The ALERT Detector in *ced*

David Heddle
Jefferson Lab and Christopher Newport University
heddle@jlab.org

2 January 2025

1 Introduction

As of *ced* version 1.70β2, there are nascent 2D and 3D views of the ALERT detector available in *ced*. At the moment *ced* will display adc hits from the drift chamber (AHDC) and time-of-flight (ATOF) hipo banks. It will also swim and display any Monte Carlo tracks found in the event. As soon as hipo files with reconstructed banks and tracks are available, we will modify *ced* to display them.

You can obtain the *ced* tarball here: ced1.70beta2 or, equivalently, here: `https://tinyurl.com/vwt2sdmn`

This is copying from my Google drive. There will be a scary warning that there was a problem with the preview (it's a tarball), but just click `Download` anyway. You can trust me.

If the download doesn't work, send me an email.

To run *ced* you must be online (to connect and get the geometry) and have a JAVA runtime of **`v17`** or later. After downloading and expanding the gzipped tarball, you should be able to double click the `ced.jar` file int the `cedbuild` directory. If that doesn't work, try cd'ing to the `cedbuild` directory and issuing the command `java -jar ced.jar`. If that doesn't work try running the `ced.sh` bash script. If that doesn't work, nothing will work. Send me an email. In the event *ced* crashes and burns at startup, the most likely culprit will be a too-old JAVA runtime. (You can enter `java -version` on a command line to check.) *ced* should work on MacOS, linux, and even the 75% of the world's laptops and desktops that we in CLAS sneer at (Windows).

Please report all bugs to `heddle@jlab.org` so that we can try to get the ALERT displays as bug-free as we can before the run starts.

1.1 Why a β version?

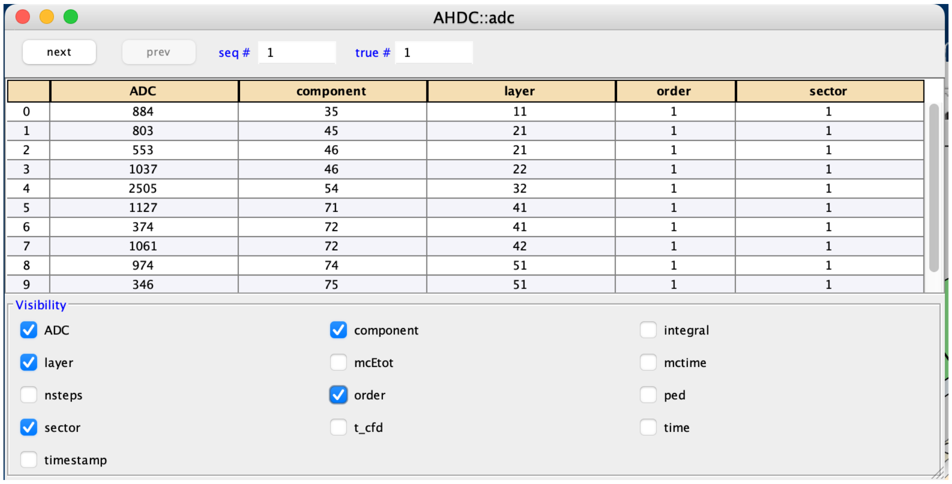
There are some outstanding issues, discussed below, with index mapping (numbering) between the ALERT hipo bank indices and the *ced* geometry service based convention. There also appears (to me) a systematic rotation of tracks when overlayed on the hits, as we'll see below.

2 Numbering

ced obtains its geometry from the geometry service in coatjava. Internally it uses, consistent with the geometry service, a 0-based indexing, but this is hidden from the user. The *ced* internal numbering convention is designed to mirror the 0-based indexing of the collections retrieved from the cotajava geometry service. Externally, in the actual *ced* views, a 1-based convention is used (for display purposes) to match more closely what is found in hipo banks. However, for some detectors, including ALERT, the mapping is not as simple as "just add or subtract 1".

