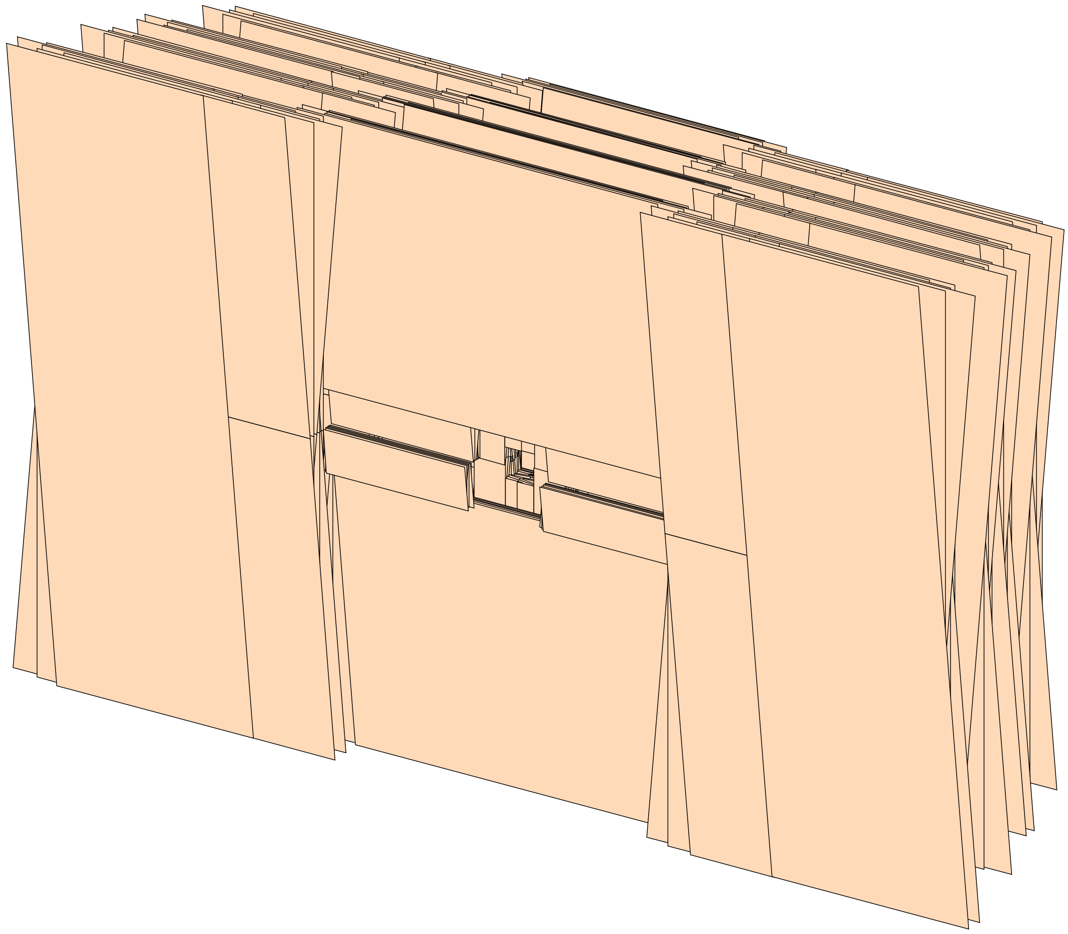


Figure 2.15: Perspective view of the pattern recognition part of the outer tracker. The inner tracker is also visible.

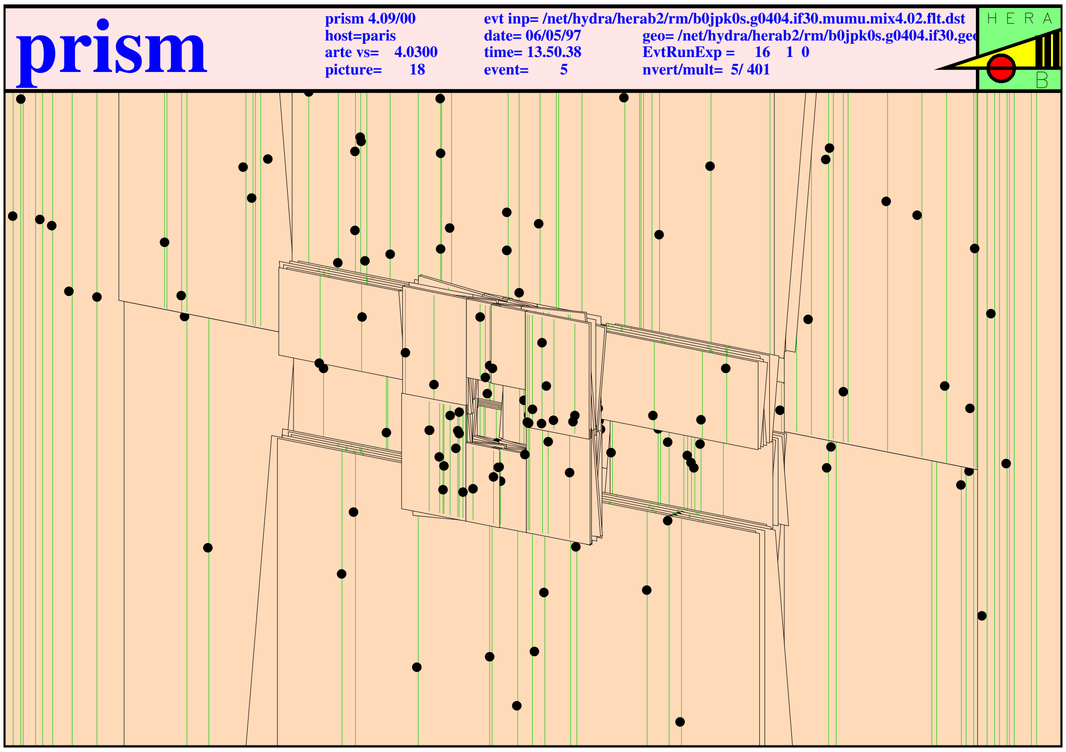


Figure 2.16: Perspective closeup of an inner tracker superlayer.

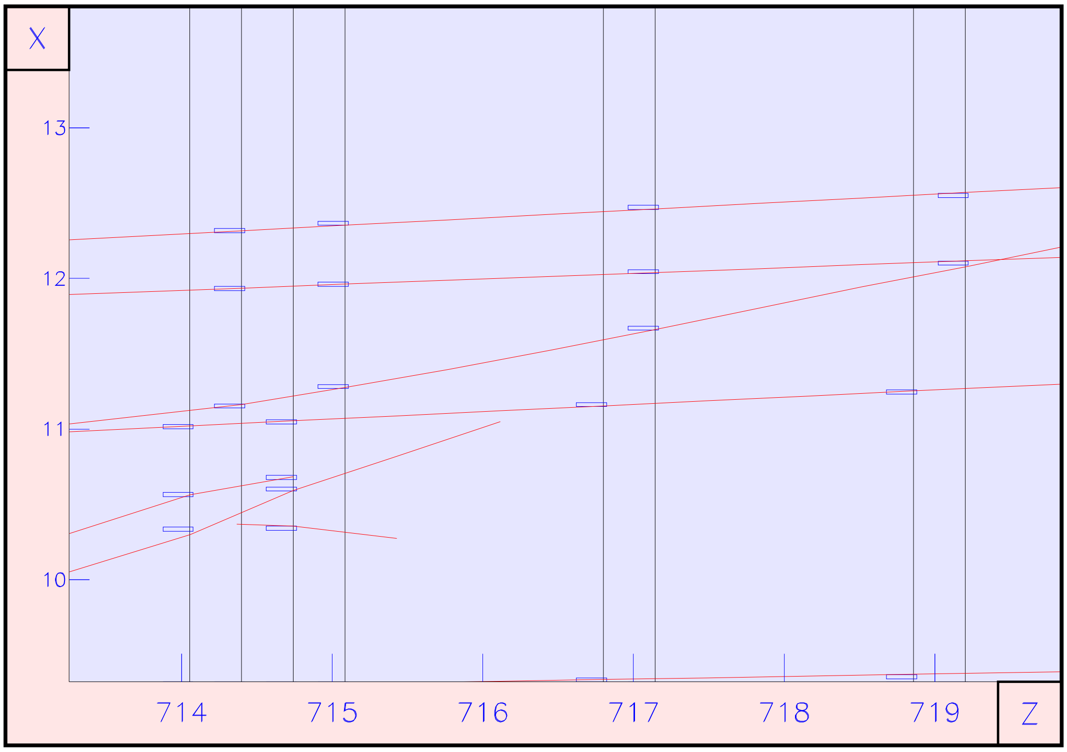


Figure 2.17: Hits in the inner tracker, displayed with the **quant** style option.

$\epsilon = \log_3 E[GeV]$	E [GeV]	Colour index	Default value
$\epsilon \leq -2$	0 – 0.11	ca1	light green
$-2 < \epsilon \leq -1$	0.11 – 0.33	ca2	medium green
$-1 < \epsilon \leq 0$	0.33 – 1	ca3	full green
$0 < \epsilon \leq 1$	1 – 3.33	ca4	yellow
$1 < \epsilon$	3.33 – ∞	ca5	red

Table 2.3: Colour values used in the ECAL display

2.5.5 The High p_T Trigger Chambers

The display of the high p_T trigger chambers is still in an experimental state. You may try it, though.

2.5.6 The Electromagnetic Calorimeter

prism contains an ad-hoc solution to display the ECAL hits, which is likely to need some refinement through input from the calorimeter subgroup. The even-sized blocks are shown as squares. Each ECAL hit with an energy above 10 MeV is displayed as a square corresponding to its cell geometry. The fill colour is adjusted according to the deposited energy. A legend is given in table 2.3. Fig. 2.18 shows a cluster created by a 40 GeV electron.

2.5.7 The Muon System

The perspective view of the muon system – including its absorber shield – in fig. 2.19 reveals an already very detailed implementation of the real geometry [5]. The muon system provides three kinds of hits: wires (both in the tube and pad chambers), pads (in the pad chambers only) and pixels (in the pixel chambers which form the inner muon tracker). Fig. 2.20 shows wires hit in the first tube superlayer MUD1, most of which originate from highly energetic muons from a golden B decay. A view of the pad chambers in the same event is shown in fig. 2.21. Note that in this case, both wire and pad coordinates are available from the hit.

