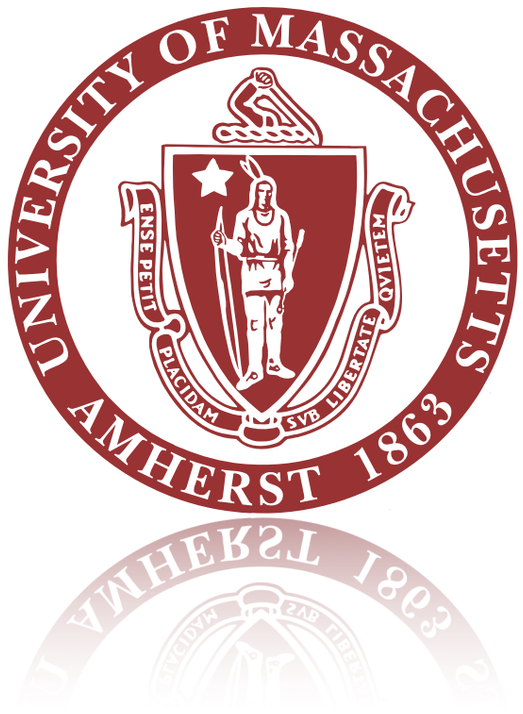


TADA + WebEventDisplay -

Edward Moyse

TADA

- TADA is the ATLAS Fast Physics Monitoring System
  - CHEP presentation 2016
- TADA has some very simple information about an event, extracted from the ‘tag’ (summary) file
  - Quantities, such as eta/phi of tracks, jets MET, and pT/ET etc are used to make plots which track physics performance ‘on-the-fly’
- Was asked to try to make a very simple 3D display to aid with the visualisation of these quantities
- Some links (unfortunately limited to ATLAS!):
  - https://atlas.web.cern.ch/Atlas/fastphys/tagmon/exo_multijet.html
  - Scroll to bottom and select ‘Display’