2.1 Drift Chamber Numbering

Like other detectors in CLAS12, ALERT uses a 1-based indexing in its hipo banks. However, the difference goes beyond the index start; the geometry service references the ALERT drift chamber wires by the pseudo-4tuple $\{sector = 0,\ superlayer,\ layer,\ component$ (wire)$\}$. It is "pseudo" because the sector is always 0. The ALERT AHDC hipo bank references components by a different pseudo-4tuple $\{sector = 1,\ layer,\ order = 1,\ component$ (wire)$\}$. This is seen in the *ced* hipo bank display in Fig. 1.



	ADC	component	layer	order	sector
0	884	35	11	1	1
1	803	45	21	1	1
2	553	46	21	1	1
3	1037	46	22	1	1
4	2505	54	32	1	1
5	1127	71	41	1	1
6	374	72	41	1	1
7	1061	72	42	1	1
8	974	74	51	1	1
9	346	75	51	1	1

Figure 1: An AHDC hipo bank from a *ced* bank display showing the columns relevant for mapping to *ced* numbering.

A mapping is required between the ALERT hipo indexing (numbering) and *ced*'s geometry-service convention indexing. The only nontrivial mapping is that the `layer` in the ALERT hipo AHDC bank is a composite of *ced*'s `superlayer` and `layer`. The mapping is (using **integer** division and mod):

$$\text{superlayer (ced)} = \text{layer (ALERT)}/10 - 1 \tag{1}$$

$$\text{layer (ced)} = \text{layer (ALERT)} \ \% \ 10 - 1 \tag{2}$$

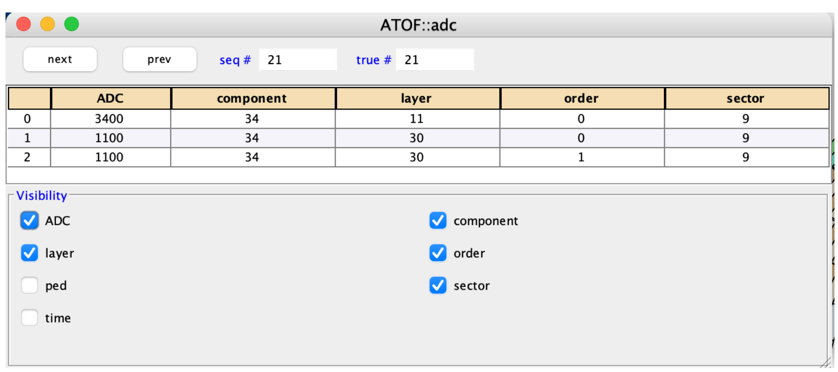
For example, layer = 51 in the hipo bank of Fig. 1 maps in ced to superlayer 4, layer 0.

This mapping was casually confirmed by the ALERT group. But if I still have it wrong, I need to know.

2.2 Time-Of-Flight (TOF) Numbering

The TOF numbering mapping between the ATOF hipo bank and *ced* used currently in *ced* is more problematic and we have NOT been able to get confirmation. The *ced* reference is the true 4tuple $\{sector,\ superlayer,\ layer,\ component$ (paddle)$\}$. Note that the component (paddle) has the range [0,3], that is, there are four paddle specifications in the geometry service for every $\{sector,\ superlayer,\ layer\}$ triplet.

The ALERT ATOF hipo bank references components by a different 4tuple $\{sector,\ layer,\ order = [0{,}1],\ component$ (paddle)$\}$. For ATOF the component (paddle) index does not reset with a new sector [1,15] index, instead it grows monotonically [1,60] as we go azimuthally around the z axis of the ALERT detector. This numbering is seen in the *ced* hipo display of an ATOF bank in Fig. 2.



	ADC	component	layer	order	sector
0	3400	34	11	0	9
1	1100	34	30	0	9
2	1100	34	30	1	9

Figure 2: An ATOF hipo bank from a *ced* bank display showing the columns relevant for mapping to *ced* numbering.