2.6 Parameters

While the *style* options are mostly just flags which can be toggled, the *parameters* specification is used to set quantitative operands which limit the range of the display. For example, the command

```
display/parameters mc=5
```

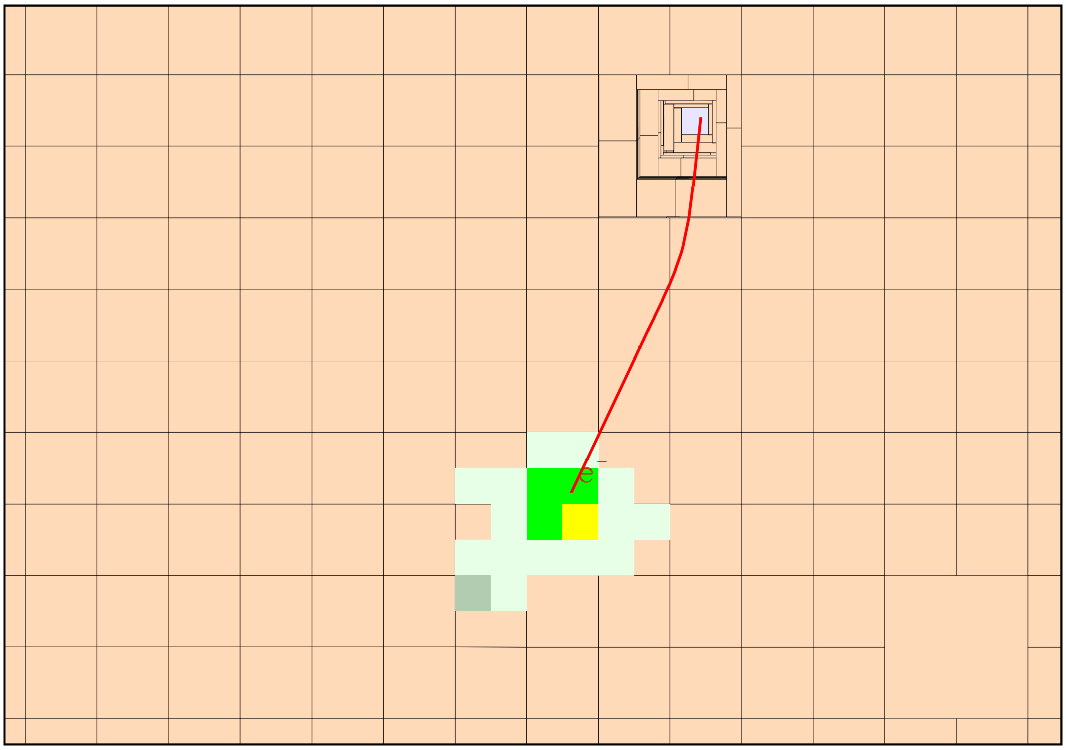


Figure 2.18: Event display showing the cluster of a 40 GeV electron in the 2x2 segmentation section of the calorimeter. The inner tracker is visible through the beam opening.

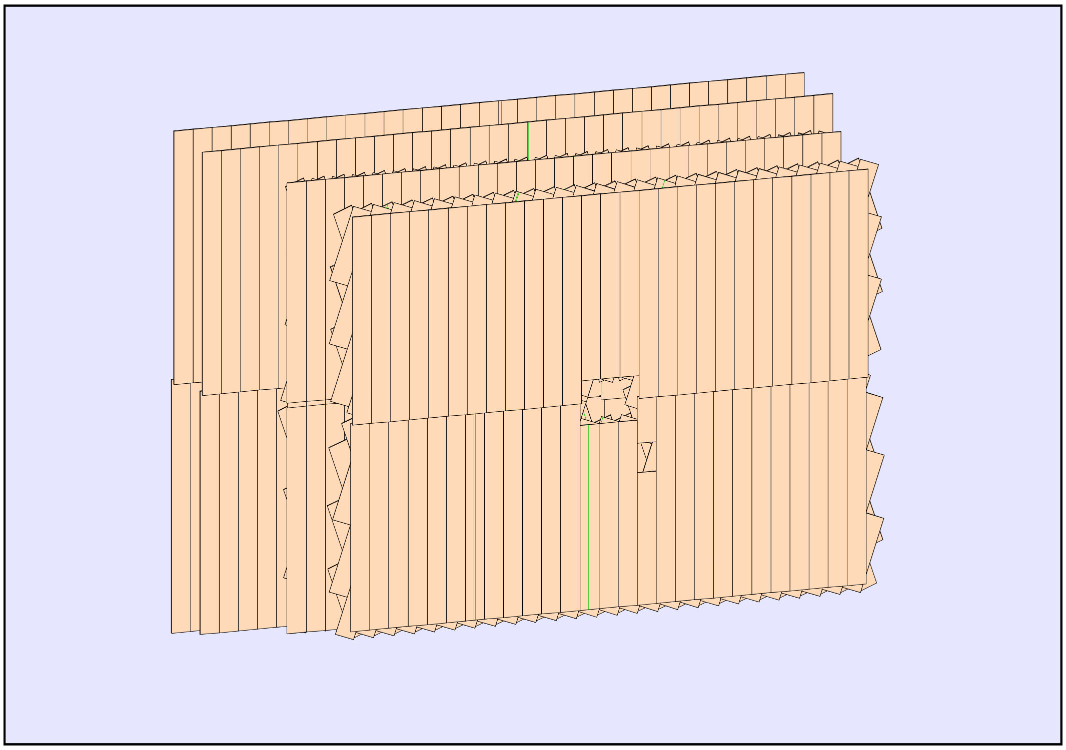


Figure 2.19: Perspective display of the muon system superlayers.

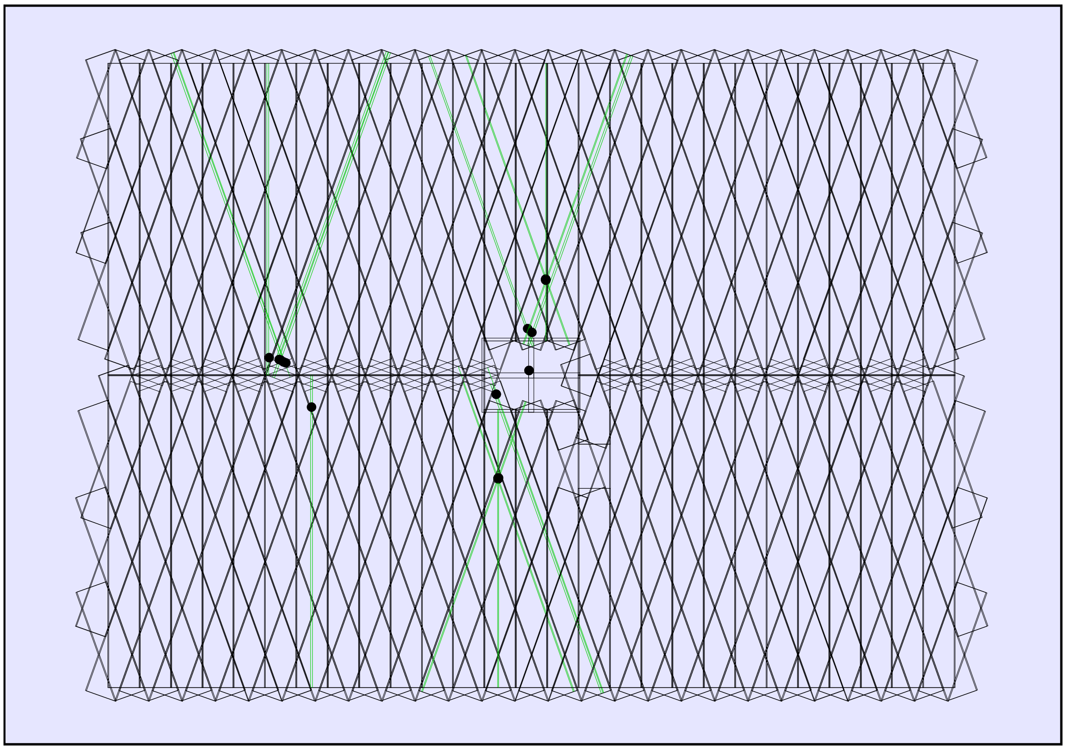


Figure 2.20: Display of muon tube chambers.

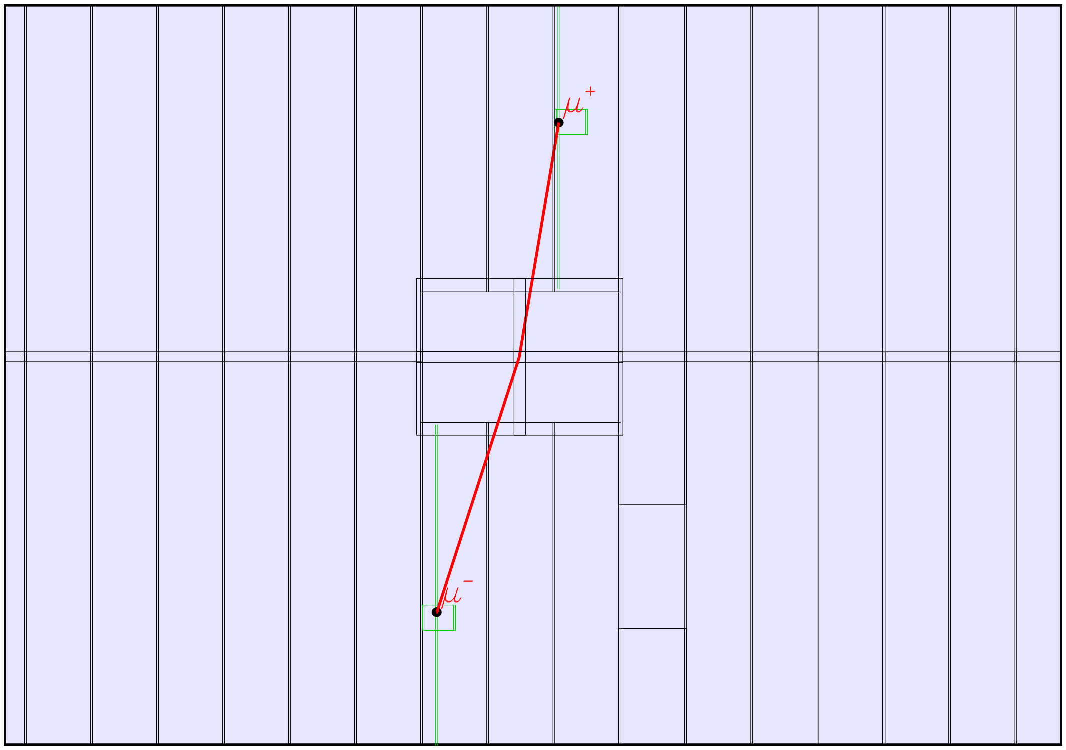


Figure 2.21: Display of hits in the muon pad chambers.

