AliEVE

ALICE Event Visualization Environment

Matevž Tadel

Alja Mrak-Tadel

Contents

ALICE: basic user environment

ALICE + ROOT → AliROOT ♥

- Only external dependencies transport engines
- Runs on almost every os/arch (but windows)

Event data is stored in a set of ROOT files:

- split by detector and type; e.g. `TPC.Digits.root`
- **all** data stored in TTree's
- TClonesArray's used for logical grouping

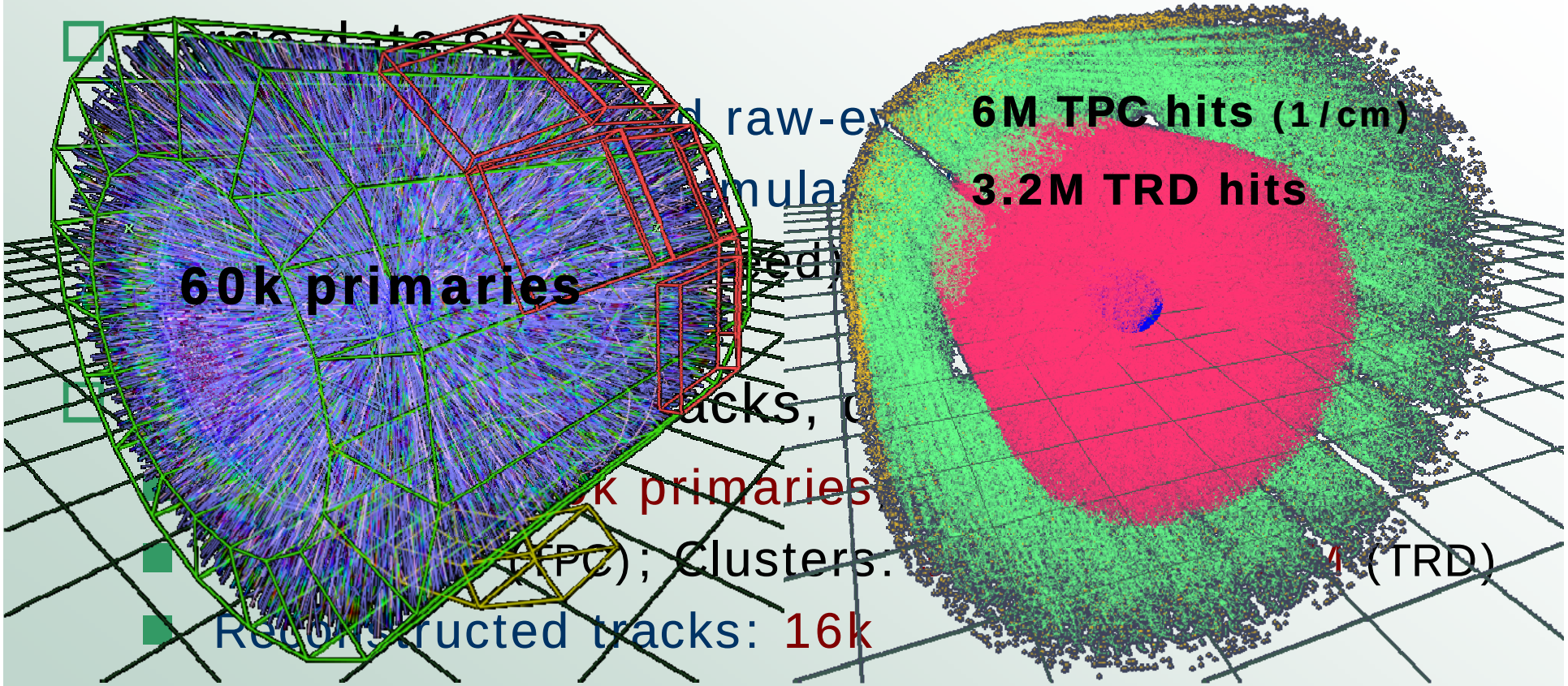
Used directly for data interaction/analysis:

- CINT scripts and ROOT prompt
- histograms, 2D-graphics, browser, context-menus

Users accustomed to this ...