The mapping between ALERT ATOF banks and *ced*'s geometry-service convention indexing was guessed. We are currently using:

$$\text{sector (ced)} = \text{sector (ALERT)} - 1 \tag{3}$$

$$\text{superlayer (ced)} = \text{layer (ALERT)}/10 - 1 \tag{4}$$

$$\text{layer (ced)} = \text{layer (ALERT)} \ \% \ 10 \tag{5}$$

$$\text{component/paddle (ced)} = (\text{component/paddle (ALERT)} - 1) \ \% \ 4 \tag{6}$$

However, there is an edge case as seen in Fig. 2, namely the hipo ATOF `layer 30` maps to *ced* `superlayer 1 layer 9`, not 2 and -1. Note also that the component gets "mod-ed" down to [0,3]. Note that the (guessed) mapping does not use the `order` value.

As stated earlier, this mapping is a guess.

3 ALERT Views

As previously stated, there are two ALERT views, a 3D view and a (more useful) 2D view, named `ALERT XY`. As with other views, they are accessible with (in this case, toward the bottom) the `Views` menu of Fig. 3. You can also use the virtual desktop selector (with a double-click) at the top of the main *ced* window.

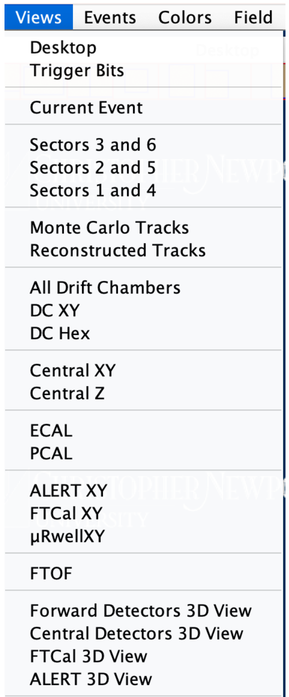


Figure 3: The *ced* `Views` menu.

3.1 The 3D View

The ALERT 3D view is shown in Fig. 4. It provides the common roller-ball type control when dragging the mouse in main view area. Different parts of the ALERT detector can be toggled on and off using the control panel. The (global) transparency of the components is controlled using the alpha slider. There is a movement up-down, left right panel. Zooming is done by a wheel motion, which has been greatly improved in this *ced* version. This is the same set of controls as the other 3D views in *ced*. There's not much to say about the 3D view. What you see is what you get.