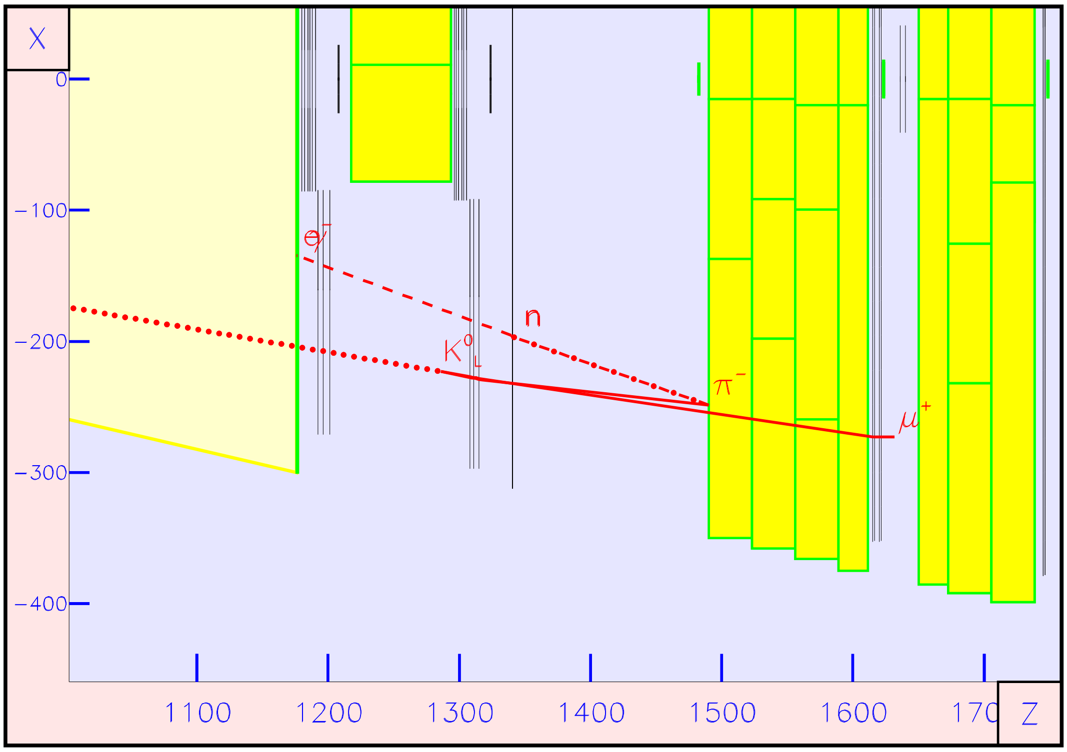


Figure 2.22: Event display restricted to long-lived neutral kaons and their daughter particles, showing a (rare) in-flight decay of a K_L^0 within the HERA-B detector. Note the backward-moving neutron generated off the the pion's interaction in the muon absorber shield.

e-	mu-	tau-	gamma	pi.	pi+
eta	rho.	rho+	omega	eta'	f_0
a_0.	a_0+	phi	a_1.	a_1+	f_2
f_1	a_2.	a_2+	K.	K+	K_L
K_S	K*.	K*+	D.	D+	D_s+
D*.	D*+	D_s*+	D_0*.	D_0*+	D_s0*
B.	B+	B_s.	B_c+	B*.	B*+
B_s*.	B_c*+	B_0*.	B_0*+	B_s0*.	B_c0*+
eta_c	Jpsi	eta_b	Upsilon	p	n
Delta++	Delta+	Delta.	Delta-	Lam.	Lam_c+
Lam_b.	Sig+	Sig.	Sig-	Sig_c++	Sig_c+
Sig_c.	Sig_b+	Sig_b.	Sig_b-	Xi.	Xi-
Xi_c+	Xi_c.	Xi_b.	Xi_b-	Xi_bc+	Xi_bc.
Xi_bb.	Xi_bb-	Ome-	Ome_c.	Ome_cc+	Ome_ccc++
Ome_b-	Ome_bc.	Ome_bcc+	Ome_bb-	Ome_bbc.	Deuteron
Tritium	Alpha				

Table 2.4: Non-exhaustive list of `ctopp` identifiers in *Arte* . Anti-particle identifiers for neutral particles are composed by appending a $\sim$.

will limit the display to the Monte Carlo track in row 5 of the MTRA table. Similarly, the display can be restricted to a certain reconstructed track, a specific detector component or sector number (see command reference section for details). A frequently used feature is to limit the display to a special kind of particle by setting the `ctopp` parameter. E.g., the command sequence

```
display/parameters ctopp=K_L
display/style mc=y neut=y chain=y
```

can be used to display all K_L^0 particles including the daughter particles. An example is given in fig. 2.22. A list of valid `ctopp` identifiers can be obtaines with the *Arte* command `ltopp`. A selection of these identifiers is given in table 2.4.

2.7 Picking

A very efficient way to analyze the objects shown in an event display is provided by 'picking' the objects directly with the mouse. After selecting the kind of object to be picked, the user points at the object of interest with a mouseclick. *prism* then prints information about the picked object to the line output and redraws the object in *highlight* mode to indicate which object has been picked. In fig. 2.23, the user has picked the third particle from the top. The information printed by *prism* may look like

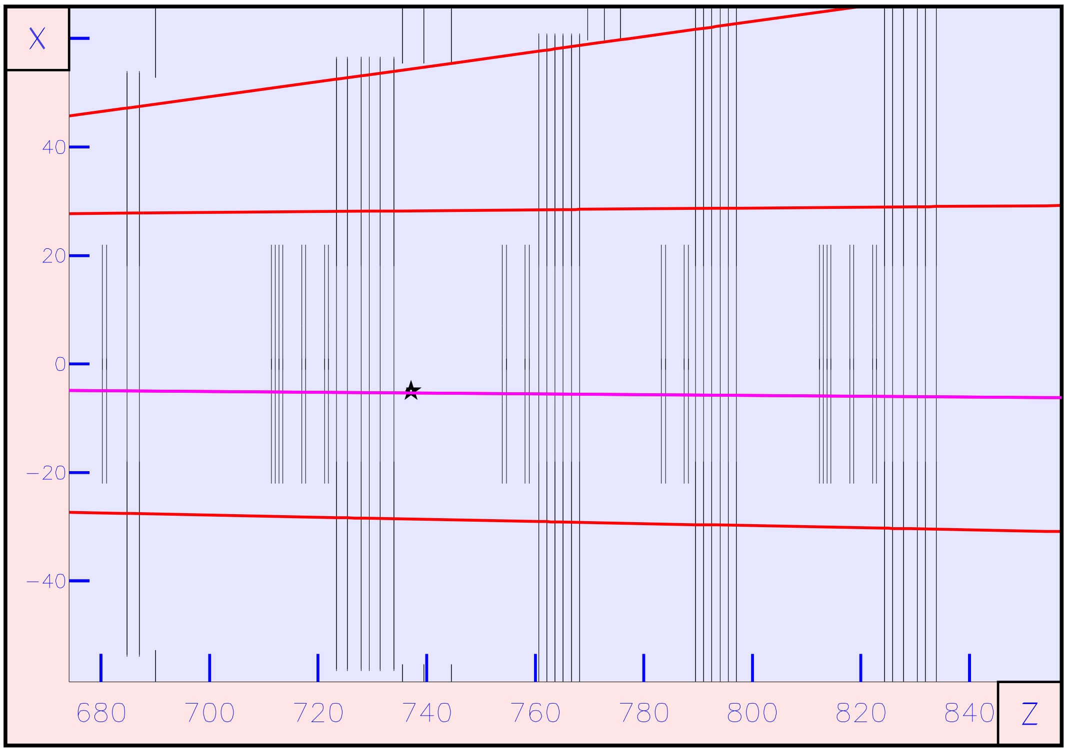


Figure 2.23: Tracks from the golden B decay in the pattern recognition area of the main tracker. The third track from the top (purple) has been picked.

```
********************************************
*      mc=    83             name=mu+      *
*       p=    29.23            px=    0.469 *
*      py=     1.99            pz=   29.163 *
*  origin= J/Psi -> mu+ mu-                *
* destiny= mu+ -> e+                       *
********************************************
```

which gives the most important whereabouts of this track.

2.8 Display modes

Everything said so far referred to a specific display mode of *prism* , the *detector* mode. In fact, *prism* supports three different display modes:

Mode	Definition	Command
Detector	Scope of active superlayers	**display/detector**
Superlayers	Range of superlayers	**display/superlayer**
Plane	Single plane	**display/plane**

The default mode is the *detector* mode, which is sufficient for normal usage. In this mode, the *scope* of active detector elements determines which parts are shown. The *plane* mode can be useful when a single detector plane has to be analyzed. Note however that each of the three commands switches to its specific mode. *prism* will remain in this mode until a command switching to a different mode is given. Example: if you issue the command

```
display/superlayer 1 7
```

then the first seven superlayers will be shown, regardless of the *scope* list. Every subsequent

```
display/draw
```

command will redisplay the first seven superlayers. If you wish to go back to the mode steered by the *scope* list, you first have to go back to the *detector* mode:

```
display/detector
```

The workstation clear which is normally issued by **display/detector** and the other display commands can be suppressed with option **-a** (=add), allowing to superimpose plots. It is useful to suppress header (option **-h**) and axes (option **x**) for the superimposed graph.

Most colour assignments of the display can be taylored by the **set/colors** command. See the command reference section for details. A table with the default colour assignments used in *prism* is given in the appendix (table B.1).

2.9 Saving pictures

2.9.1 PostScript

All pictures produced on the display are by default written to a PostScript metafile with the name defined at initialization time (see description of `prism` command above), the default name is `events.ps`. After the display session, this file can be browsed e.g with *ghostview* or *psselect* to select pictures for printing.

In principle, HIGZ applications can also save pictures in HIGZ metafile format, resulting in a picture database which can be converted to several other formats, like encapsulated PostScript or latex metafile using the "higzconv" utility from the CERN library, and which is readable by PAW. This feature was built into an earlier *prism* version 4.3 but has been removed since it was apparently not used.

2.9.2 Encapsulated PostScript

Since the HIGZ metafile driver for encapsulated PostScript (EPS) does not support output files with more than one picture, *prism* writes the pictures to the PostScript file without bounding box. Since the *prism* picture format is always the same, it is easy to insert the bounding box by hand into the selected pictures. The following should be inserted after the first line into the PostScript file:

```
%%BoundingBox: 95 136 496 702
```

If you want to include event displays into a LaTeX document, it is easier still to specify the bounding box in the epsfig command. Such a statement could look like

```
\epsfig{file=cpviol.ps,width=14cm,clip=,angle=270,%
        bbllx=95pt,bblly=136pt,bburx=496pt,bbury=702pt}
```

Note that in this case `cpviol.ps` does not have to contain a bounding box.

A useful method to obtain an EPS file with upright orientation is through using ps utilities. Example:

```
psselect -p5 events.ps >myTMP.ps
pstops 1:0R@1.(-4.5cm,25cm) myTMP.ps >myTMProt.ps
pstoepsi -bbonly myTMProt.ps event.eps
rm myTMP.ps myTMProt.ps
```

will select the fifth picture from the prism output file `events.ps`, rotate it from landscape to upright orientation and add a bounding box.

2.10 Querying tables with *list* commands

The `list` menu supplies a number of commands which print information from the arte tables. They complement the list commands provides by *Arte* in the same menu. Printing information from one table requires simply a command like

```
list/mtra 3000 3200
```

which prints rows 3000–3200 from the MC track table (MTRA). If the range is omitted, all rows are printed. For some tables the Xterm window should be set to maximum width and smallest font to still get a one-line-per-row output.

The particular advantage of some print commands is however that they can follow up relations between the tables, e.g.

```
list/mimp mtra=5
```

prints all Monte Carlo impact points related to MC track 5 (one-to-many relation).

```
list/hitb mimp=3299
```

prints all hits from the hitb table which are related to mimp 3299 (many-to-many relation). This works also in reverse: the command

```
list/mimp hitb=1304
```

prints the mimps from all MC particles which have "caused" the hit. The command

```
list/rtra -m
```

does not only print the reconstructed tracks (from the RTRA table), but also prints a matching MC particle (if existent) and its track parameters for comparison. The covariance matrix may be obtained by adding the `-c` option. Note that all options and parameters involving MC information work only in Monte Carlo events - otherwise the MC banks are empty and the output is void.

Please refer to the command reference section for a full description of the *prism* `list` commands.

Chapter 3

The *prism* Commander – a Tcl/Tk Based User Interface

The *prism Commander* was introduced with *prism* 4.4. It has been completely redesigned to its present attractive form by Arnim Wagner (a student from the Humboldt University, Berlin).

