

Visualizations and interfaces for end users

*Super-B Computing R&D Workshop
Ferrara, Italy*

M. Bellis

Department of Physics
Stanford University

Mar. 11th, 2010



PREPARE YOUR LAPTOPS!

Links you may want to have open.

<http://vpython.org/>

<http://www.assembla.com/wiki/show/viewpoints>

http://www.slac.stanford.edu/~bellis/viewpoints_demo.html

http://www.slac.stanford.edu/~bellis/data_formats.html

http://en.wikipedia.org/wiki/Extreme_Programming

Links you may want to download to your desktop

http://www.slac.stanford.edu/~bellis/HEP_data_for_education/video_Y3S_to_hb.mpg

http://www.slac.stanford.edu/~bellis/HEP_data_for_education/video_BtoDpi_PID.mpg

http://www.slac.stanford.edu/~bellis/HEP_data_for_education/video_PID_process.mpg

http://www.slac.stanford.edu/~bellis/HEP_data_for_education/viewpoints_usage.mpg

http://www.slac.stanford.edu/~bellis/HEP_data_for_education/svt_lumi_0.ogg

OUTLINE

- 1 INTRODUCTION
- 2 4-VECTOR VIEWERS
- 3 LASS EXPERIENCE
- 4 MUSINGS

INTRODUCTION

- Much of this has come out my efforts in:
 - Outreach and education (public, high school)
 - Training undergraduate students.
 - Training graduate students.
- Not exactly the same, but similar challenges.
- Training a new collaborator, whether undergrad or post-doc, presents challenges.
- How do you get them contributing as soon as possible?

INTRODUCTION

- 4-vector viewers.
 - Developing animations to better understand complex physics.
- Viewpoints
 - NASA package for visualizing multi-variate datasets.
- LASS
 - Experience resurrecting a 30-yo dataset.
- General comments.
 - My opinions based on Jefferson Lab (CLAS, GlueX) and BaBar experience.
 - Extreme programming.

4-vector viewers

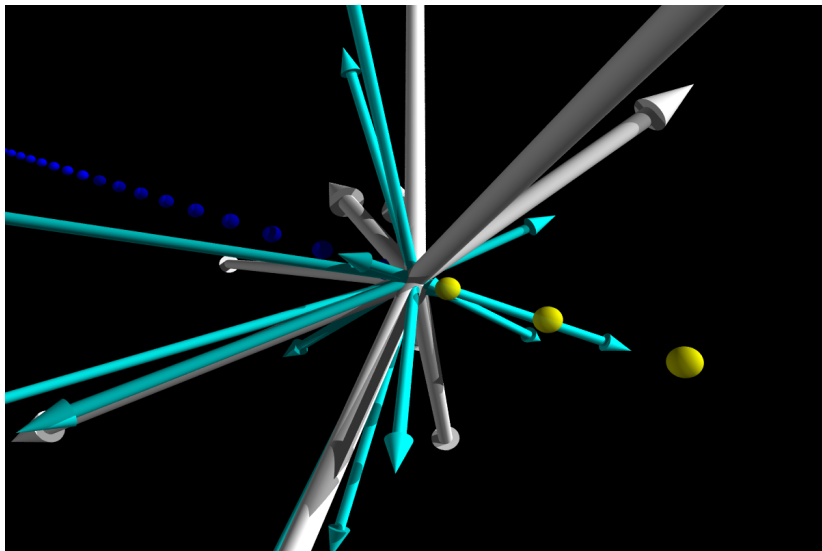


FIGURE:

INTRODUCTION

- 4-vector viewers.
 - Developing animations to better understand complex physics.
 - <http://vpython.org/>
 - 3-D visual package.
 - Python interface.
 - Used in many university physics programs.
 - I read in from text files created from our root ntuples.
 - For now I show you some movies made from these scripts.
 - These are also uploaded to the Indico workshop site.

