Track Reconstruction: the ftf and trf toolkits

Norman Graf (SLAC)

Common Software Working Meeting
CERN, January 31, 2013

ftf Track Finding

- Using a conformal mapping technique
  - Maps curved trajectories onto straight lines
  - Simple link-and-tree type of following approach associates hits.
  - Once enough hits are linked, do a simple helix fit
    - circle in r-phi
    - straight line in s-z
    - simple iteration to make commensurate
  - Use these track parameters to predict track into regions with only 1-D measurements & pick up hits.
  - Outside-in, inside-out, cross-detector: completely flexible as long as concept of *layer* exists.
    - Runtime control of finding details.
  - Simple fit serves as input to final Kalman fitter.

ftf for LC

- Worked with Steve Aplin last year during his stay at KEK.
- Implemented ftf functionality into a Marlin Processor
  - able to find tracks in ttbar events
- Development work stalled as pressure of the DBD mounted.
- Interested in collaborating to investigate whether this provides useful functionality
- Topological finder, only uses concept of layer, does not need geometry

trf

- A toolkit for:
  - Describing material and hits along a trajectory.
  - Propagating from one surface to the next.
    - with a model of multiple scattering.
    - with a model of energy loss.
  - Kalman filtering tracks that are described by the above components
  - Providing track fits and covariance matrices at locations along the track
- trf has its own geometry and hit classes
- trf has its own choices for track parameters,

Surfaces

- Surfaces generally correspond to geometric shapes representing detector devices.
- They provide a basis for tracks, and constrain one of the track parameters.
- The track vector at a surface is expressed in parameters which are “natural” for that surface.

- Surfaces require some user input to decide which detector elements should be compressed, where hits should be defined, etc.
  - Interaction with geometry and digitization

1.) Cylinder

- Surface defined coaxial with z, therefore specified by a single parameter r.
- Track Parameters: $(\phi, z, \alpha, \tan\lambda, q/p_T)$
- Bounded surface adds z_{min} and z_{max}.
- Supports 1D and 2D hits:
  - 1D Axial: ϕ
  - 1D Stereo: $\phi+\kappa z$
  - 2D Combined: (ϕ, z)

- Needs r, zmin, zmax

2.) XY Plane

- Surface defined parallel with z, therefore specified by distance u from the z axis and an angle ϕ of the normal with respect to x axis.
- Track Parameters: (v, z, dv/du, dz/du, q/p)
- Bounded surface adds polygonal boundaries.
- Supports 1D and 2D hits:
  - 1D Stereo: $w_v*v + w_z*z$
  - 2D Combined: (v, z)

- Needs u, ϕ and polygonal boundaries

3.) Z Plane

- Surface defined perpendicular to z, therefore specified by single parameter z.
- Track Parameters: (x, y, dx/dz, dy/dz, q/p)
- Bounded surface adds polygonal boundaries.
- Supports 1D and 2D hits:
  - 1D Stereo: $w_x*x + w_y*y$
  - 2D Combined: (x,y)
- Needs z and polygonal boundaries

4.) Distance of Closest Approach

- DCA is also a 5D **Surface** in the 6 parameter space of points along a track.
- It is **not** a 2D surface in 3D space.
- Characterized by the track direction and position in the (x,y) plane being normal; $\alpha=\pi/2$.
- Track Parameters: $(r, z, \phi_{dir}, \tan\lambda, q/p_T)$

