FPTrack

A tracking algorithm for forward physics

Will Plano
University of Manchester

What is FPTrack

- FPTrack initially written by Peter Bussey of Glasgow.
- A fast simulation of the beam optics from the interaction point through to forward detectors down the beamline.
- Allows tracking of protons from any central diffractive event (e.g. CEP of Higgs) to detectors.
- Currently used to study CEP Higgs etc. at FP420 including calibration of detectors.

What is FPTrack...

- Beamline modelled as a series of optical elements, Quads, Dipoles etc.
- Each element is a set of transport equations for each of the 6 particle variables x position/angle, y position/angle, z position, momentum loss.
  N.B. transport equations need not be linear.
- E.g. (lossless) horizontally focussing quad.

$$x_f = \cos(\sqrt{|k|}\,l)\,x_i + 1/\sqrt{|k|}\sin(\sqrt{|k|}\,l)\,x'_i$$

$$x'_f = -\sqrt{|k|}\sin(\sqrt{|k|}\,l)\,x_i + \cos(\sqrt{|k|}\,l)\,x'_i$$

$$y_f = \cosh(\sqrt{|k|}\,l)\,y_i + 1/\sqrt{|k|}\sinh(\sqrt{|k|}\,l)\,y'_i$$

$$y'_f = \sqrt{|k|}\sinh(\sqrt{|k|}\,l)\,y_i + \cosh(\sqrt{|k|}\,l)\,y'_i$$

$$z_f = z_i + l$$

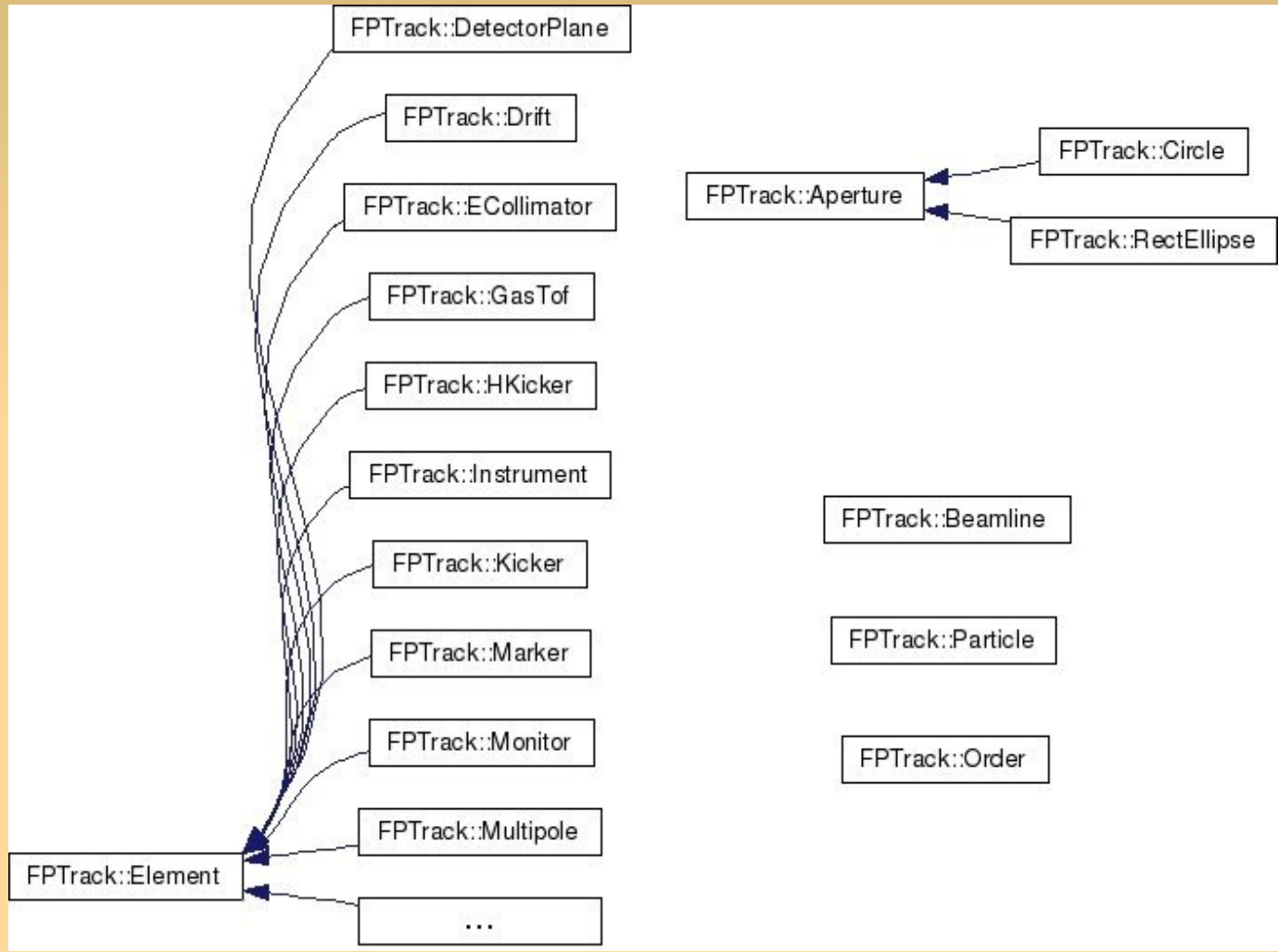
$$dp_f = dp_i$$

Using FPTrack

- Re-written in C++ to allow ease of modification (changing detector geometry, adding/modifying collimators etc.) and to allow running alongside MC generator on event by event basis.
- Uses an object oriented approach with classes for different magnet types, apertures etc.
- Builds into a library so that it does not need to be re-compiled each time the run parameters change.

Class Diagram

- Main class is the `Beamline` class.
- `Beamline` contains a vector of `Element`s.
- Each `Element` has an `Aperture`.
- `Particle`s are passed down the `Beamline` and stored in each `Element`.

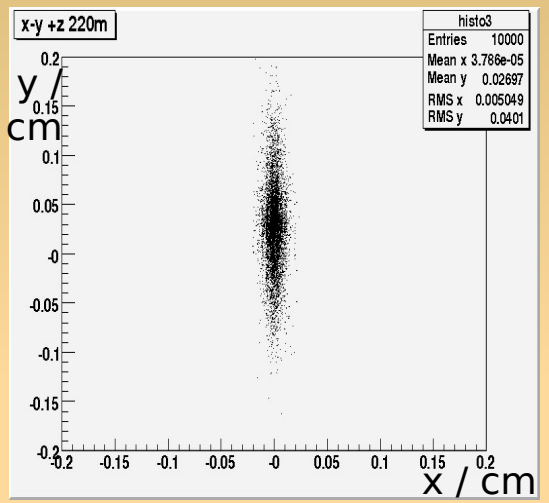
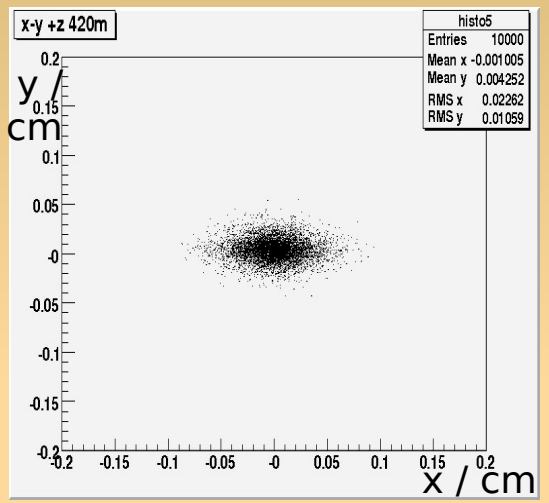
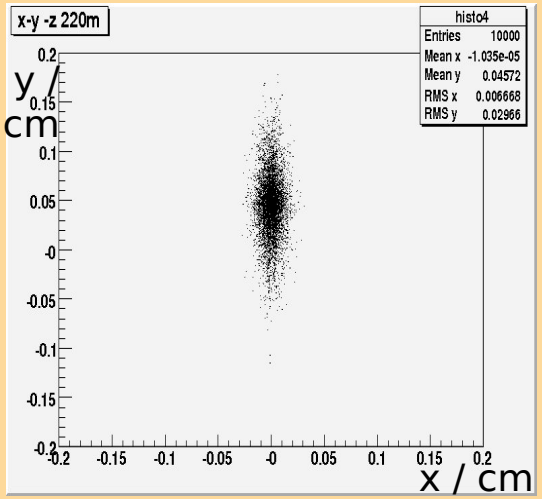
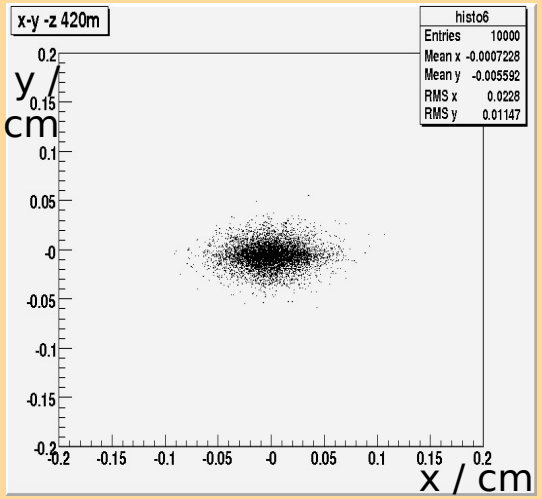