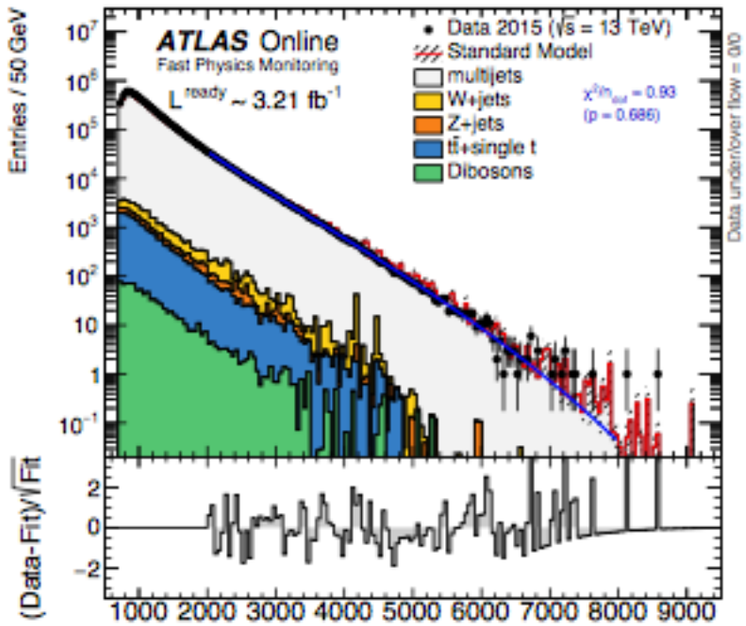


Figure 6: Di-jet invariant mass

SELECTED EVENTS

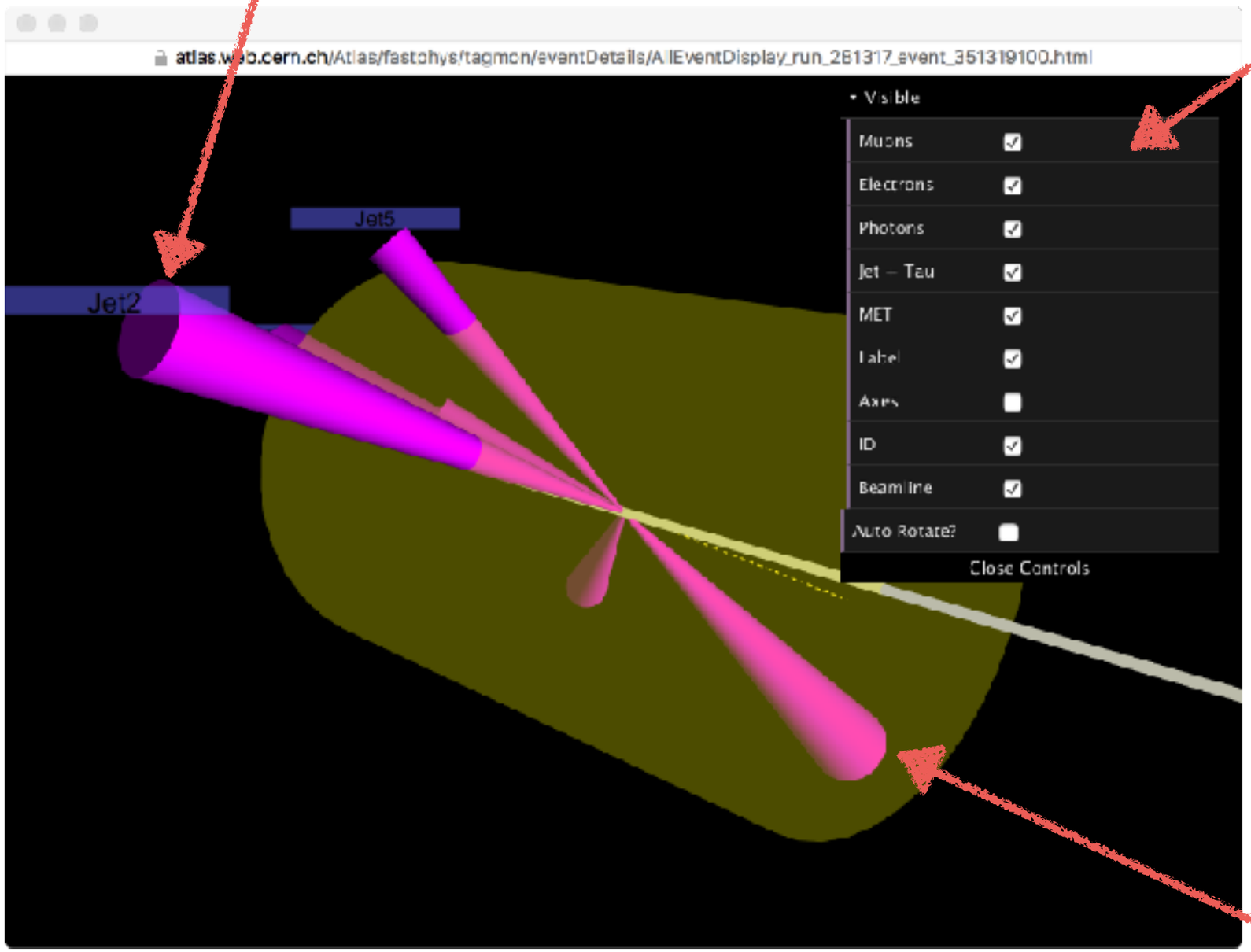
6 jets (tail)

RunNumber	EventNumber	LumiBlockN	NJet	JetPt5	JetPt6	Details	Display	TAG file
279169	1156465504	851	6	412723	178746	click	click	click
279932	987274395	706	6	401056	177351	click	click	click
279984	1079767163	677	6	420585	268564	click	click	click
280231	39964989	151	6	439350	126322	click	click	click
280231	113898628	184	6	420087	81391.4	click	click	click
280520	84518052	240	6	443761	51657.2	click	click	click
280862	683322481	301	6	400898	118825	click	click	click
280950	1353852600	521	6	433061	129448	click	click	click
281317	351319100	217	6	407030	176453	click	click	click
284213	2457796736	719	6	447879	81887.3	click	click	click
284285	3804162492	1219	6	409183	235544	click	click	click