In the command mode, the user interface is handled with the KUIP package [4] from the CERN library. The fact that the *Arte* uses the same user interface allows the user to intermix freely *prism* and *Arte* commands from the same command input. Moreover, dedicated display tools for event processing modules, e.g. track reconstruction, detector simulation etc. can be integrated with additional command sets in a perfectly modular way.

On the other hand, intensive work with the event display requires a frequent change from keyboard command input (changing view, frame or scope of display) to mouse actions on the graphics window (like click-and-drag rubberbanding to define a new frame). Also, with the display package becoming more and more powerful, the growing variety of commands and options is increasingly difficult to memorize especially for the casual user. A graphical user interface is the ideal solution to overcome both problems.

In principle there is no problem in setting up a program such that it can create widgets under X-Windows which are managed by the X11 display manager, and handle user requests indicated by *X events* encountered in these widgets. It is however difficult to match this with the general HERA-B philosophy that

1. there should be identical reconstruction executables for both interactive and batch usage

2. it should be possible to drive the program entirely in command mode

3. the event display should be able to operate *on demand*, i.e. the user should be able to invoke the event display at will at an arbitrary point in an interactive *Arte* session

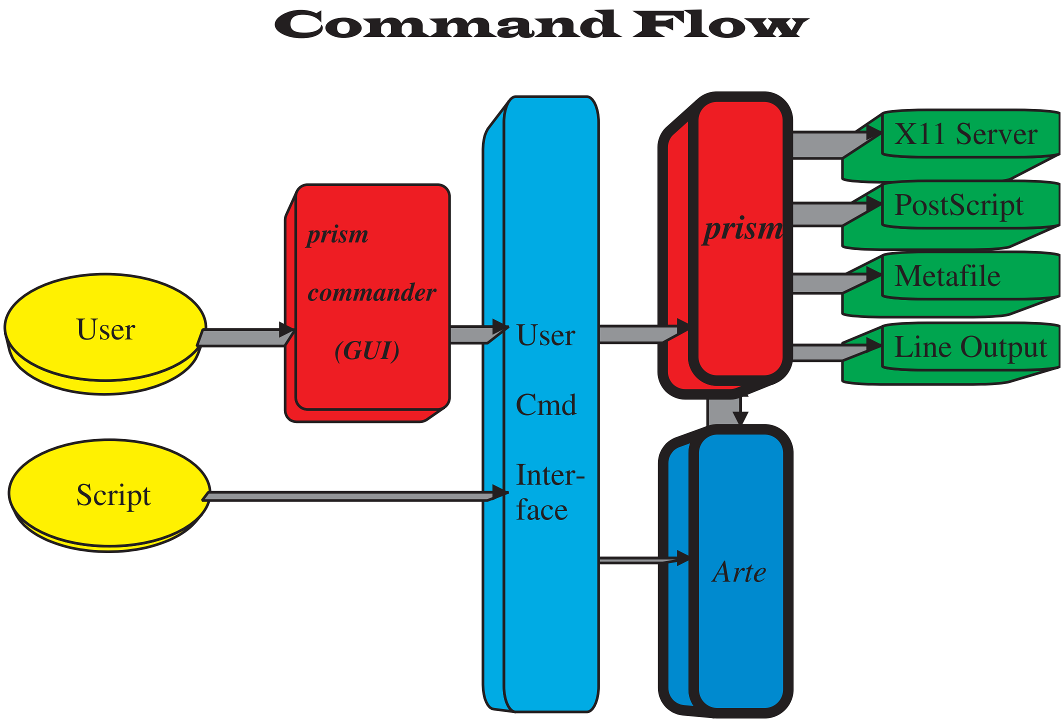


Figure 3.1: Event display command flow with integrated graphical user interface

4. the modular structure of the HERA-B reconstruction system should not be impaired or complicated by adding a graphical user interface

These seemingly contradictive requirements can be nicely met by using a Tcl/Tk script to generate a graphical user interface. Its inclusion into the command flow is shown in fig. 3.1. Normally, the user, possibly represented by a command macro linked to the module's standard input, will type *prism* and *Arte* commands into the *user command interface* (compare fig. 1.1). The latter will interprete these commands and call directly the appropriate *prism* and *Arte* routines. *prism* itself then produces directives for the graphical display (*X11 Server*), the *PostScript* output file, the *HIGZ metafile*[1] and information in form of text for the *Line Output Window*. The graphical user interface is introduced in shape of the *prism Commander*, a Tcl script which calls the reconstruction module in a process pipe and drives its actions by command lines generated from the graphical user input.

[1] Feature has been disabled

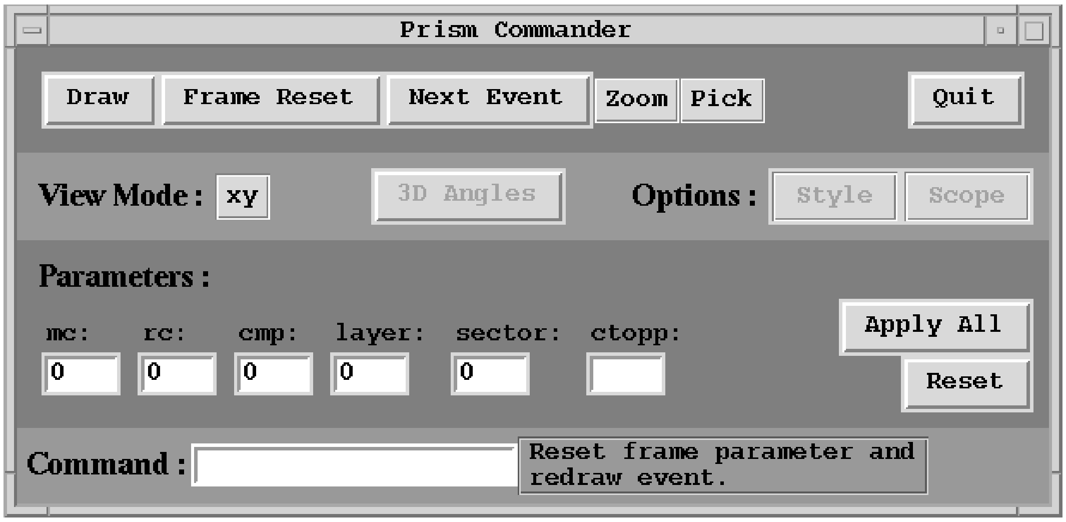


Figure 3.2: Prism commander panel

3.1 Layout of the *prism Commander*

The layout of the *prism Commander* panel with its buttons and other widgets is shown in fig. 3.2. It consists of five major sections:

1. the top section with basic steering of drawing and frame adjustment

2. the *view* section to set view direction and mapping

3. the *options* section, consisting of *style* options which determine how event data are displayed and the *scope* options which define the activated detector parts

4. the *parameters* section in which parameters restricting the displayed range of entities can be given, and

5. the *command entry* window for invocation of any *prism* or *Arte* command

3.2 The top section

This part of the panel steers the basic flow of actions. The `Draw` button causes rebuilding of the display with the most recent options. The `Frame Reset` button leads to a default frame which is calculated from the activated detector parts (see `Scope` button in the *View* section). The `Next Event` button causes *Arte* to read the next event from input and wait for new display demands.

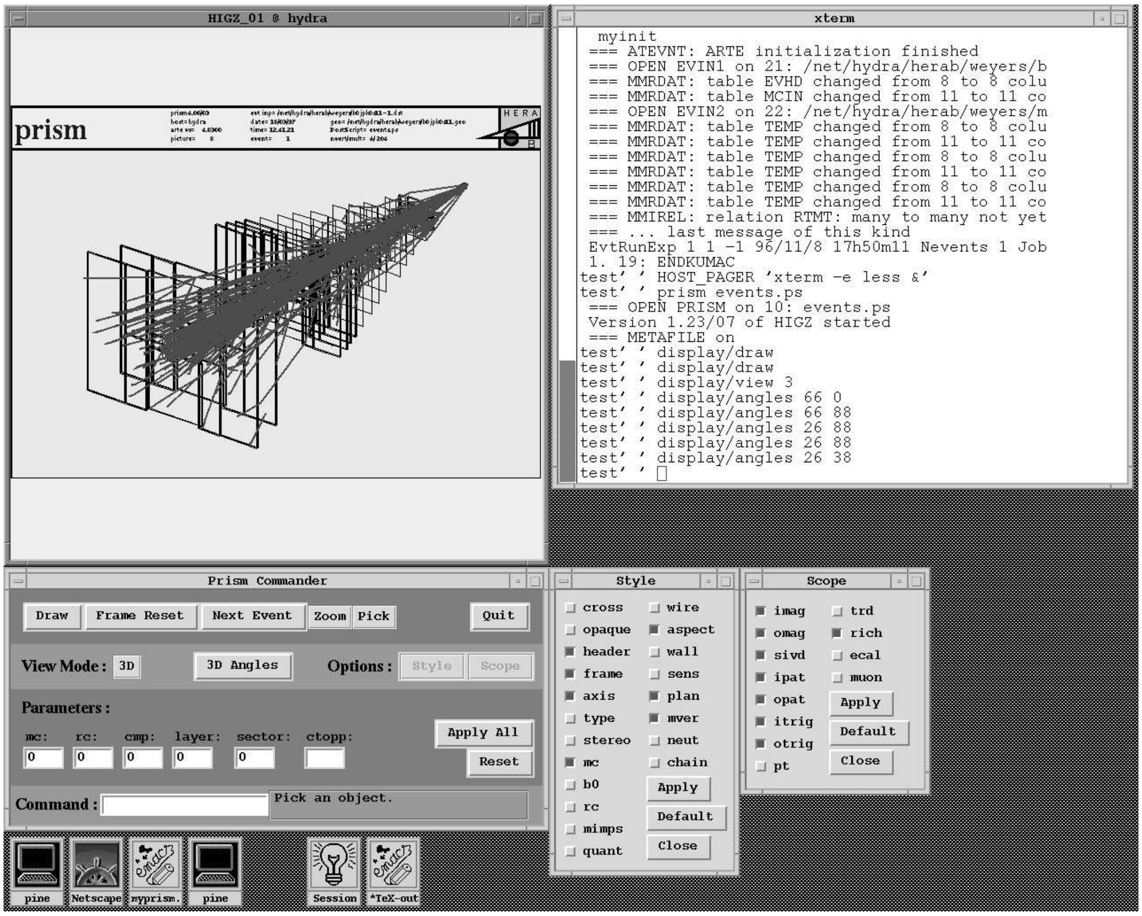


Figure 3.3: Typical appearance of desktop in a *prism* session with graphical user interface

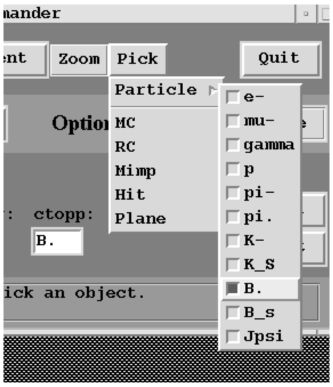


Figure 3.4: The pull-down menu for particle selection

The `Zoom` button provides a menu which leads to a choice of zoom commands. The `2x` and `1/2x` entrybuttons allow to zoom in and out by a fixed ratio without additional mouse actions, the `Free Zoom` entrybutton causes the program to prompt the user for a rubberband input to define the new frame borders.