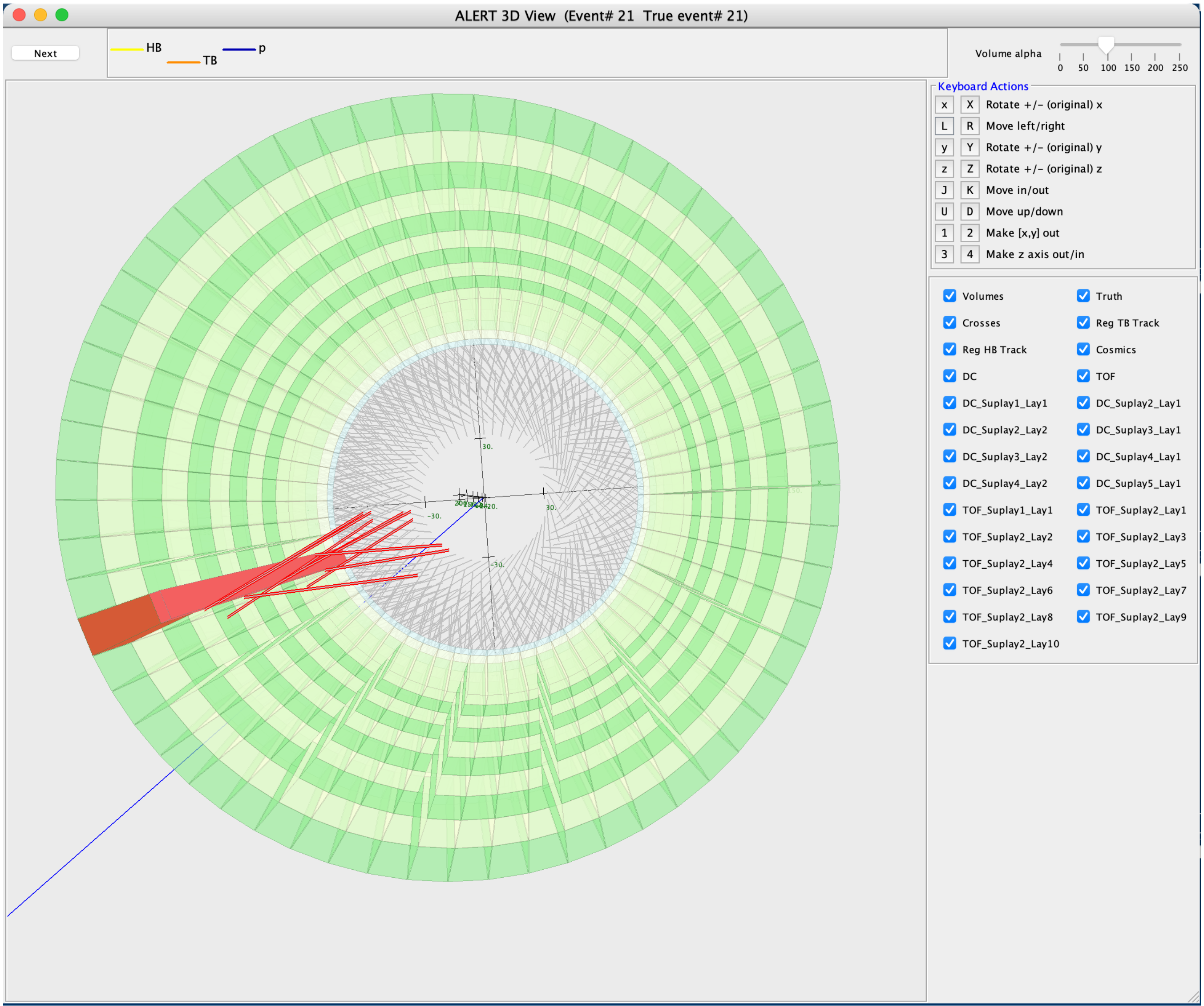


Figure 4: The *ced* 3D ALERT view.

3.2 The 2D View

We fully expect that, as usual, the 2D view will be more useful. The view at startup (and after loading an event) is shown in Fig. 5. The drift chamber wires are the ovals in the center area. The first wire (number 1) in each layer has a red outline instead of black, just as a reference. These can be seen near the bottom of the drift chamber area. The generally do not line up because of the stereo wire orientations. (They do line up when the view plane, as discussed below, is set to $z = 0$.) The TOF paddles are in the outer ring. (As discussed below, the TOF as shown in the Fig. 5 are not geometrically faithful.)

The ALERT 2D view `ALERT XY` has a major difference from the other 2D "XY" views in *ced* in that the view is from a position on the *positive* z axis rather than the negative. (Which is why the little xy axes in the lower left of the view (see Fig. 5) are flipped to the standard orientation as opposed to the other XY *ced* views) This orientation matches a picture of the ALERT TOF's we were referencing. Should it be required, this is easy to change.