Example Code

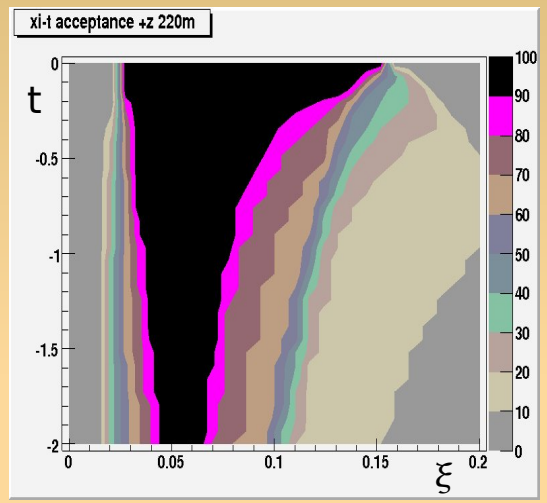
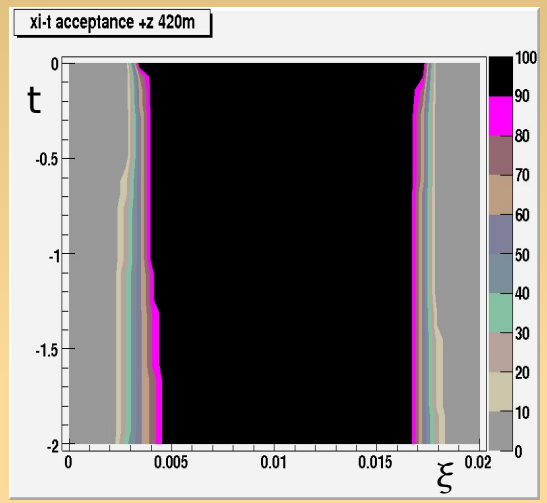
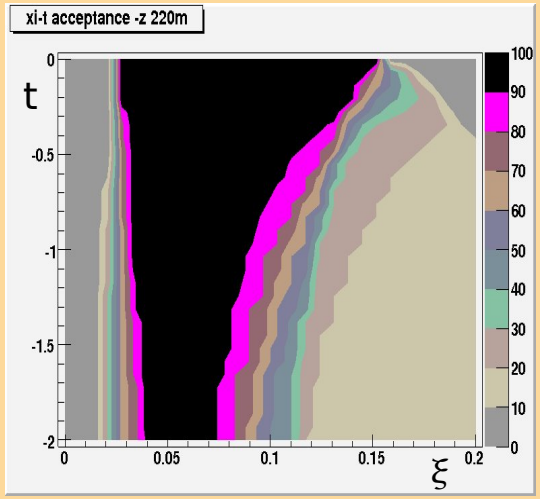
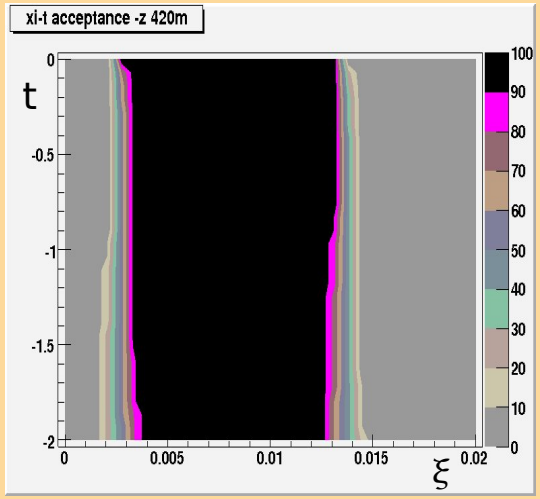
- Only needs a few lines of code to run.
- Create a beamline.
- Optionally add elements.
- Make / generate a Particle (e.g. from ExHuME).
- Process particle in beamline.
- Retrieve data from beam elements.

```
1  #include <iostream>
2  #include "CLHEP/Vector/LorentzVector.h"
3  #include "Beamline.h"
4  #include "DetectorPlane.h"
5
6  int main(int argc, char** argv) {
7    FPTrack::Beamline myBeam(FPTrack::IP1,
                              FPTrack::PLUS_Z,
                              "twiss_b1.txt");
8    myBeam.AddElement(
       new FPTrack::DetectorPlane(
         "FP420 Pot",
         CLHEP::Hep3Vector(0.0, 0.0,  420.0)));
9    CLHEP::HepLorentzVector ip(0.0, 0.0, 0.0, 0.0);
10   CLHEP::HepLorentzVector pmom(0.0, 0.0,  7000.0, 7000.0);
11   FPTrack::Particle proton(ip, pmom);
12   myBeam.ProcessParticle(proton);
13   FPTrack::Element* myPot =
       myBeam.GetElementPtrByName("FP420 Pot");
14   std::cout << "Proton went here: "
             << myPot->GetParticleOut() << std::endl;
15   return 0;
16 }
```