Pick stands for a pulldown-menu which is separated in two parts. The `Particle` menu-entry (fig. 3.4) provides another cascade-menu with a set of radio buttons to restrict the displayed Monte Carlo particles to a particle type out of the `ctopp` collection. (For full particle list write "ltopp" in the `Command` entryline.). Below the separator line in the `pick` menu, pickable objects are listed. Choose an object and select it per mouse-click in the *prism* display window. Information on the picked object is (at the time of writing) available in the XWindow from which you started *prism* . The `Quit` button terminates the session.

3.3 The *view* section

On the left side of this section is the *View Mode* menubutton which set the observer's direction of view. Pressing the button right of the *View Mode* label provides a menu to choose the view mode (fig. 3.5). Please refer to the explanation of the `display/view` command in the first part of this note for the meaning of the various choices.

When choosing the *View Mode* `3D`, the `3D Angles` button becomes active. Pressing this button opens a new window (fig. 3.6): the two sliders in the *Angle Selection* window are used to define arbitrary viewing angles. The selected angles become effective when the `Ok` button is operated.

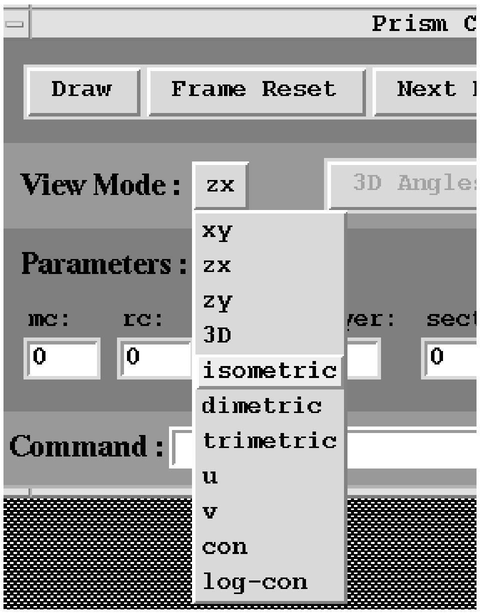


Figure 3.5: The view selector pop-up menu

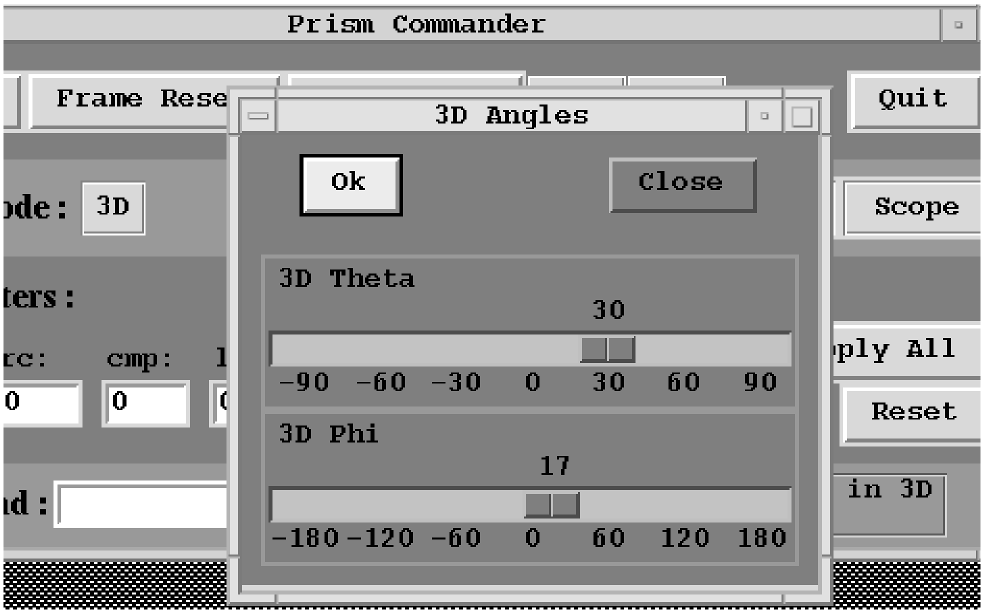


Figure 3.6: The angles selector window

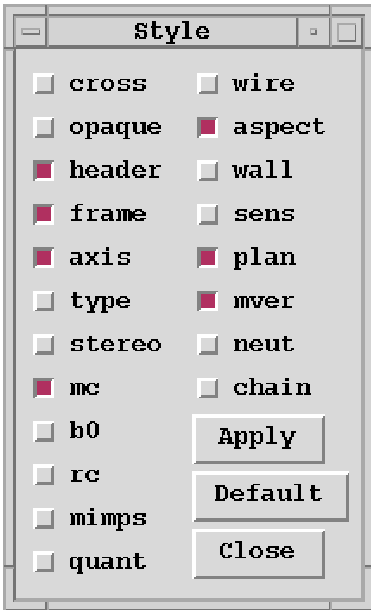


Figure 3.7: The style selector panel

3.4 The *options* section

Pressing the **Style** button opens a new window - the *Style Selection* window (fig. 3.7). It contains buttons which allow to toggle a number of display options which correspond mostly to the command line operands of the **display/style** command (see earlier section in this note).

Pushing the **Scope** button also opens a new window on the screen, named *Scope Selection* window (fig. 3.8). Its buttons are used to define which detector parts are to be displayed. The scope is divided into the Silicon vertex detector (**sivd**), the inner (**imag**) and outer (**omag**) magnet main tracker, the high-pt trigger chambers (**pt**), the pattern recognition part of the main tracker (**ipat** and **opat**), the trigger chambers (**itrig** and **otrig**), the electromagnetic calorimeter (**ecal**) and the muon tracker (**muon**). Included but currently without function are switches for the transition radiation detector (**trd**) and the ring imaging cherenkov detector (**rich**). After changing the scope, the **Apply** button can be used to display the frame accordingly.

3.5 The *parameters* section

This region is used to enter various parameters which restrict the range of displayed objects. For example, the parameters **mc** and **rc** restrict the display to a specific simulated or reconstructed particle. The **cmp**, **layer** and **sector** pa-

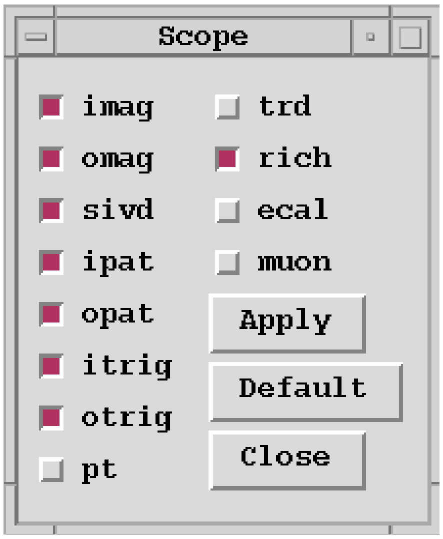


Figure 3.8: The scope selector panel

rameters restrict the display to a specific component type, layer or sector of the tracking system. The **ctopp** field takes a particle type in the standard HERA-B nomenclature (a short-cut selection can be made in the *Pick* pull-down menu – see above). A number entered in this area becomes effective after either the *Return* key is pressed in its entry widget or the **Apply All** button is operated. The **Reset** button sets the default parameter values.

3.6 The *command* section

A text string typed into the entry field following the *"Command:"*- label is directly relayed to *prism* 's command input when the *Return* key is given. This is very useful for less frequently used commands which are not bound to buttons in the other *Prism Commander* sections. The command entry field can also be used to type in answers to 'questions' of the executed program.

On the right side of the command field, a message field displays a short information string about the widget currently under focus.

Figure 3.3 shows the typical look of the user's desktop when operating the *prism Commander* .

Chapter 4

Practical Help

4.1 Online documentation

You can use the help command to get a list of available commands and their online descriptions. E.g.

```
help d/det
```

will explain the command `display/detector` along with all available parameters & options.

Note: the online help command may show additional commands and options not explained in this manual. They are mostly for development purposes and of no relevance to the "normal" user. Some of them may even be obsolete and defunct.

The *prism Commander* gives short explanations of buttons in its message window near the bottom of the panel. Just move the mouse cursor to the button you are interested in.

4.2 Updates

Since *prism* is in the state of continuous improvement with its capabilities growing, it is unavoidable that existing commands may change and new commands may appear.

The newsgroup `desy.hera-b.software` is used to announce updates and new releases of the *prism* package. This manual and its sequels will be installed on the WWW page of the HERA-B software subgroup. The URL is

```
http://castor.ifh.de/subgroup/software/welcome.html
```

4.3 Caveats, trouble shooting and common problems

4.3.1 X Windows authorization

Ensure that your DISPLAY variable is set to the name of the X display you are using, like for example `lhotse.physik.hu-berlin.de:0.0`, `Xtek23.ifh.de:0.0` or `hbx005.desy.de:0.0` etc. If you login remotely using non-X utilities like telnet, rlogin etc the DISPLAY variable is usually *not* set, hence *prism* does not know which server to contact. In addition, make sure your X client is authorized to connect to your X display. The best recommendation is to always use the proper X utilities like xrsh for remote login, which passes the DISPLAY variable and performs MIT-magic-cookie-authorization (don't forget to put your local host into the remote .rhosts file). Typing "xhost +" circumvents the authorization but is not recommended for security reasons (every jerk in the internet world can play pranks on you if you use this command).

Example: you are sitting in front of an X terminal logged in to the x4u2.desy.de and want to run *prism* on the hydra.ifh.de:

If you haven't done so before, add the line

```
x4u2.desy.de
```

to the file .rhosts in your home directory on the hydra. (You may have to use xrlogin or rlogin this once to do it.) If necessary create the file first but don't forget to type

```
chmod go-rwx .rhosts
```

to prevent bad people from accessing this important file. Then type

```
xrsh hydra.desy.de
```

to get an xterm running on the hydra and start your application in this window.

This method does not work anymore if you login remotely to a system with AFS home directories, because AFS always requires the password. In Zeuthen a new script **xssh** is under test which prompts for the password. On other systems, please ask your local administrator for the recommended X11 login procedure to an AFS system.

4.3.2 Obsolete CERN library versions

The 95a release of the CERN library had a bug in the HIGZ locator function, which shows during sophisticated use of the different display window and viewport normalization transformation (as is done in *prism*). We received an improved version of the HIGZ routine *irqlc* from Olivier Couet, which is included in the HERA-B standard software distribution. However, this fixed routine does not work with the obsolete 95a release of the CERN libraries, an error message "`locator on metafile`" is produced.

HIGZ also had some problems with clipping in older versions ≤ 1.22. It is therefore strongly recommended to use *prism* with at least release 96a of the CERN libraries.

4.3.3 Rendering of Monte Carlo tracks

prism uses the Monte Carlo **Im**pact **P**oints (MIMPs) generated by HBGEAN [2] together with begin and end vertex to display the trajectory of a particle. This method is ideal for analyzing the generation of hits etc in the digitization step. The technique has however some consequences for the interpretation of MC tracks displayed by *prism* :

1. Tracks from stable particles outside the detector are mostly not shown, because they have neither MIMPs – they do not hit a sensitive detector area – nor a decay vertex. This easily gives the impression that all events just fit snugly into the detector acceptance

2. Neutral particles can only be displayed if they have both a production and decay vertex

3. Low momentum particles curling in the magnetic field have a poor rendering. The spiral is only visible as such if it traverses at least two superlayers. Hence, spirals with a radius below ≈ 30 cm are only displayed as irregular zigzags. Particles spiralling in mid-air are completely invisible.