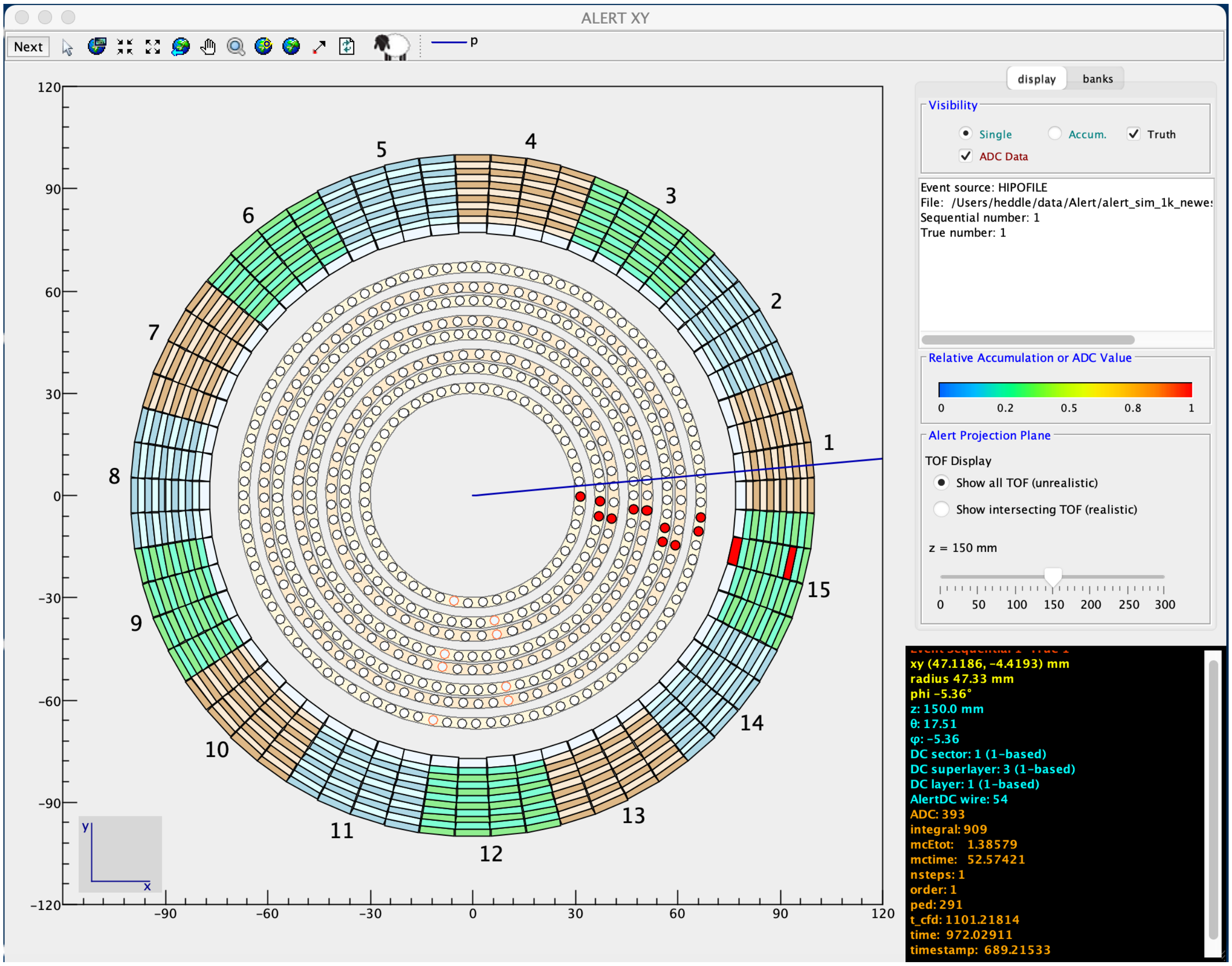


Figure 5: The *ced* The ALERT XY view at startup (except for showing an event.)

To the right of the detector display are two panels. The gray panel (including the white text area) with `display` and `banks` tabs at the top is the *control panel.* (Fig. 6)The black area is the mouse-over context feedback. Point at something on the detector active area and you'll get information on geometry and hits at the mouse location.