http://www.slac.stanford.edu/~bellis/HEP_data_for_education/video_Y3S_to_hb.mpg

http://www.slac.stanford.edu/~bellis/HEP_data_for_education/video_BtoDpi_PID.mpg

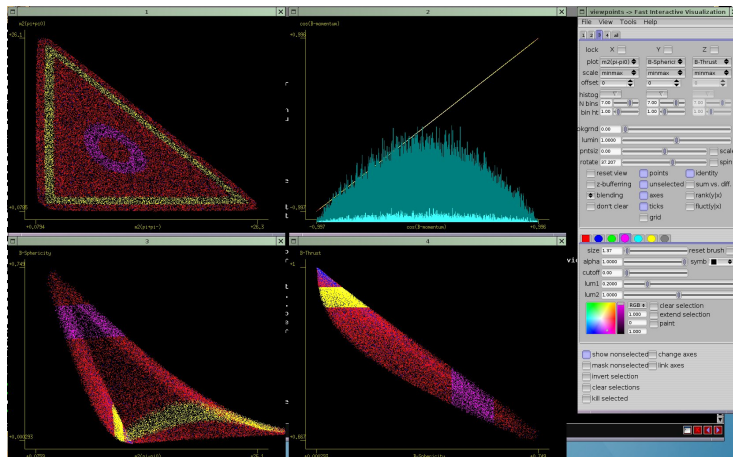
http://www.slac.stanford.edu/~bellis/HEP_data_for_education/video_PID_process.mpg

Viewpoints

CURRENT EFFORT

<http://www.assembla.com/wiki/show/viewpoints>

http://www.slac.stanford.edu/~bellis/viewpoints_demo.html



WHAT IS VIEWPOINTS?

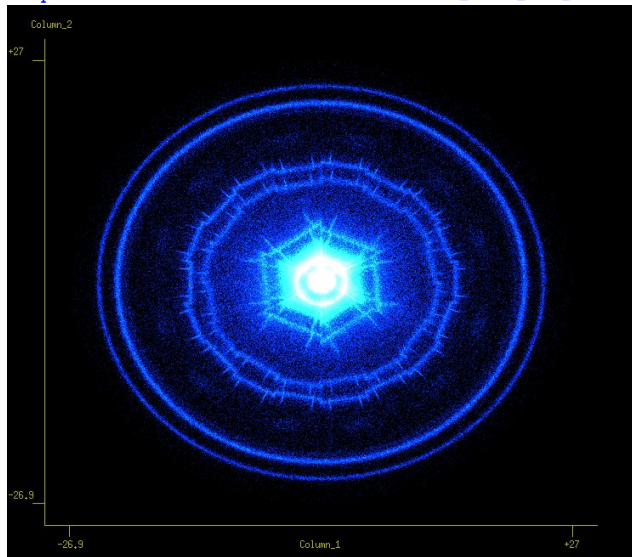
- NASA analysis tool.
- Viewpoints (vp) is a high-performance visualization and analysis tool for large, complex, multidimensional data sets.
- It allows interactive exploration of data in 100 or more dimensions with sample counts, or the number of points, exceeding 10^6 (up to 10^8 depending on available RAM).
- Viewpoints was originally created for use with the extremely large data sets produced by current and future NASA space science missions, but it has been used for a wide variety of diverse applications ranging from aeronautical engineering, quantum chemistry, and computational fluid dynamics to virology, computational finance, and aviation safety.

WHAT IS VIEWPOINTS?

- http://www.slac.stanford.edu/~bellis/viewpoints_demo.html
- http://www.slac.stanford.edu/~bellis/HEP_data_for_education/viewpoints_usage.mpg
 - Demo video is uploaded to Indico workshop page.
- Multivariate analysis
- Event shape variables and discriminating variables.
- Viewpoints is *not* a replacement for ROOT...but can augment your analysis and learning process.
- Cuts should *never* be decided upon using Viewpoints, but can perhaps help provide a more intuitive understanding of your analysis.
- We can't underestimate the importance of developing that intuition within the collaboration.
- *You must try this tool out on your own data!*

PAIR PRODUCTION

http://www.slac.stanford.edu/~bellis/HEP_data_for_education/svt_lumi_0.ogg



LASS experience

DATA FORMATS

- http://www.slac.stanford.edu/~bellis/data_formats.html
- File formats.
- Consider documentation.
 - `man`
 - `man` pages in use starting in 1971.
 - Huge boon to always have documentation at fingertips.
 - Imposed pseudo-requirements on programmers.
 - FITS (Flexible Image Transport System)
 - Used in astro community.
 - Header with information about the image.
 - Many implementations: Fortran, C/C++, Perl, Python, Java, etc.
 - Can even read with **gimp** or **Photoshop**.
 - I want a file format that carries its own documentation around with it.