Objects can be labelled

Simple menu

Allow user to switch off and on quantities, Enable/disable auto-rotate feature



Controls

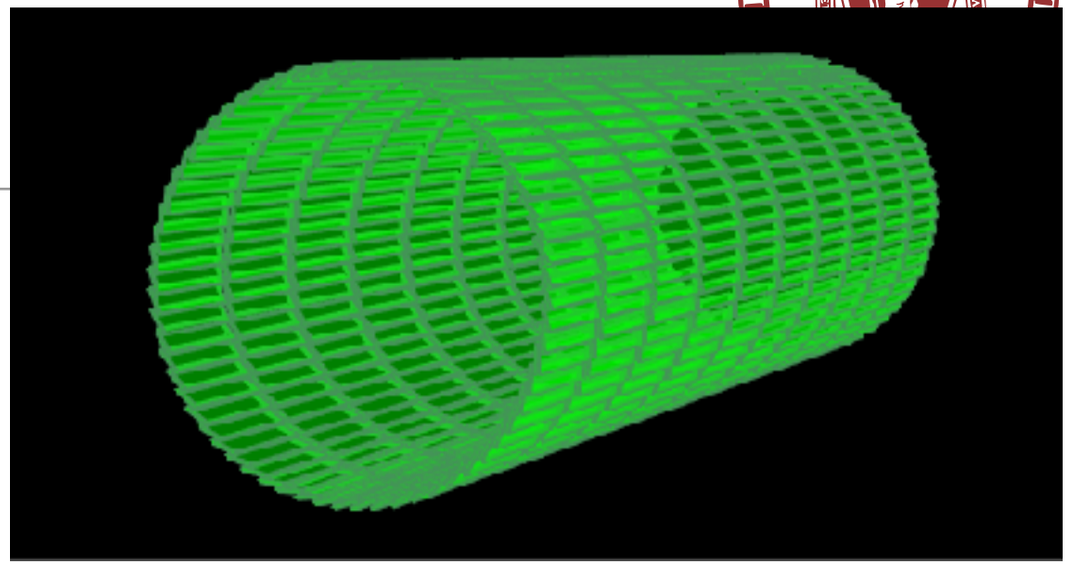
Mouse wheel - zoom in/out
Click and drag - rotates
Right-click and drag - pan
'H' - hides menu

VERY Simple representations of objects

ID volume shown, to give scale

Moving on from TADA

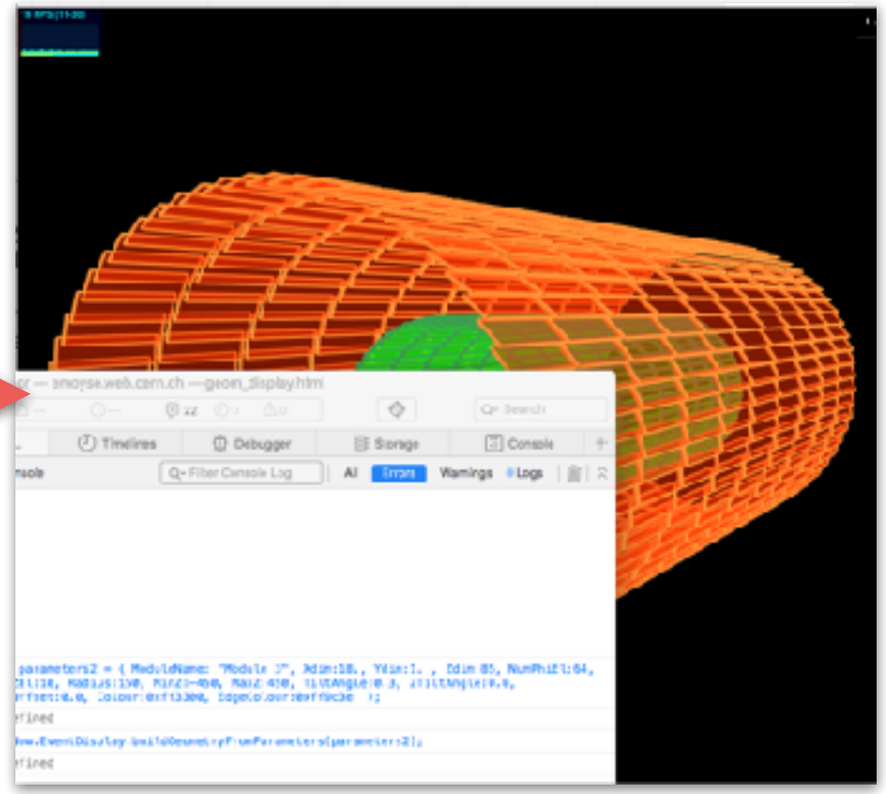
- Was then asked to expand on this, for visualising test geometries (e.g. for Kaggle challenge)
- So made an upgraded version of the viewer which can show ‘procedurally generated’ geometries
  - http://emoyse.web.cern.ch/emoyse/WebEventDisplay/geom_display.html
  - JS code is shown to the right
  - Relatively short and readable



```
  var parameters = { ModuleName: "Module 2", Xdim:10.,
Ydim:1. , Zdim:45, NumPhiEl:64, NumZEl:10, Radius:75,
MinZ:-250, MaxZ:250, TiltAngle:0.3, PhiOffset:0.0,
Colour:0x00ff00, EdgeColour:0x449458  };

window.EventDisplay.buildGeometryFromParameters(paramete
rs);
```