ALICE central Pb-Pb event properties

- Image data size:
  - raw-ev
  - mula
  - ed
- acks, c
  - k primaries
  - (TPC); Clusters: 4 (TRD)
  - Reconstructed tracks: 16k



- It is not about showing everything …

Event Visualization in 3D

"Visualization of" and "GUI to":

- detector geometry
- event data:
  - simulation records: kinematics, hits, digits
  - raw data
  - reconstructed objects: clusters, tracks, V0's ...
  - physics objects: b-tags, Z^0/H-candidates, ...
- reconstruction & analysis algorithms
- calibration and alignment data

Used by:

- experts: visual debugging, development of algorithms
- non-experts: understand detector, event structure, ...
- presentations, demonstrations ... outreach

Too many elements, too many use-cases ...

Event Visualization Environment

Framework for specific event-displays.

Build on top of ROOT, same philosophy:

modular, loosely coupled class toolbox

- ROOT GUI – full toolkit, **interpreted!**

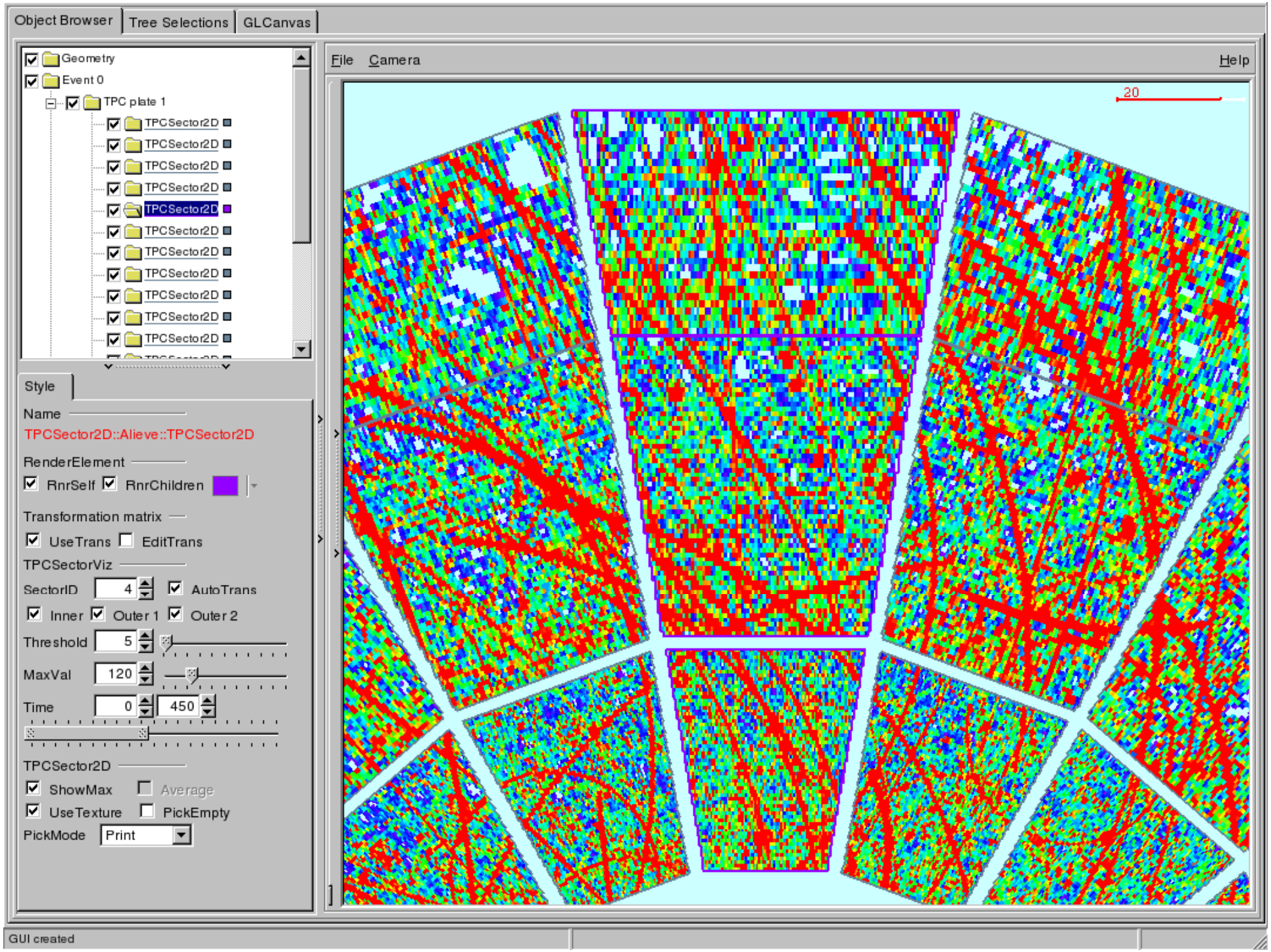
  GED – class editor: `TPCSector2D` → `TPCSector2DEditor`