On the other hand, the alternative option of displaying the trajectories on the basis of their track parameters is problematic as soon as stochastic effects play an important rôle. Unlike many other experiments, the HERA-B detector is so large and massive that *all* types of particles with the exception of neutrinos can decay or interact in its volume with non-neglibible probability.

4.3.4 Perspective rendering

prism offers some perspective views, which can be combined with a simple hidden lines removal to give a pseudo three-dimensional image (see the `opaque` option of the `display/style` command). The algorithm behind this is fast and very simple, and not intended to give a perfect three-dimensional rendering. In fact, if you examine closely perspective pictures of more complicated arrangements, you will frequently find flaws in the way geometrical bodies appear to hide each other.

It is hence important to bear in mind that *prism* is designed as a two-dimensional display package, intended for event analysis rather than virtual reality animations. It is not foreseen to implement more complicated 3D rendering algorithms on top of the 2D display package which is used internally by *prism* .

4.3.5 Entering *prism* commands with the *prism Commander*

If *prism* is used with its Tcl/Tk interface, commands may be typed into the *Command* window of the *prism Commander*, but not into the *xterm* window in which the executable was started. Commands typed into this window will be ignored by *prism* but the shell will execute them after *prism* has finished. Hence: if you accidentally typed in something which would be disastrous if executed as a UNIX command, it is best to abort everything with ctrl-c and start over again.

4.3.6 Missing colours

It may occur that *prism* 's display window shows less colours than usual. A common reason is lack of memory of your X server when several applications are running at the same time. A typical situation is the Netscape Navigator running in parallel on an X terminal with only 8 MB of memory. The depth of colour will return when you terminate the competing applications.

4.3.7 Missing MIMPs and RHITs

Some older versions of the L1SIMU package contained dummy routines to disable the polymarker feature in HIGZ. The effect was that *prism* could not draw polymarkers, as are needed to honour the `mimp` and `cross` features of the `display/style` command. In the meantime this problem has been cured, but some old L1SIMU libraries may still be around. A quick remedy is to remove the L1SIMU library from the link path in your makefile if trigger simulation is not in use.

4.3.8 Missing walls

Some walls in the detector need a rotation matrix to describe them, for example the wedge-shaped scintillator hodoscopes used in the 96/97 runs. In the past there was no rotation matrix in *Arte* 's `GWAL` table, hence these objects were not described properly and could not be displayed. This will hopefully change in the future.

Acknowledgements

I would like to express my gratitude to those persons who actively contributed special routines to this package: Ivan Belyaev (RVER display), Alexander Lanyov (outer tracker cells) and Arnim Wagner (restyled Tcl/Tk script). I would also like to thank the authors of the packages which are indispensible for the operation of *prism* , especially Hartwig Albrecht (*Arte*) and Siegmund Nowak (HBGEAN), for their cooperation and help. The Monte Carlo events used for display in this manual were generated by Stefan Scharein and Peter Weyers. More thanks go to the authors and maintainers of HIGZ and KUIP, who made this work a pleasure. I would like to thank Fatima Streit (DESY-Zeuthen) and Olivier Couet (CERN) for their help with the CERN libraries.

Bibliography

[1] H. Albrecht et al., *ARTE. The Event Reconstruction and Analysis Tool for HERA-B*, HERA-B Internal note (1996). `http://www-hera-b.desy.de/subgroup/software/arte/arte_manual.ps`

[2] S. Nowak, *HBGEAN and HBRCAN*, HERA-B Internal Note (1995). `http://www-hera-b.desy.de/subgroup/software/hbgean.html`

[3] Application Software Group, *HIGZ – High Level Interface to Graphics and Zebra, User's Guide*, CERN Program Library Long Writeups Q120 and Y251, Geneva 1994

[4] Application Software Group, *KUIP – Kit for a User Interface Package*, CERN Program Library Long Writeup I102, Geneva 1993

[5] B. Fominykh, S. Nowak, *HERA-B Muon Detector in HBGEAN versions 04-04 and 05-01*, HERA-B note 97-075

Chapter 5

Command Reference

5.1 DISPLAY

Commands related to the event visualisation tool prism.

5.1.1 DETECTOR [islfirst isllast chopt]

ISLFIRST	I	"Id of first superlayer to be displayed"	D=0
ISLLAST	I	"Id of last superlayer to be displayed"	D=0
CHOPT	C	"Options"	D='␣'

Display detector within scope defined by the display/scope command. Optionally restrict to a given superlayer range. Options:

```
-A : add to existing plot (no workstation clear)
-B : B0 daughters only
-C : display isochrones for drift devices
-F : display found tracks only
-G : display GEANT hits (Monte Carlo Impact Points)
-I : plot and print plane info, pause after each plane
-L : wire addresses are listed while drawn
-M : draw the Monte-Carlo track
-N : display number of MC particle on the right side of the hit
-P : list MC information for each hit
-R : display reference tracks only
-T : draw the reconstructed track
-U : show only ghost tracks (with -T)
-V : show vertices
-W : draw the active wires
-0 : show only track reconstructed in 0-deg planes
-Y : show only track reconstructed in stereo planes
-Z : use polyline (Zigzag) rather than smooth curve for MC track
```

5.1.2 SUPERLAYER [islfirst isllast chopt]

ISLFIRST	I	"Id of first superlayer to be displayed" D=0
ISLLAST	I	"Id of last superlayer to be displayed" D=0
CHOPT	C	"Options" D='␣'

Display range of superlayers. Options:

```
-A : add to existing plot (no workstation clear)
-B : B0 daughters only
-C : display isochrones for drift devices
-F : display found tracks only
-G : display GEANT hits (Monte Carlo Impact Points)
-I : plot and print plane info, pause after each plane
-L : wire addresses are listed while drawn
-M : draw the Monte-Carlo track
-N : display number of MC particle on the right side of the hit
-P : list MC information for each hit
-R : display reference tracks only
-T : draw the reconstructed track
-U : show only ghost tracks (with -T)
-V : show vertices
-W : draw the active wires
-O : show only track reconstructed in 0-deg planes
-Y : show only track reconstructed in stereo planes
-Z : use polyline (Zigzag) rather than smooth curve for MC track
```

5.1.3 PLANE ipl [chopt]

`IPL` I "Id of plane" `D=28`
`CHOPT` C "Options" `D='␣'`
Display plane IPL. Options:

```
-A : add to existing plot (no workstation clear)
-B : B0 daughters only
-C : display isochrones for drift devices
-F : display found tracks only
-G : display GEANT hits (Monte Carlo Impact Points)
-I : plot and print plane info, pause after each plane
-L : wire addresses are listed while drawn
-M : draw the Monte-Carlo track
-N : display number of MC particle on the right side of the hit
-P : list MC information for each hit
-R : display reference tracks only
-T : draw the reconstructed track
-U : show only ghost tracks (with -T)
-W : draw the active wires
-0 : show only track reconstructed in 0-deg planes
-Y : show only track reconstructed in stereo planes
-Z : use polyline (Zigzag) rather than smooth curve for MC track
```

5.1.4 DRAW

(Re)draw the detector using the latest (or default) set of redraw parameters.

5.1.5 STYLE [mtmimp rtrhit cross opaque header frame axis type stereo background mc b0 rc mimps quant wire aspect wall sens mver plan sclip neut chain ever magnet rver spline rseg cell target widmc widrc chopt]

MT*MIMP	C	"use mimps to draw MC tracks (y/n) ?" D='␣'
RT*RHIT	C	"use rhits to draw RC tracks (y/n) ?" D='␣'
CR*OSS	C	"draw crosses at rhit position (y/n) ?" D='␣'
O*PAQUE	C	"draw opaque detector planes (y/n) ?" D='␣'
H*EADER	C	"draw header (y/n) ?" D='␣'
F*RAME	C	"draw frame (y/n) ?" D='␣'
AX*IS	C	"draw axes (y/n) ?" D='␣'
T*YPE	C	"draw particle type string (y/n) ?" D='␣'
ST*EREO	C	"show stereo plane (y/n) ?" D='␣'
BA*CKGROUND	C	"use background colour (y/n) ?" D='␣'
MC	C	"show MC tracks (y/n) ?" D='␣'
B0	C	"restrict to golden B0 daughters (y/n) ?" D='␣'
RC	C	"show RC tracks (y/n) ?" D='␣'
MI*MPS	C	"show MIMPS (y/n) ?" D='␣'
QU*ANT	C	"show QUANTITY (y/n) ?" D='␣'
WI*RE	C	"show wires/hits (y/n) ?" D='␣'
AS*PECT	C	"force conserved aspect ratio (y/n) ?" D='␣'
WA*LL	C	"draw walls (y/n) ?" D='␣'
SE*NS	C	"use sensitive areas for drawing planes (y/n) ?" D='␣'
MV*ER	C	"draw MC vertices (y/n) ?" D='␣'
PL*AN	C	"draw planes (y/n) ?" D='␣'
SC*LIP	C	"apply software clipping (y/n) ?" D='␣'
NE*UT	C	"show neutral MC particles (MC must be active) (y/n) ?" D='␣'
CH*AIN	C	"show whole decay chain (y/n) ?" D='␣'
EV*ER	C	"show end vertices (y/n) ?" D='␣'
MAG*NET	C	"show magnet (y/n) ?" D='␣'
RV*ER	C	"show reconstructed vertices (y/n) ?" D='␣'
SPL*INE	C	"use spline to display tracks in magnet (y/n) ?" D='␣'
RS*EG	C	"show reconstructed segments (y/n) ?" D='␣'
CE*LL	C	"show detector cells (y/n) ?" D='␣'
TA*RGET	C	"show target (y/n) ?" D='␣'
WIDMC	R	"width factor of MC lines" D=0.
WIDRC	R	"width factor of RC lines" D=0.
CHOPT	C	"Options" D='␣'

Set parameters which influence the drawing style. Option L displays the current values. Options:

```
-L : display current option values
```

5.1.6 SCOPE [scope1 scope2 scope3 scope4 scope5 scope6 scope7 scope8 scope9 scope10 scope11 scope12 scope13 scope14 scope15 chopt]

SCOPE1...15	C	"Scope descriptor" D='␣'
CHOPT	C	"Options" D='␣'

Define the display scope through descriptors for superlayer names. Option 'L' just lists the current scope, ignoring all other arguments. The scope arguments may be superlayer names - including a trailing asterisk as placeholder - or a predefined group name. Examples:

```
display/scope TC*        - display OT trigger chambers only
display/scope SI08 SI09  - display some magnet silicons only
display/scope MX*        - display muon pixel chabmers only
display/scope SIVD IMAG  - display vertex detector and inner
                           magnet tracker
display/scope MS03 -O    - remove MS03 from present scope
```