- **Interactivity**: If you have Safari / Chrome, you can open the developer tools and get a console & enter the following:

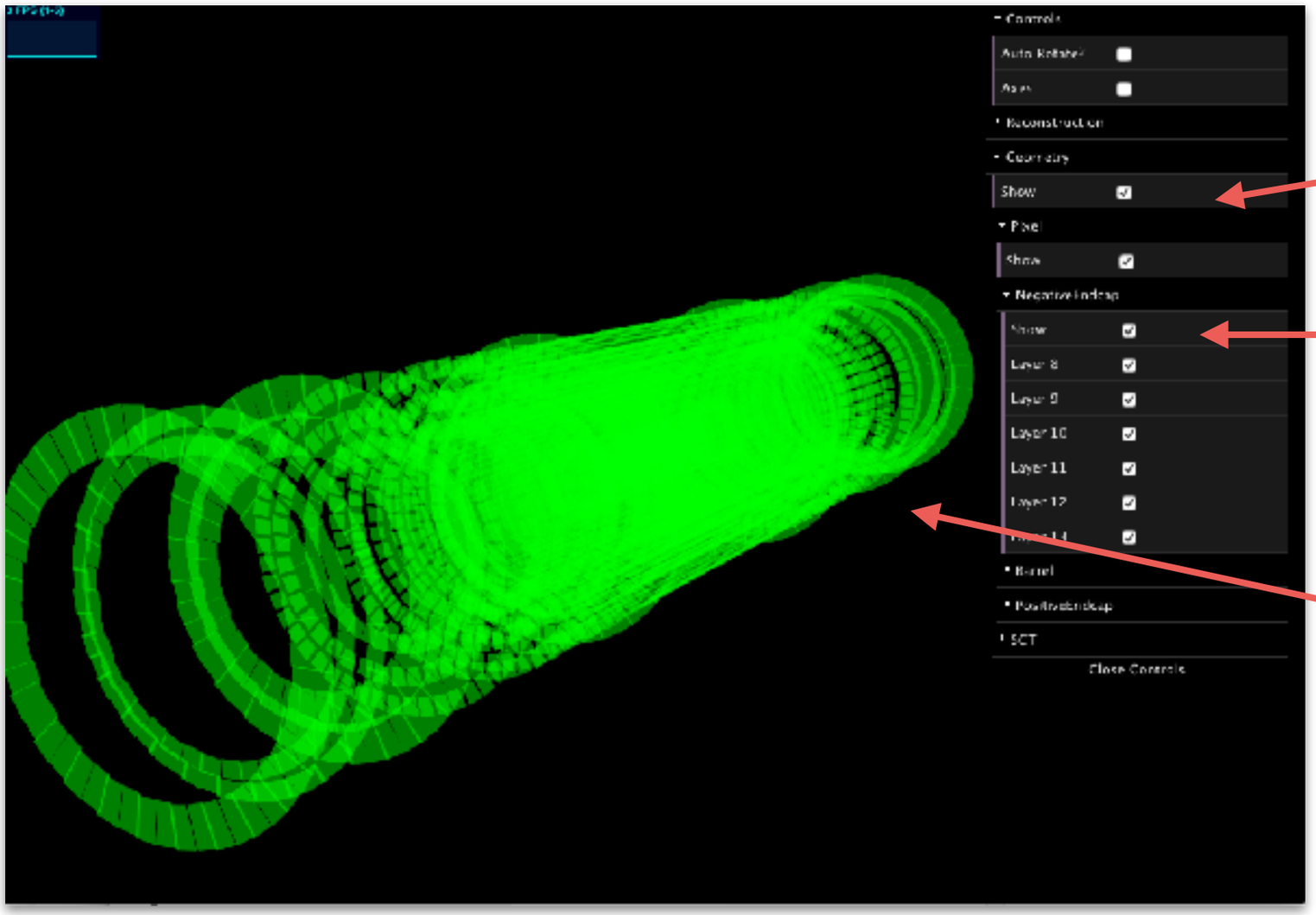
```
var parameters2 = { ModuleName: "Module 3", Xdim:18., Ydim:1. , Zdim:
85, NumPhiEl:64, NumZEl:10, Radius:150, MinZ:-450, MaxZ:450,
TiltAngle:0.3, ZTiltAngle:0.0, PhiOffset:0.0, Colour:0xff3300,
EdgeColour:0xff9c3e  };

window.EventDisplay.buildGeometryFromParameters(parameters2);
```