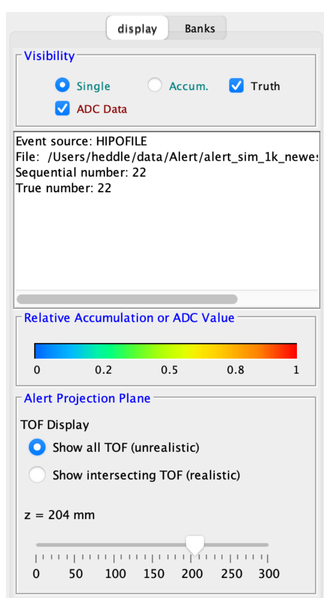


Figure 6: The control panel with the `display` tab selected.

3.2.1 The `display` tab

The drift chamber wires and the tracks (trajectories) are projected onto an adjustable z = constant plane. The user can adjust the plane, using the z-slider on the `display` tab, from $z = 0$ to $z = 300$mm. There is an (immovable) "camera" location used in the projection; it is placed at $(0, 0, 450)$ mm. As far as we know, this is as good as we can do in making the drift chamber display as geometrically faithful as possible in 2D.

The `Single` and `Accum.` radio buttons on the control panel toggle the viewing mode between "one event at a time" and accumulated mode. The normalized color scale is used to display accumulated data, like a visual 2D histogram, useful for finding holes or hotspots with real data. This is common for all *ced* views, so we will not discuss it further.

The section of the control panel labeled `TOF Display` allows you to toggle between the unrealistic default view of the TOF paddles shown in Fig. 5 and a more realistic rendering in Fig. 7 In Fig. 5, the unrealistic view, all paddles are shown at once, as if they are arranged radially, which they are not. This is done merely for the convenience of seeing all the data at once. In the realistic view of Fig. 7, only the layer (in the *ced* and geometry service definition of *layer*) of paddles that intersects the given constant-z plane are shown. So as you move the z slider you will see different layers. The colors will change indicating that you have move to the next (or previous) layer.