- ROOT OpenGL – geometry + basic atoms

  Direct rendering: `TPCSector2D` → `TPCSector2DGL`

- Provide additional elements as needed
  - Start with basic low/medium-level elements

    A lot provided by ROOT anyway (browser/list-tree)
  - Build composite/top-level widgets from those
  - Concrete cases can be done w/ gui-builder/scripts


Object Browser
Tree Selections
GLCanvas
Geometry
Event 0
TPC plate 1
TPCSector2D
Style
Name
TPCSector2D::Alieve::TPCSector2D
RenderElement
RnrSelf
RnrChildren
Transformation matrix
UseTrans
EditTrans
TPCSectorViz
SectorID
4
AutoTrans
Inner
Outer 1
Outer 2
Threshold
5
MaxVal
120
Time
0
450
TPCSector2D
ShowMax
Average
UseTexture
PickEmpty
PickMode
Print
File
Camera
Help
20
GUI created

Meta-architecture

- Two main modules:
  1. Reve/ ROOT only
     1. Application core
     2. Framework base-classes for GUI and VIZ
     3. Basic visualization classes (points, tracks, ...)

     Can be compiled in stand-alone mode

     Will eventually become part of ROOT
  2. Alieve/ AliROOT
     1. Event loading / navigation
     2. Classes for detector-module representation (raw-data)
     3. CINT scripts perform most of the work
- Non-standard features – (Ali)ROOT standpoint

  namespaces, exceptions, minimal use of STL

Application core

1. Management of browsers & viewers

   Including redraw / refresh hooks

2. Registration of visualization objects

   a) Global storage: geometry, layout, ...

   b) Event storage: wiped on event change

3. Execution environment for CINT scripts

   Variables / functions to make scripts simple, e.g.:

   `Reve::AssertMacro(const char*)`

4. Event management & navigation

   Things to do/execute for each event

5. Task / thread management (data import/selection) – wait for CINT

Reve::RenderElement

Base-class for inclusion into object hierarchies:

1. List of children – graph traversal / rendering
2. List of browser / GUI representations – update!
3. List of parents – destruction!
4. Visibility of self / children
5. Main color pointer (show in browsers)

Not TObject derived!

- Has `virtual TObject* GetObject()`
- Use double inheritance or
- Use class `RenderElementObjectPtr`

  This can wrap any existing TObject.

Visualization classes – I.

Geometry:

- TGeo-interface classes
  a) Direct usage via TGeoPainter: requires geometry
  b) Extracted shape-data: standalone, fully configurable

Hits, clusters:

- PointSet : public TPolyMarker3D, extended
  Allow per-point TRef (optionally owned by the object)
  Direct-rendering, secondary-selection
  Special TSelector provided for filling – use Tree machinery
- PointSetArray – an array of point-sets
  Select on external criteria provided during filling
  Currently only 1D ... but we can do n-D!
  Interactive histogram

Visualization classes – II.

Trajectories, particles, tracks:

- Line : public PointSet
- Track : public Line

  Supports extrapolation in (const) magnetic field

  Field-maps needed (w/ material: fast-sim, vizualization)

  Can specify position/momentum at (interesting for kinematics):

  1. arbitrary reference points (enter/leave certain volume)
  2. daughter creation points / momentum
  3. decay
- TrackList – a collection of tracks

  Again, can provide external criteria for interactive selection: pT, chi2, ...

Visualization classes – III.

Digits, raw-data:

- QuadSet – set of rectangles, lines or hexagons
  Supports per quad TRef → secondary-selection
  Used for silicon detectors, TRD, TOF, PMD, ...
- BoxSet – set of boxes (calorimeters)
- TriangleSet – arbitrary triangle mesh

Support classes:

Instances shared among several modules

- RGBAPalette: map signal to colors + thresholds
- FrameBox: provide a uniform frame

Ref-counting (auto-deletion) & back-refs (update)

Reve as stand-alone environment

So we have many features ... we need to glue them.