- Of course we could make menus to do all of this!

Visualising ‘real data’

- Within ATLAS we have some tools to dump tracking geometry to JSON
  - Format is still evolving and is currently a bit basic, but it is detector agnostic (and will stay so)



Menu auto-generated from JSON content

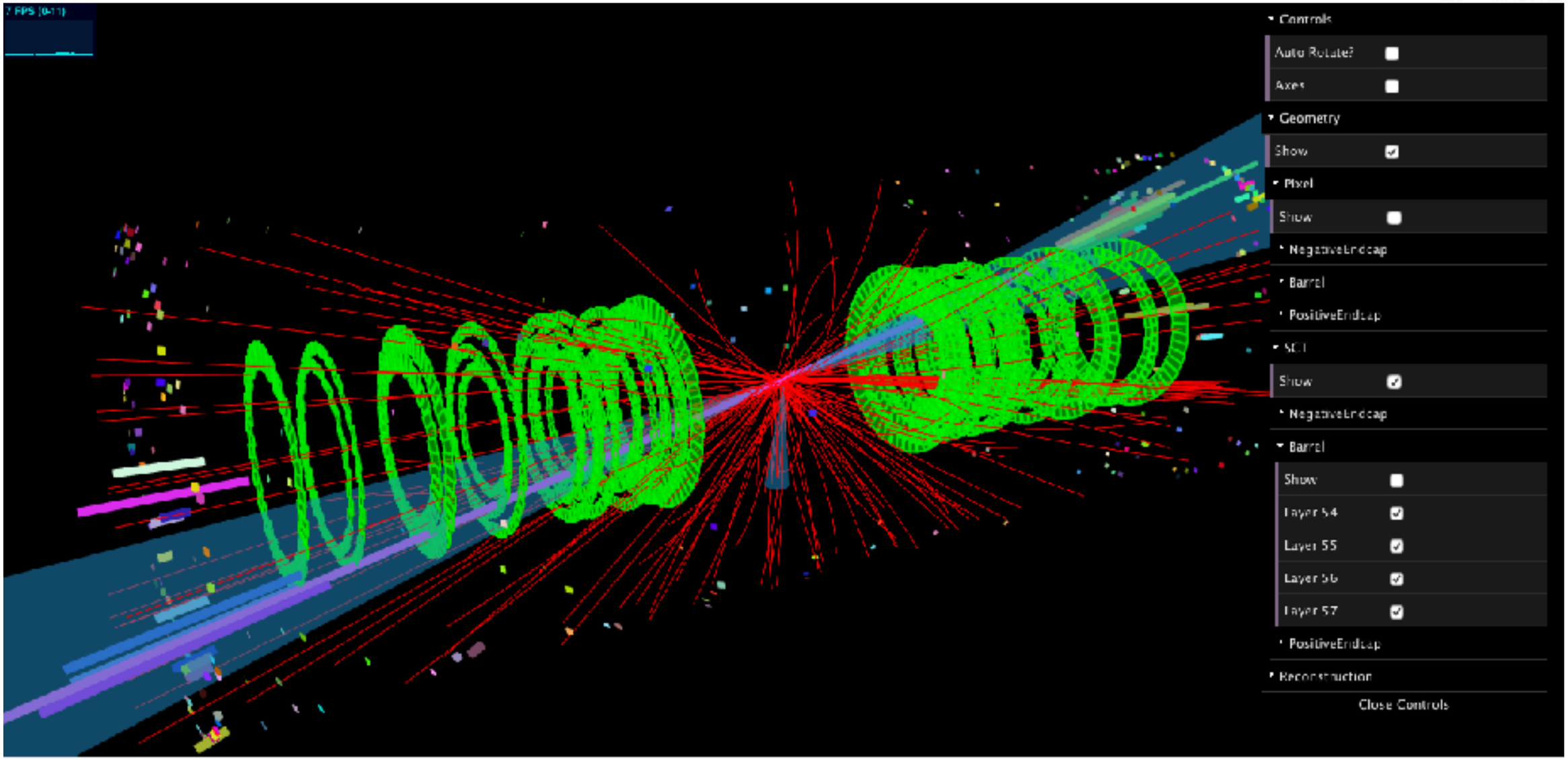
Can turn off/on components

Colour defined in the JSON

Visualising more complex i.e. event data

- Again: want to keep this as detector agnostic as possible
- Format is same JSON we sent to Ilija for initial use in **ATLASrift** (see other talk today)
  - Basically:
    - “OBJECTTYPE" : {“COLLECTION_NAME":{"OBJ 0”:{ … } } } etc
    - e.g. ”TrackParticle” : {"InDetTrackParticles":{"Trk 0”:{ … } } }
      - To see it all, have a look at ‘http://emoyse.web.cern.ch/emoyse/WebEventDisplay/EventDump.json' (Can enter this URL into JSONlint.com and validate it to see it formatted nicely).
- Currently we have:
  - Track(particles)
  - Jets
  - MET
  - Cells (hardcoded to be displayed at the ATLAS calo boundaries at the moment)

Visualising more complex data



https://emoyse.web.cern.ch/emoyse/WebEventDisplay/jsdisplay.html

(might need to disable geometry on slower machines)