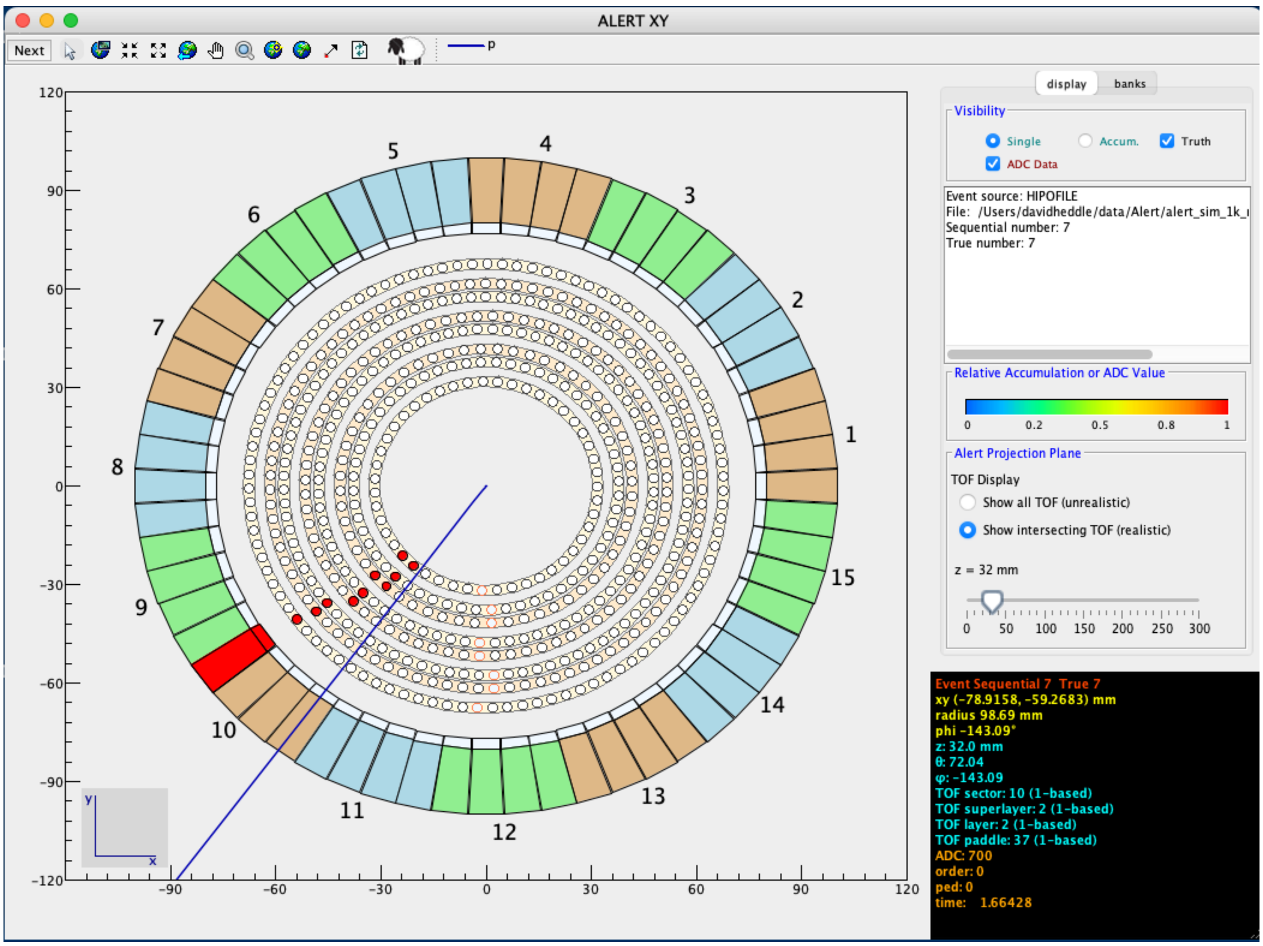


Figure 7: The *ced* The ALERT XY view with a more realistic (but less information content) view of the TOF. Only the layer whose paddles intersect the constant z-plane are shown.

3.2.2 The `banks` tab

Selecting the `Banks` (Fig. 8) tab gives access to the hipo bank displays (see Figs. 1 and 2.) The top panel (which is modifiable) shows the hipo banks of interest. The lower panel shows, in red, any of the "interesting" hipo banks that were found in the current displayed event. Double-clicking on the red label will display the hipo bank and its contents, as in Figs. 1 and 2. This functionality is redundant with more full-featured `Current Event` *ced* view, but it is convenient to have only banks of interest to a given view close-by.

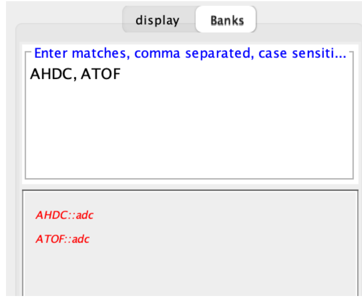


Figure 8: The `Banks` tab on the control panel

3.2.3 Bank Row Highlighting

Bank row highlighting was implemented for the ALERT XY 2D view. Bring up the ADDC and/or ATOF bank displays as discussed above. Any of the rows correspond to a component being filled in red (i.e., a “hit”) assuming the component is visible. If you click on a row in the bank display, the corresponding component on the view will be displayed in orange rather than red, as in Fog. 9