Use CINT scripts for:

- steering of data extraction
- preparation of visualization objects
- creation of GUI elements (if object-editors are not enough)

VSD – Visualization Summary Data

- produce for selected events or obtain from `grid`
- extract data interesting for specific use

Just a tree (collection) in a ROOT file, pack with it:

1. Scripts to extract & visualize data

   Potentially one can include Reve as a compilable script (20k loc)
2. Scripts to create GUI
3. Detector geometry

These can be shipped and used w/ standard ROOT!

ALICE specific parts

1. Run / event loader

   Interface to geometry/mag-field/alignment/calibration data

2. Display of raw-data / digits

   Mostly use base-classes from Reve

   TPC visualization very specific ... 60MB event data!

3. VSD can be produced [minimal data classes part of Reve]

   Kinematics, hits, clusters, rec-tracks, V0s and kinks

   Not really used now ... prefer full-data for debugging.

   We expect this to change when data comes (windows)

4. Scripts

   Most have very little dependence on AliROOT

   Some of them quite elaborate (selection, object hierarchy)

   Can be potentially converted to compiled code/classes

Example script: silicon detector hits

```
Reve::PointSet*
its_hits(const char          *varexp    = "fX:fY:fZ",
         const char          *selection = "",
         Reve::RenderElement *cont      = 0)
{
  // GET THE TREE – ALICE SPECIFIC, BUT COULD AS WELL GET IT MYSELF!
  AliRunLoader* rl = Alieve::Event::AssertRunLoader();
  rl->LoadHits("ITS");
  TTree* ht = rl->GetTreeH("ITS", false);

  // CREATE VISUALIZATION OBJECT
  Reve::PointSet* points = new Reve::PointSet
    (Form("ITS Hits '%s'", selection));
  TPointSelector ps(ht, points, varexp, selection);
  ps.Select();

  // REGISTER THE RESULT
  gReve->AddRenderElement(cont, points);
  gReve->Redraw3D();

  return points;
}
```

Demo I: p-p@14TeV

1. Overall look
2. Geometry
   1. ITS from TGeo: render-modes, lights, clipping
   2. Reduced ALICE

      Explain render-element, editor, browser
3. Kinematics:
   1. Selection / picking, spawn editor, export to CINT
   2. Track-refs
   3. Choose-by-pT
   4. Import hits/clusters from context-menu
4. Next-event: geometry remains, event removed
5. TPC-hits
   1. by eta and by charge
   2. show histogram by charge from the file

Demo II: HIJING cocktail 1k primaries

1. ITS digits
   1. Show the 6 layers
   2. Select inner SSD ... show palette
      sec-selection
   3. Select inner SDD ... show how small it is
      sec-selection
2. ITS module stepper
   1. Zoom ... easier in 2D
   2. Scale
   3. Modes: average / occupancy
3. TPC digits preview

   Time, threshold/max-val, sec-selection

Demo III: TPC commissioning

Run 872_2

1. 2D view, gating pulse at low times

   Threshold, sec-selection, noise

2. 3D view, zoom in

   point-fractions and thresholds

3. Disable auto-pedestal, load event 6

   Show base-line – non-zero suppressed data

Demo IV: p-p@900GeV visual scanning

1. Primary-vertex
2. ESD track criteria
3. ITS hits
4. Select tracks as primaries
5. Top-level UI: event-nav, camera
6. Report histograms

Demo V: HIJING cocktail 10k primaries

To show something more hairy ...

- ITS hits
- TPC hits eta-split
- Kinematics (higher cutoff pT>500 MeV)
- ESD tracks w/ vertex cuts
- TPC digits
- Ludovic Gaudichet: V0 viewer

Conclusion

1. Minimal application core + base-classes

   Solve general problem in Reve ...

   and specific ones in Alieve.

2. Via scripts users gain full flexibility
   1. edit them for specific selections, packing of output
   2. extend them to include user's private data
   3. combine them together

   Allows fast development of specific display programs

3. Often data can be visualized directly from ROOT
   1. trees, scripts and other data can be packed together in a single ROOT file containing several events
   2. independent of experiment software

      laptops, unsupported platforms; use by universities, outreach