Used in two talks,

@ Connecting the dots, Vienna 2016

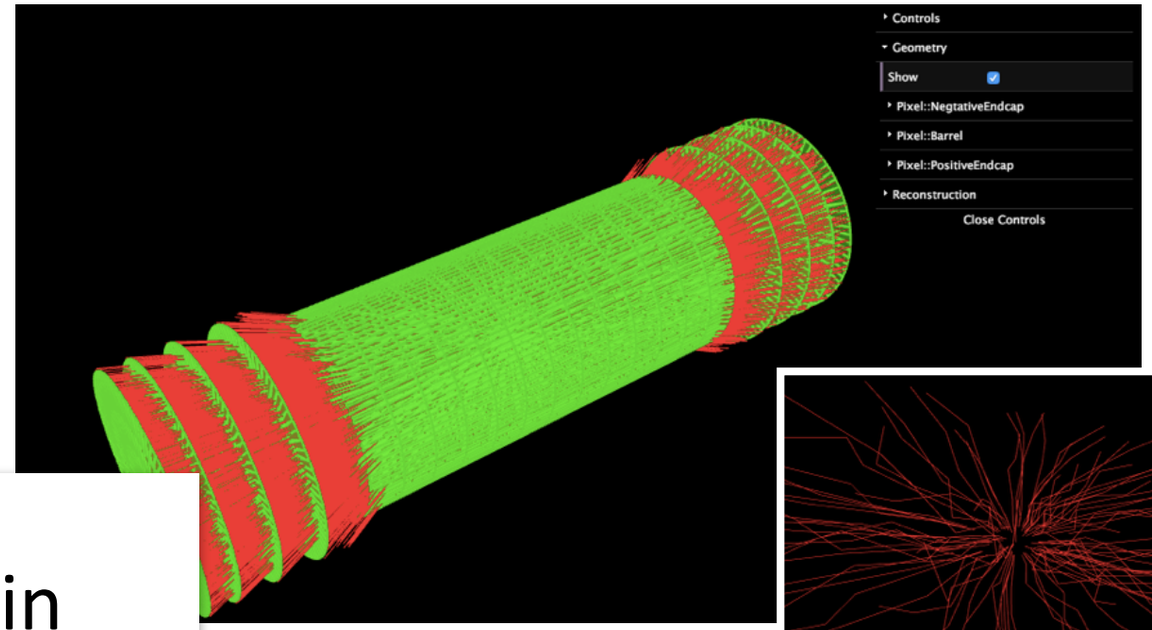
link

ATS release planning - alpha

0.1.0
alpha release: first repository build Athena/Gaudi
Release date: 19/Feb/16
14 Issues · Release Notes

14 of 14 issues have been resolved

WebEventDisplay by Edward Moyse



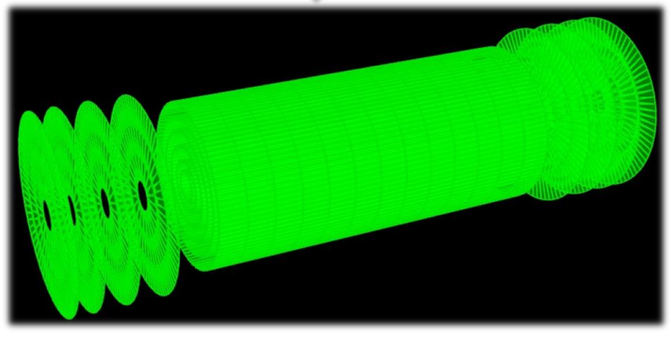
the frank reason why we haven't released

18

https://indico.hephy.oeaw.ac.at/event/86/session/5/contribution/37

Example: GenericDetectorPlugin

```
21        # the pixel module
22        PixelModule = DetectorModule(None,8.4,32.0,0.15)
23        # the first layer
24        PixelLayer0 = CylinderLayer(PixelModule, 33., 24, 13, 0.2, 2., 0.5, 5.)
25        PixelLayer1 = CylinderLayer(PixelModule, 55., 40, 13, 0.2, 2., 0.5, 5.)
26        PixelLayer2 = CylinderLayer(PixelModule, 88., 60, 13, 0.2, 2., 0.5, 5.)
27        PixelLayer3 = CylinderLayer(PixelModule, 120., 72, 13, 0.2, 2., 0.5, 5.)
28        PixelLayer4 = CylinderLayer(PixelModule, 150., 84, 13, 0.2, 2., 0.5, 5.)
29        # define the pixel barrel volume
30        PixelBarrel = BarrelVolume( [ PixelLayer0, PixelLayer1, PixelLayer2, PixelLayer3, PixelLayer4 ] )
```



24/02/2016 J.Hrdinka, A.Salzburger, C.Gumpert - Overview of the ATS project and its use for FCC - Connecting the Dots 2016 15

https://indico.hephy.oeaw.ac.at/event/86/session/5/contribution/36

Technologies and problems

- Resources used:
  - Three.js - fantastic javascript webgl library
  - dat.gui - used for menus
- Data input:
  - TADA - ATLAS dedicated format feeds into JSON and then to webpage template
  - WebEventDisplay - separate JSON formats for geometry and event data
- Problems:
  - for TADA - nothing really. It has very simple needs!
  - WebEventDisplay
    - some limitations with transparency in three.js
    - performance can get a bit slow (but still pretty impressive for a browser, and many optimisations possible)
    - biggest issue is GUI - dat.gui is a bit limited if we want to do more

One issue worth addressing…

- Shared formats would allow us to easily share tools e.g. web based viewers, but also stand-alone tools
  - I chose JSON, primarily for its portability and usability
    - Human readable, easily manipulated inside javascript/python etc
    - Obviously not suitable for large scale storage - but that's not the need I wanted to address
  - I have some unfinished documentation on the WebEventDisplay (**terrible** name I know) repo:
    - https://gitlab.cern.ch/emoyse/WebEventDisplay
  - Happy to keep working on this, but I admit that it's very basic - does anyone have better? Shared format possible?
- I think that even if a 'better' shared format becomes available/commonly used, it is still worth having a simple JSON format