Beam spots at ATLAS ±220/420m

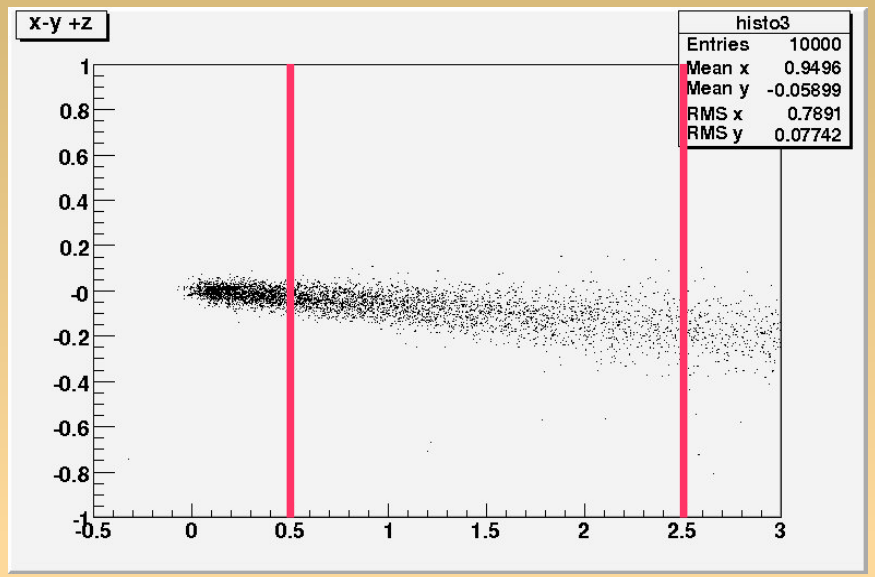
x-y / cm	220m	420m
+z		
-z		

ξ-t acceptance, ATLAS ±220/420m

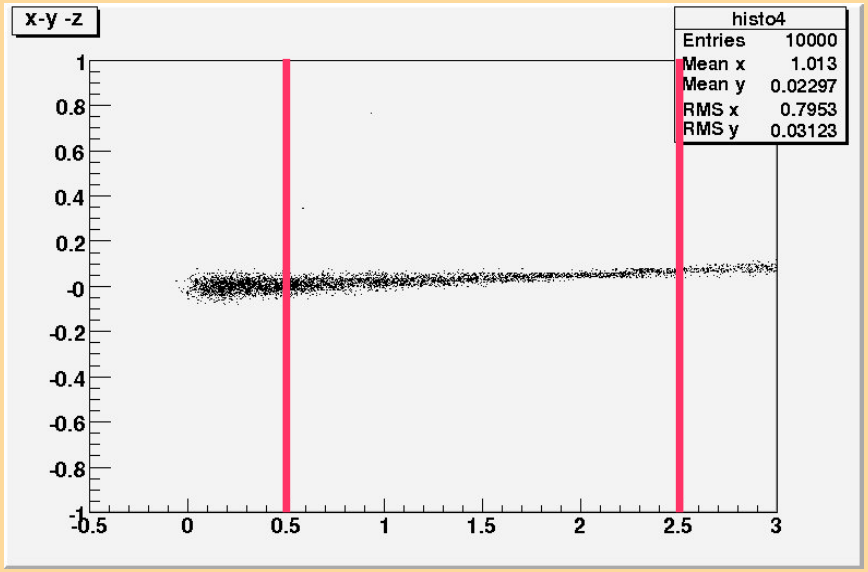
ξ - t	220m	420m
+z		
-z		

CEP Higgs (120) protons at 420m

+z



-z



Summary

- Fast, ~0.3ms per proton per magnet on my Intel® Centrino™ laptop @ 600MHz i.e. 50,000 events to ±420m (35 magnets each way) takes ~20mins.
- Validated, gives the right answers :)
- Easy to use (subjective but try it and see)
- Ask me if you have any questions:

w.g.plano@postgrad.manchester.ac.uk