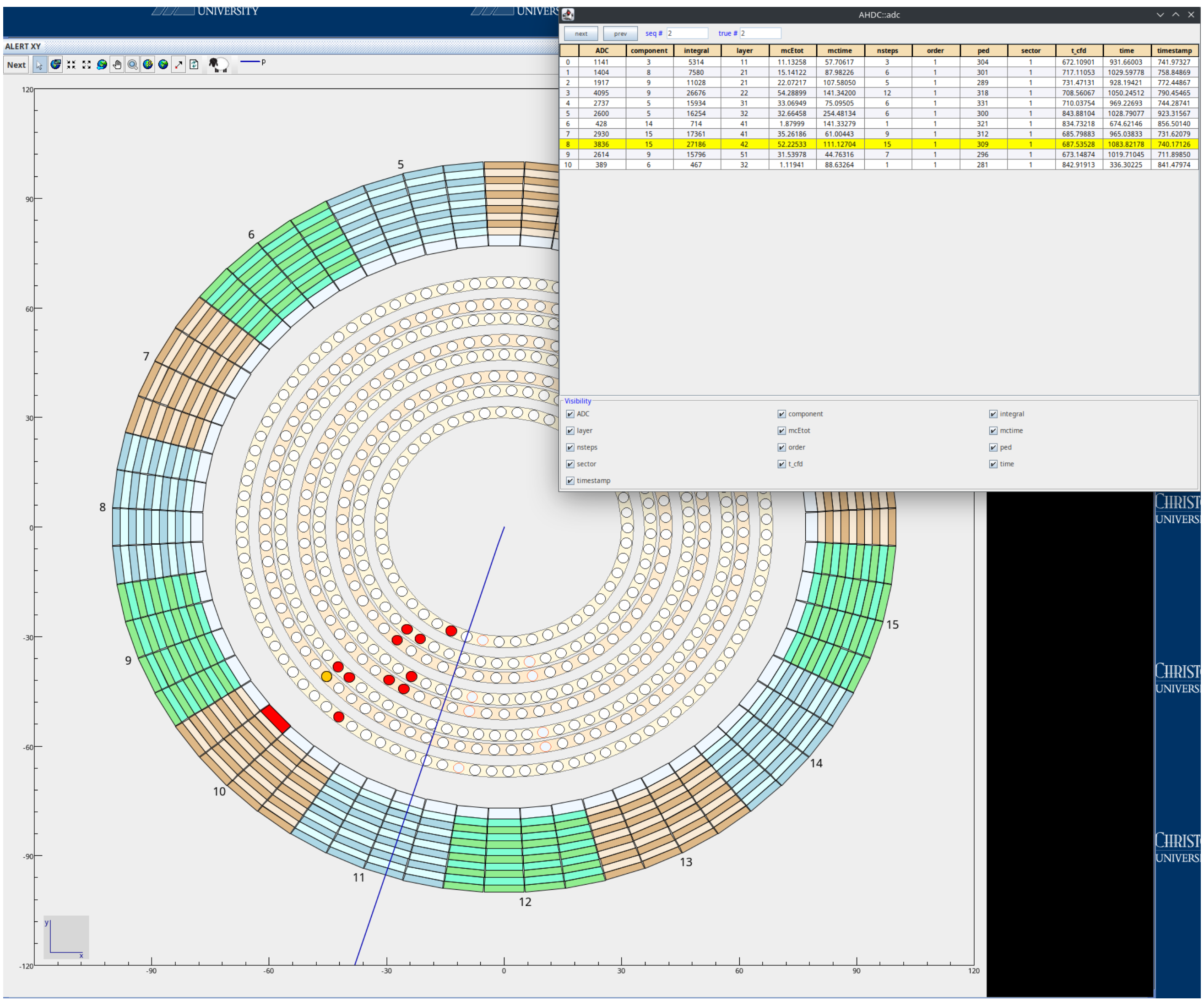


Figure 9: Clicking on a row in the AHDC or ATOF bank display results in the corresponding adc hit in the view to be filled in orange rather than red, as shown.

3.2.4 Context Mouse-Over Feedback

As with all 2D *ced* views, moving the mouse over the main part of the view results in context "feedback" in the black panel on the bottom right. This is visible in the figures. You can also hover the mouse over a trajectory and *ced* will display a little unobtrusive (perhaps too much so) "tool-tip" text pop-up that gives the track parameters (and possibly the source) that *ced* used to swim the track.

4 Trajectory Matching Problem

You will note from the 2D views that there is not good alignment between the straight tracks and the adc hits. It seems to be systematic: the track is approximately 15° to 20° counter clockwise of the best alignment of the hits that you can achieve visually by using the z-plane slider.