Valid group names:

```
SIVD  - all silicons with z<200cm
MAIN  - all silicon, MSGC and HDC trackers with z>200cm
FULL  - all silicon, MSGC and HDC trackers
IMAIN - all inner main trackers
OMAIN - all outer main trackers
MAG   - all magnet trackers
IMAG  - all inner magnet trackers
OMAG  - all outer magnet trackers
PAT   - all pattern trackers
IPAT  - all inner pattern trackers
OPAT  - all outer pattern trackers
TRIG  - all trigger trackers
ITRIG - all inner trigger trackers
OTRIG - all outer trigger trackers
ECAL  - calorimeter
MUON  - all muon chambers
PT    - all high-pt trigger chambers
```

Options:

```
-I  : add items to present scope
-O  : remove items from present scope
-L  : list present scope
```

5.1.7 PARAMETERS [mc rc cmp layer sector topp ctopp chopt]

`MC`	I	"Restrict to MC track " `D=-1`
`RC`	I	"Restrict to RC track " `D=-1`
`CMP`	I	"Restrict to component " `D=-1`
`LAYER`	I	"Restrict to layer" `D=-1`
`SECTOR`	I	"Restrict to sector" `D=-1`
`TOPP`	I	"Restrict to particle type" `D=0`
`CTOPP`	C	"Restrict no particle name" `D='␣'`
`CHOPT`	C	"Options" `D='␣'`

Define parameters for subsequent display/draw commands. Option -R resets all parameters

5.1.8 FRAME [xleft xright ylow yhigh chopt]

`XLEFT`	R	"Left x bound of frame " `D=0.0`
`XRIGHT`	R	"Right x bound of frame " `D=0.0`
`YLOW`	R	"Lower y bound of frame " `D=0.0`
`YHIGH`	R	"Higher y bound of frame " `D=0.0`
`CHOPT`	C	"Options" `D='␣'`

Define new frame via explicit world coordinates, or via rubberband. Option 'R' resets to original frame. Options:

```
-R : reset to standard frame. The standard frame is determined from
     the detector dimensions in x,y,z projected into the current
     view
-I : Zoom in (2x)
-O : Zoom out (2x)
-N : No automatic redraw
```

5.1.9 ANGLES theta phi [chopt]

`THETA`	R	"Polar angle of observers view" `D=70. R=-180.:180.`
`PHI`	R	"Azimuth angle of observers view" `D=0. R=-360.:360.`
`CHOPT`	C	"Options" `D='␣'`

Define the 3D viewing angles used for view 3. Options:

```
-N : No automatic redraw
```

5.1.10 SCALE [fscalx fscaly fscalz]

`FSCALX`	R	"x axis scale factor" `D=1.`
`FSCALY`	R	"y axis scale factor" `D=1.`
`FSCALZ`	R	"z axis scale factor" `D=1.`

Define additional scaling factors for each axis.

5.1.11 VIEW [iview chopt]

`IVIEW` I "Identifier of view selection" `D=-1`
`CHOPT` C "Options" `D='␣'`
Select the VIEW identifier, which determines the detector projection into the twodimensional plane. Possible values:

```
IVIEW = 0      xy projection
        1      xz
        2      yz
        3      free 3D   (see ANGLES command)
        4      isometric projection
        5      dimetric
        6      trimetric
        7      u
        8      v
        9      cone-corrected
       10      cone-corrected, pseudo-logarithmic x-scale
```

Any other integer keeps the view unchanged. Options:

```
-F : view is directed forward in z (relevant for hidden lines removal)
-B : view is directed backward in z (standard)
-N : No automatic redraw
```

5.1.12 HEADER

Draw prism header & edge on top of current picture.

5.1.13 CLIP iclip

`ICLIP` I "Clipping mode" `D=1 R=0,1`
See HIGZ manual under ISCLIP

5.1.14 SELNT nt

`NT` I "Normalization transform id" `D=1`
See HIGZ manual under ISELNT

5.1.15 COLOR cr cg cb

`CR` R "Red volume" `D=1.0`
`CG` R "Green volume" `D=1.0`
`CB` R "Blue volume" `D=1.0`
Test a color mixture in rgb space.

5.1.16 SETCOL [ic cr cg cb chopt]

IC	I	"Color index" D=15
CR	R	"Red volume" D=1.0
CG	R	"Green volume" D=1.0
CB	R	"Blue volume" D=1.0
CHOPT	C	"Options" D='␣'

Add/change colour mixture. Option -L displays colors defined. Note: Use the ASSIGNCOL command without parameters to see which colour indices are already used.

5.1.17 ASSIGNCOL object color

OBJECT	C	"Object to be assigned" D='␣'
COLOR	I	"Color index" D=-1

Assign color index to object name. If the color index is not predefined it must have been generated from rgb code by the SETCOL command. If no OBJECT argument is given, the current assignments will be listed.

5.1.18 PICK pickitem

PICKITEM	C	"Kind of item to be picked" D='␣'

Pick graphical objects in display window. Possible pick items are:

```
MC    - pick Monte Carlo track
RC    - pick reconstructed track
MImp  - pick Monte Carlo impact point
Hit   - pick detector hit
Plane - pick detector plane
MV    - pick Monte Carlo vertex   (planned)
RV    - pick reconstructed vertex (planned)
WAll  - pick wall                 (planned)
```

5.1.19 DQUERY

Query status of graphical workstations

5.1.20 SAVE kuname [kudir]

KUNAME	C	"Name of KUMAC file where parameters should be saved " D='dummy.kumac'
KUDIR	C	"Directory in which to store KUMAC file" D='␣'

Store graphics constants (frame etc) to a KUIP macro

5.2 LIST

Print information stored in detector banks

5.2.1 HITB [hitb1 hitb2 gede gesl mimp mtra rtra rhit chopt]

HITB1	I	"Id of first hit" D=0
HITB2	I	"Id of last hit" D=0
GEDE	I	"Associated plane" D=0
GESL	I	"Associated superlayer" D=0
MIMP	I	"Associated mimp" D=0
MTRA	I	"Associated MC track" D=0
RTRA	I	"Associated reconstructed track" D=0
RHIT	I	"Associated reconstructed hit" D=0
CHOPT	C	"Options"

Print hit information (from HITB table). When the number of an MC, mimp, reconstructed track, reconstructed hit, plane or superlayer object is specified, only hits associated with this object will be printed. Options:

```
no options defined yet
```

5.2.2 GEDE [gede1 gede2 gesl mimp hitb chopt]

GEDE1	I	"Id of first plane number" D=0
GEDE2	I	"Id of last plane number" D=0
GESL	I	"Superlayer" D=0
MIMP	I	"Associated mimp" D=0
HITB	I	"Associated hit" D=0
CHOPT	C	"Options" D='␣'

Print plane information (from GEDE table). When the number of an associated superlayer, mimp, hit object is specified, only planes associated with this object will be printed. Options: if no options are given, lengthy geometry information is suppressed. This information can be retrieved via options

```
-B : print border information
-G : print GEPT entries
-R : print rotation matrix & stereo angles
-W : print wire geometry information
```

5.2.3 GWAL [gwal1 gwal2 chopt]

GWAL1	I	"Id of first wall" D=0
GWAL2	I	"Id of last wall" D=0
CHOPT	C	"Options" D='␣'

Print wall information (from GWAL table).

5.2.4 GESL [gesl1 gesl2]

`GESL1` I "Id of first superlayer" `D=0`
`GESL2` I "Id of last superlayer" `D=0`
Print superlayer information on superlayer range.

5.2.5 MTRA [id1 id2]

`ID1` I "Id of first MC particle" `D=0`
`ID2` I "Id of last MC particle" `D=0`
Print MC particles

5.2.6 MVER [id1 id2]

`ID1` I "Id of first MC vertex" `D=0`
`ID2` I "Id of last MC vertex" `D=0`
Print MC vertices

5.2.7 RTRA [id1 id2 mtra chopt]

`ID1` I "Id of first track" `D=0`
`ID2` I "Id of last track" `D=0`
`MTRA` I "Associated MC particle" `D=0`
`CHOPT` C "Options" `D='␣'`
Print reconstructed tracks Options:

```
-C : in addition print covariance matrix from track fit
-M : show matching information from closest mimp
```

5.2.8 RSEG [id1 id2 mtra chopt]

`ID1` I "Id of first reconstructed segment" `D=0`
`ID2` I "Id of last reconstructed segment" `D=0`
`MTRA` I "Associated MC particle" `D=0`
`CHOPT` C "Options" `D='␣'`
Print reconstructed tracks Options:

```
-L : show track parameters of last instead of first point
-C : in addition print covariance matrix from track fit
```

5.2.9 RHIT [rhit1 rhit2 rtra rseg hitb gede gesl chopt]

RHIT1	I	"Id of first reconstructed hit"	D=0
RHIT2	I	"Id of last reconstructed hit"	D=0
RTRA	I	"Associated reconstructed track"	D=0
RSEG	I	"Associated reconstructed segment"	D=0
HITB	I	"Associated hit (from hitb bank)"	D=0
GEDE	I	"Associated plane"	D=0
GESL	I	"Associated superlayer"	D=0
CHOPT	C	"Options"	D='␣'

Print reconstructed hit coordinates When the number of a reconstructed track, hit, plane or superlayer object is specified, only reconstructed hits associated with this object will be printed. Options:

```
no options defined yet
```

5.2.10 MIMP [mimp1 mimp2 mtra rtra hitb gede gesl chopt]

MIMP1	I	"Id of first mimp"	D=0
MIMP2	I	"Id of last mimp"	D=0
MTRA	I	"Associated MC particle"	D=0
RTRA	I	"Associated reconstructed track"	D=0
HITB	I	"Associated hit"	D=0
GEDE	I	"Associated plane"	D=0
GESL	I	"Associated Superlayer"	D=0
CHOPT	C	"Options"	D='␣'

Print MIMP (Monte Carlo impact point) information on detector range. When the number of an MC, hit, plane or superlayer object is specified, only mimps associated with this object will be printed. Options:

```
no options defined yet
```

5.2.11 TOPP [topp1 topp2]

TOPP1	I	"Id of first particle"	D=0
TOPP2	I	"Id of last particle"	D=0

Print TOPP (Table of particles) entries.

5.2.12 DIGB [digb1 digb2]

DIGB1	I	"Id of first DIGB digit"	D=0
DIGB2	I	"Id of last DIGB digit"	D=0

Print rows from DIGB digit table

5.2.13 HITC [hitc1 hitc2 gcal]

`HITC1` I "Id of first HITC digit" `D=0`
`HITC2` I "Id of last HITC digit" `D=0`
`GCAL` I "Id of ECAL block number" `D=0`
Print rows from HITC (ecal hit) table

5.2.14 MCAL [mcal1 mcal2 gcal]

`MCAL1` I "Id of first MCAL digit" `D=0`
`MCAL2` I "Id of last MCAL digit" `D=0`
`GCAL` I "Id of ECAL block number" `D=0`
Print rows from MCAL (ecal hit) table

5.2.15 GCAL [gcal1 gcal2 chopt]

`GCAL1` I "Id of first GCAL object" `D=0`
`GCAL2` I "Id of last GCAL object" `D=0`
`CHOPT` C "Options" `D='␣'`
Print rows from GCAL (ecal blocks) table Options: if no options are given, lengthy geometry information is

```
            suppressed. This information can be retrieved via options
      -B : print border information
      -G : print GEPC entries
```

5.2.16 CHAIN mtra

`MTRA` I "Id of MC particle" `D=0`
Print decay chain for MC particle

Appendix A

Coordinate Systems

The coordinate systems described here are defined externally in the framework of *Arte* and the HERA-B geometry description. Since knowledge of the coordinate systems is essential for understanding many of the things *prism* is doing, we will review here the most important conventions.