DATA FORMATS

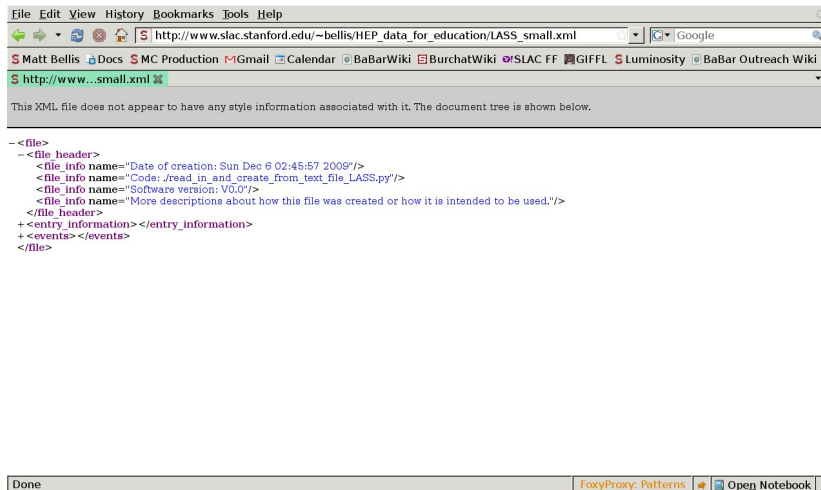
- How can I learn from this?
- Both for collaboration analysis and outreach efforts.
- Try converting LASS data.
 - Fixed target SLAC experiment.
 - 11 GeV K^+/K^- beams on hydrogen target.
 - 1977-1978, 1981-1982
 - $\sim 100\text{M}$ triggers.
 - David Aston (SLAC) revived the data.
 - “The original format (in fact, still is the format, since the files are just straight bit-copies of the originals) is IBM VBS – “Variable Blocks Spanned” – format, processed with VM/CMS; big-endian with IBM’s mainframe floating point format (not IEEE).”
 - David gave me text files...

LASS TEXT FILES

```
new EVENT: run event hw & sw triggers    9550      3    18  8E18
Topology : 1
Vertices  tracks:  1 5
  Primary vtx: x y z d^2   -0.763999999 -0.40200001  81.5400009  0.0351999998
Beam charge px py pz :  1 -0.00964925718  0.00366006303  10.9029951
tracks/charge px py pz :
3  0.00761500327  0.309128046  8.70612812
5  0.133273423  -0.16253081  0.656466305
2  0.0946368724  -0.149057582  0.53865546
-1 -0.0939974785  0.214401409  0.689579844
Topology : 2
Vertices  tracks:  2 4
  Primary vtx: x y z d^2   -0.786000013 -0.375999987  80.5699997  0.0177999996
Beam charge px py pz :  1 -0.00964925718  0.00366006303  10.9029951
tracks/charge px py pz :
3  0.00738755707  0.309133559  8.70612812
2  0.0962994769  -0.147988901  0.53865546
0  0.037703827  0.0509431213  1.34604621
  Secondary vtx: x y z d^2   -0.858938336 -0.303011149  81.9899979
0.00257943221
daughters/charge px py pz :
5  0.13265042  -0.163039669  0.656466305
-1 -0.094946593  0.213982791  0.689579844
new EVENT: run event hw & sw triggers    9550      91    18  CE1C
Topology : 1
Vertices  tracks:  1 5
```

Convert to XML...

LASS XML FILES



File Edit View History Bookmarks Tools Help

http://www.slac.stanford.edu/~bellis/HEP_data_for_education/LASS_small.xml

Matt Bellis Docs MC Production Gmail Calendar BaBarWiki BurchatWiki SLAC FF GIFFL Luminosity BaBar Outreach Wiki

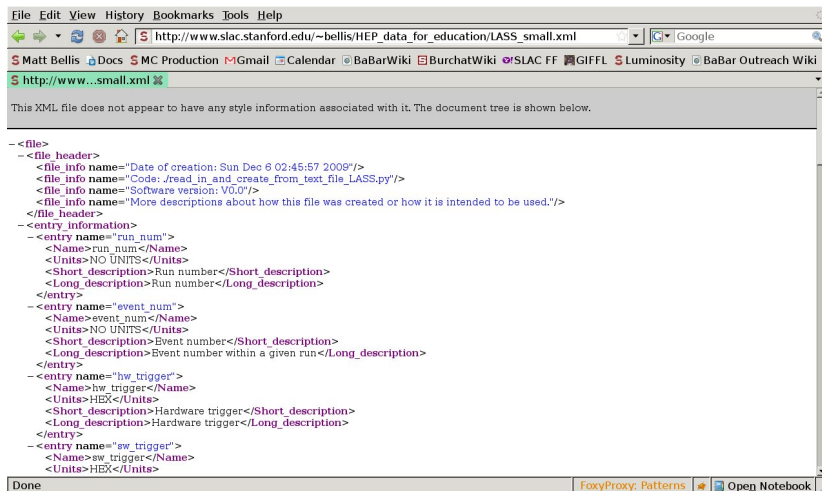
http://www...small.xml

This XML file does not appear to have any style information associated with it. The document tree is shown below.