A.1 The *Arte* coordinate system (ACS)

This is the coordinate system in which the absolute positions of superlayers, layers etc in the geometry tables as well as spatial Monte Carlo information are given. These are also the coordinates which are drawn on the axes in part of the views. The origin is located near the target position, the z axis points in the direction of the proton beam, the x axis is oriented normal to z in the horizontal plane, and the y axis is directed upwards such that (x, y, z) form a right-handed system.

The exact definition of the *Arte* coordinate system is of no significance to *prism* . Only the relations to the other coordinate systems are relevant.

A.2 The plane coordinate system (PCS)

Each detector plane from the GEDE table and each wall from the GWAL table may be rotated around arbitrary angles. Their geometry is defined in the *plane coordinate system* (x_P,y_P,z_P). The origin of this coordinate system is identical to that of the *Arte* coordinate system. Figure A.1 shows the case where the plane coordinate system is rotated around the z axis by an angle ρ. In this case, the relation is

$$x_P = x \cos \rho - y \sin \rho$$

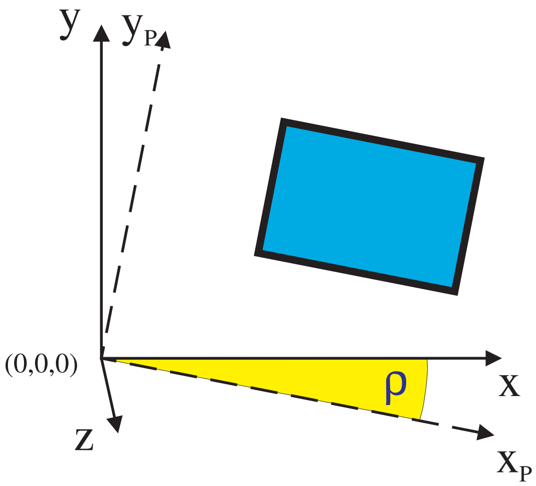


Figure A.1: Rotated plane coordinate system

$$y_P = x \sin\rho + y \cos\rho$$
$$z_P = z$$

where $\cos\rho$ = `fgede_rot(1,pl)` and $\sin\rho$ = `fgede_rot(2,pl)` if `pl` is the row number of the detector plane. In the case of a general rotation, the full rotation matrix must be applied:

$$\begin{pmatrix} x_P \\ y_P \\ z_P \end{pmatrix} = \begin{pmatrix} R_1 & R_4 & R_7 \\ R_2 & R_5 & R_8 \\ R_3 & R_6 & R_9 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

with R_i = `fgede_rot(i,pl)` in case of a detector plane and R_i = `fgwal_rot(i,pl)` for a wall. Note that this definition of the rotation matrix differs from the one used in GEANT.[1]

The plane coordinate system is used to define the borders of the planes and walls. Figure A.2 illustrates the definition of `xmin` and `xmax` attributes for a simple rectangle-shaped object (`shp=1`). If the plane is a sheared, epiped-like object (`shp=10`), the borders are defined as in fig. A.3.

A.3 The measurement coordinate system (MCS)

Since eg. silicon strips can be printed at an angle to the outside borders of a plane, the measurement coordinate system (x_m,y_m,z_m) is introduced to describe the information derived from the detector. The origin of this coordinate system

[1]In GEANT, the inverse matrix is used.

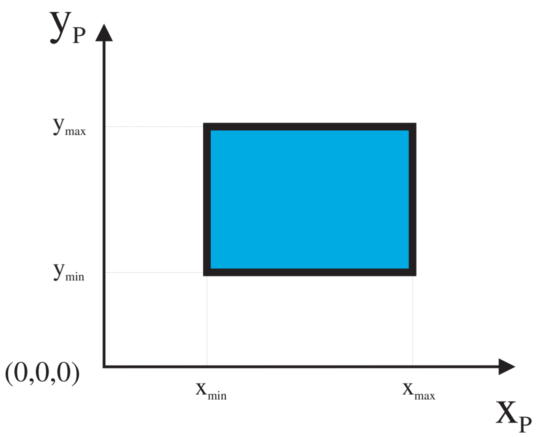


Figure A.2: Definition of borders for a `shp=1` plane or wall

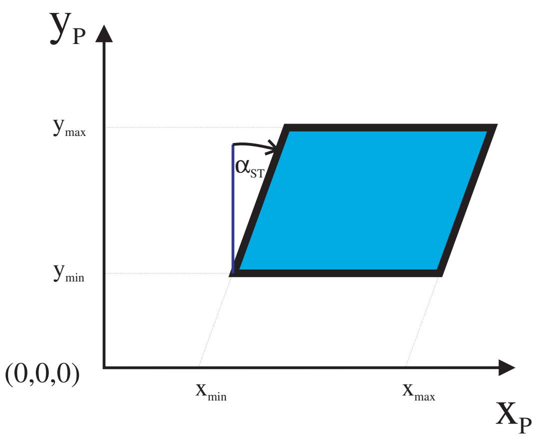


Figure A.3: Definition of borders for a `shp=10` plane or wall

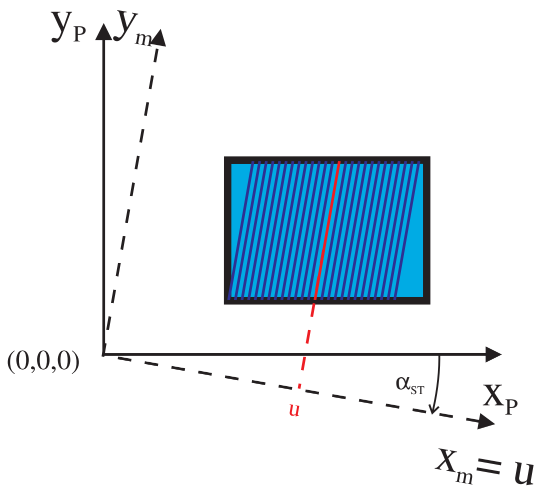


Figure A.4: Relation between measured coordinates and plane coordinates.

is identical to that of both the *Arte* and plane coordinate systems. The rotation is described by the intrinsic stereo angle α_{ST}. Figure A.4 shows the case for a strip device which measures only one coordinate u, which is identified with x_m. The relation is then

$$u = x_m = x_P \cos \alpha_{ST} - y_P \sin \alpha_{ST}$$

where $\cos \alpha_{ST} =$ `fgede_cos(pl)` and $\sin \alpha_{ST} =$ `fgede_sin(pl)` if `pl` is again the row number of the detector plane.

Appendix B

HIGZ Interface

B.1 HIGZ colours used by *prism*

Per default, the colour palette provided by the HIGZ package consists only of the basic colours of the *rgb* and *cmyk* colour models, which correspond to the HIGZ colour indices 0–7. All other colours needed by *prism* are assigned to the spare colour indices 8–255 with *rgb* descriptors. Inside *prism* , colours are addressed with an internal code of up to three letters. Table B.1 shows the assignment of colours inside *prism* .

B.2 Normalization transformations used by *prism*

The HIGZ package allows to define an arbitrary number of coordinate translations for an application. Table B.2 shows the normalization transformations used inside *prism* .

prism Colour	Object	HIGZ Index	r	g	b	redefine	default
loq	logo square	11	.5	1	.5	X	
ltv	logo triangle	5	1	1	0		X
lcv	logo circle volume	2	1	0	0		X
dis	display box volume	13	.9	.9	1	X	
		14	1	.9	1	X	
txt	text & axis	15	0	0	1	X	
mc	MC track	16	1	0	0	X	
mim	MIMP	1	0	0	0		X
rc	RC track	17	0	0	1	X	
cir	circle	18	0	1	0	X	
mav	magnet volume	19	1	.8	1	X	
mab	magnet border	20	1	0	1	X	
riv	RICH volume	21	1	1	.8	X	
rib	RICH border	22	1	1	0	X	
plv	plane volume	23	1	.855	.725	X	
plb	plane border	1	0	0	0		X
wir	wires & hits	24	0	.8	0	X	
wav	walls volume	5	1	1	0		X
wab	walls border	3	0	1	0		X
ca1	ECAL level 1	25	.9	1	.9	X	
ca2	ECAL level 2	26	.7	.8	.7	X	
ca3	ECAL level 3	3	0	1	0		X
ca4	ECAL level 4	5	1	1	0		X
ca5	ECAL level 5	2	1	0	0		X
bac	background	27	1	.9	.9	X	
rs	RC segments	28	0	.7	0	X	
cev	cell volume	29	.7	1	.7	X	
ceb	cell border	1	0	0	0		X
tar	target	4	0	0	1		X

Table B.1: Table of colours used by *prism* , and their default assignment

prism NT	HIGZ NT	coordinate system	borders
`nt_fram`	51	picture coordinates	picture rectangle
`nt_det`	52	world coordinates	picture rectangle without axis areas
`nt_pict`	53	world coordinates	picture rectangle including axis areas

Table B.2: Table of normalization transformations used by *prism* .

Appendix C

Versions History

A list of *prism* versions and the most important updates is given in table C.1.

Version	Changes	Date
1.xx	Some simple routines to display MC hits and tracks	Aug 93
2.xx	Views, drifttime isochrones	Dec 93
3.00/00	restructured as separate package	20/09/94
3.00/01	axis text, magnet fill	23/10/94
3.00/02	Improved init & track display	04/12/94
3.01/00	uses statement functions to access /zd/, /zh/, /zm/	14/01/95
3.01/01	before RS6000 implementation	21/01/95
3.01/02	version running on hp, sgi/c, rs6000	21/01/95
3.01/03	small corrections	28/01/95
3.01/04	view 9 (spreads SVD)	28/01/95
3.01/05	view 10 (spreads SVD & inner tracker)	28/01/95
4.00/00	now based on hbrcan/arte table structure	07/04/95
4.01/00	new definition of fhitb_x & stereo angle, (for Rcan/Arte 1.04)	21/04/95
4.02/00	many updates for arte embedded display, printing routines	12/07/95
4.03/00	vertex display (MC), new golden particles selection	03/08/95
4.03/01	improved color management, new command d/assign	04/08/95
4.03/02	include MC vertex when drawing MC tracks	23/11/95
4.03/03	supports cheap bw terminals	08/12/95
4.04/00	style, scope, parameter commands, NT-Handling, VP-Handling, introduction of Tcl/Tk interface picking for planes, MC tracks, MIMPs, HITBs	01/02/96
4.04/01	adjustments for SP2 and alpha	26/03/96
4.05/00	prism for Arte	15/04/96
4.05/01	software clipping in dimc	18/05/96
4.05/02	end vertex extended for charged particles	30/05/96
4.05/03	display decay chains	01/06/96
4.06/00	neutrals, decay chains	27/06/96
4.06/01	zarfun rearrangements for RS6000	28/06/96
4.06/04	cvs code management	24/10/96
4.07/01	catch missing tables, RVER display (I. Belyaev) spline parameter	28/11/96
4.07/02	Picking of reconstructed tracks	29/04/97
4.07/03	reconstructed segments, cells (A. Lanyov)	30/04/97

Table C.1: *prism* versions