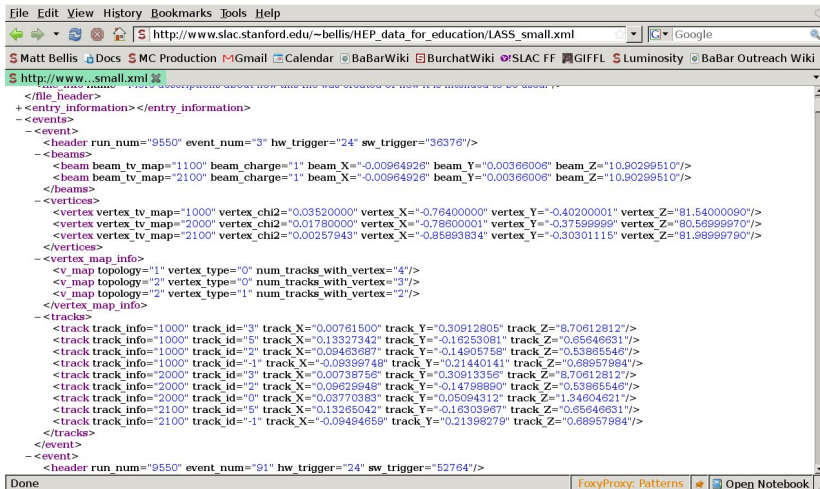
```
- <file>
  - <file_header>
    <file_info name="Date of creation: Sun Dec 6 02:45:57 2009"/>
    <file_info name="Code: ./read_in_and_create_from_text_file_LASS.py"/>
    <file_info name="Software version: V0.0"/>
    <file_info name="More descriptions about how this file was created or how it is intended to be used."/>
  </file_header>
  + <entry_information> </entry_information>
  + <events> </events>
</file>
```

Done FoxyProxy: Patterns Open Notebook

LASS XML FILES



LASS XML FILES



```
</file_header>
+ <entry_information> </entry_information>
- <events>
  - <event>
    <header run_num="9550" event_num="3" hw_trigger="24" sw_trigger="36376"/>
    - <beams>
      <beam beam_tv_map="1100" beam_charge="1" beam_X="-0.00964926" beam_Y="0.00366008" beam_Z="10.90299510"/>
      <beam beam_tv_map="2100" beam_charge="1" beam_X="-0.00964926" beam_Y="0.00366008" beam_Z="10.90299510"/>
    </beams>
    - <vertices>
      <vertex vertex_tv_map="1000" vertex_chi2="0.03520000" vertex_X="-0.76400000" vertex_Y="-0.40200001" vertex_Z="81.54000090"/>
      <vertex vertex_tv_map="2000" vertex_chi2="0.01780000" vertex_X="-0.78600001" vertex_Y="-0.37599999" vertex_Z="80.56999970"/>
      <vertex vertex_tv_map="2100" vertex_chi2="0.00257943" vertex_X="-0.85893834" vertex_Y="-0.30301115" vertex_Z="81.98999790"/>
    </vertices>
    - <vertex_map_info>
      <v_map topology="1" vertex_type="0" num_tracks_with_vertex="4"/>
      <v_map topology="2" vertex_type="0" num_tracks_with_vertex="3"/>
      <v_map topology="2" vertex_type="1" num_tracks_with_vertex="2"/>
    </vertex_map_info>
    - <tracks>
      <track track_info="1000" track_id="3" track_X="0.00761500" track_Y="0.30912805" track_Z="8.70612812"/>
      <track track_info="1000" track_id="5" track_X="0.13327342" track_Y="-0.16253081" track_Z="0.65646631"/>
      <track track_info="1000" track_id="2" track_X="0.09463687" track_Y="-0.14905758" track_Z="0.53865546"/>
      <track track_info="1000" track_id="-1" track_X="-0.09399748" track_Y="0.21440141" track_Z="0.68957984"/>
      <track track_info="2000" track_id="3" track_X="0.00738756" track_Y="0.30913356" track_Z="8.70612812"/>
      <track track_info="2000" track_id="2" track_X="0.09629948" track_Y="-0.14798890" track_Z="0.53865546"/>
      <track track_info="2000" track_id="0" track_X="0.03770383" track_Y="0.05094312" track_Z="1.34604621"/>
      <track track_info="2100" track_id="5" track_X="0.13265042" track_Y="-0.16303967" track_Z="0.65646631"/>
      <track track_info="2100" track_id="-1" track_X="-0.09494659" track_Y="0.21398279" track_Z="0.68957984"/>
    </tracks>
  </event>
  - <event>
    <header run_num="9550" event_num="91" hw_trigger="24" sw_trigger="52764"/>
```

DATA FORMATS

- Can we replicate this in ROOT?
- Suggestion from Rene Brun: GetUserInfo in TTree.
- Uses TList to hold anything derived from TObject.
- Define my own standard: TList's of TString's
 - 0th entry: Information about the file.
 - 1st entry: Defines what information is held for each entries.
 - 2nd – n^{th} entry: Information about the TTree entries.
- Only store int's, float's and arrays of these in the TTree.
- Provide PyRoot script to dump information.

LASS ROOT FILES

```
> ./dump_header_info.py -h
Usage: dump_header_info.py [options]

Options:
  -h, --help            show this help message and exit
  -a, --all              Dump all the information
  -t TREE_NAME, --tree-name=TREE_NAME
                        Name of the TTree object
  -i INFO, --info=INFO  Append these choices for specific information to dump
                        out. Note that only the # of the information is
                        necessary. e.g: -i 0 -i2
```

LASS ROOT FILES

```
> ./dump_header_info.py LASS_small.root
Information about how this file was generated
Date of creation: Sun Dec 6 02:45:58 2009
Code: ./read_in_and_create_from_text_file_LASS.py
Software version: V0.0
More descriptions about how this file was created or how it is intended to be used.

Description of entries in header
0: Name
1: Units
2: Short description
3: Long description
```

LASS ROOT FILES

```
> ./dump_header_info.py LASS_small.root -a
Information about how this file was generated
Date of creation: Sun Dec 6 02:45:58 2009
Code: ./read_in_and_create_from_text_file_LASS.py
Software version: V0.0
More descriptions about how this file was created or how it is intended to be used.
```

Description of entries in header

Name

Units

Short description

Long description

run_num

Name: run_num

Units: NO UNITS

Short description: Run number

Long description: Run number

event_num

Name: event_num

Units: NO UNITS

Short description: Event number

Long description: Event number within a given run

hw_trigger

Name: hw_trigger

Units: HEX

Short description: Hardware trigger

Long description: Hardware trigger

Musings

MY PAST EXPERIENCE

- Jefferson Lab (CLAS, GlueX)
- SLAC (BaBar)
- Computing issues for end-user are lab/experiment independent.
 - How do new analysts come up to speed?
 - Physics?
 - Computing environment?

MY PAST EXPERIENCE

- Common tools are a big help!
 - ROOT
 - CVS, Subversion, make
- People still find ways to make things different.
 - brrroot, BOS files, srtpath, Scientific Linux
- Everytime we write our own solution, we close ourselves off from the broader computing community.
- Or at least we increase the time to ramp up for new collaborators.
- *This has huge implications when it comes to service tasks.*
- Most analysts will contribute code at some point.

MY PAST EXPERIENCE

- Every experiment has problems with analysis code bifurcation.
- Why?
 - Code prima donnas.
 - Original code does do what they want.
 - Too difficult to change.
 - Not their native computing language.
 - Original author no longer around.
 - *Original code is poorly documented.*
 - Very little mandate *not* to duplicate analysis code from within collaboration.

- Extreme programming.
- http://en.wikipedia.org/wiki/Extreme_Programming
 - Emphasizes agile programming.
 - Programming in groups.
 - This means two people at the same keyboard!
 - Unit testing.
 - Write the test case first!
 - My two cents...write the documentation zeroth!

- Almost everyone will contribute code at some point.
- We all spend so much time wading through code to find out what's in our data!
 - “Code is self-documenting”
 - “If you really want to know what's in the ntuple, you *have* to go to the source code”
 - “You can't enforce documentation.”
- Revisiting the LASS experience...
 - Can you enforce that *every* code that returns user data must also be able to return it's own documentation?
 - This could be stored in some percentage of the files.
 - It could just be links to arxiv postings? Collaboration links?
 - Could you require that this documentation is written first?

- If you have a routine with *English*, not meta, information embedded...as well as a test suite which was written first, does this allow you to be more flexible?
- If I have written words on what some code should do, I can code that in FORTRAN, C++ or Go if I want.
- If I have a test suite and text files of input and testable output, I can know whether or not my code is equivalent or faster.
- Maybe this isn't scalable...maybe it's a way for this to become scalable?
- In any event, things to think about...

Thanks for your time

BACKUP SLIDES