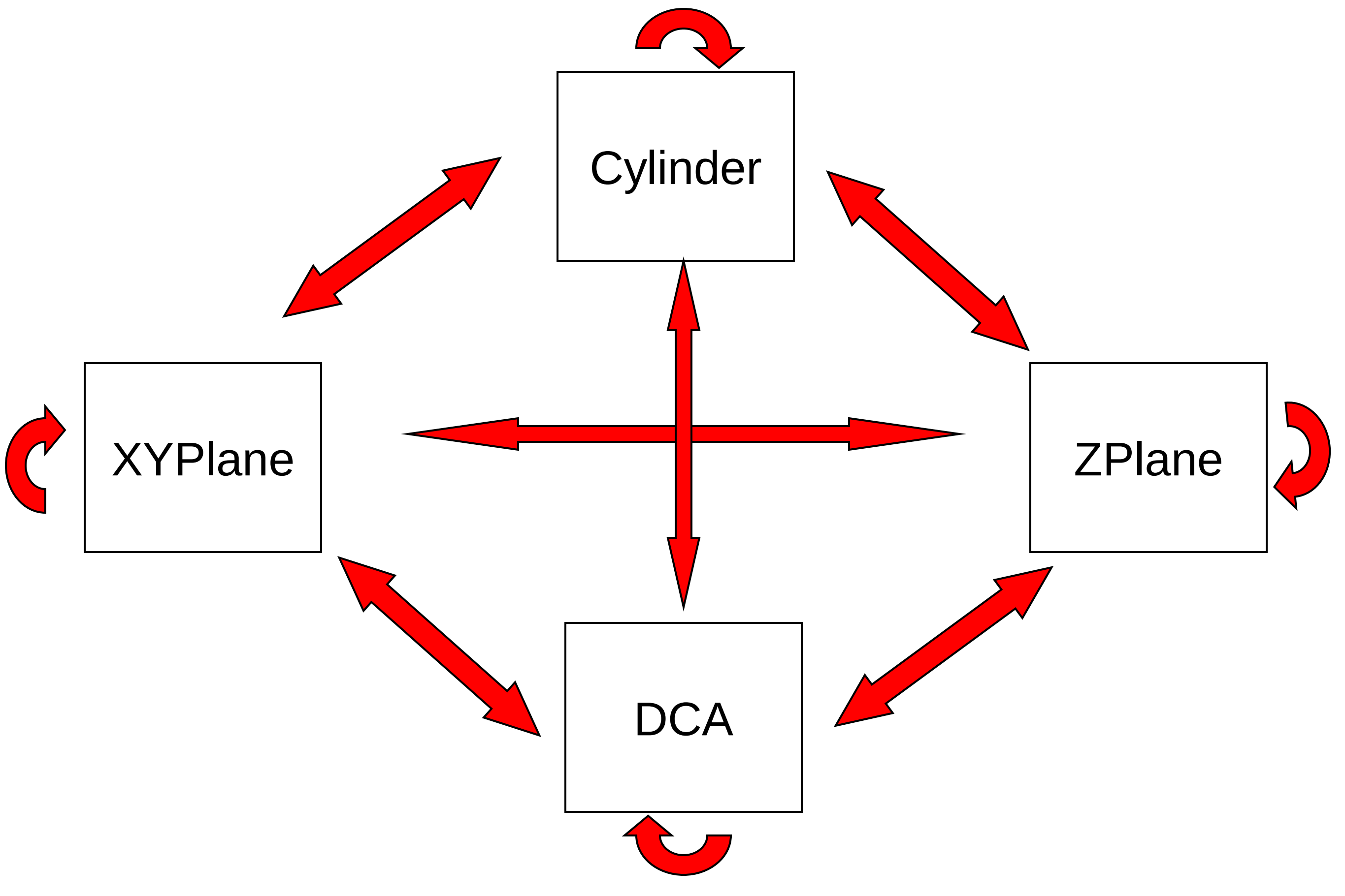
- Not a geometrical Surface.

Propagator

- Propagators propagate a track (and optionally its covariance matrix) to a new surface.
- A propagator returns an object of type PropStat which describes the status of the attempted propagation:
  - *i.e.* whether it was successful and, if so, in which direction the track was propagated (forward or backward).
- Interacting Propagators modify the track and its covariance matrix (in case of energy loss), or just the covariance matrix (thin multiple scattering.)
- Needs a Magnetic Field manager.

Propagators

- Propagators are defined for all combinations of surfaces:



Propagators

- Propagators take a track from its current state and transport it to a given surface.
  - Currently does not know or care about intervening Surfaces.

- Needs intelligent Geometer to decide, given a current track state, what the next Surface should be.

Interactors

- Describes the interface for a class which modifies a track. Examples are:
- Multiple Scattering
  - ThickCylMS
  - ThinXYPlaneMS
  - ThinZPlaneMS
- Energy Loss
  - CylELoss

- Existing Interactors use effective X0 for Surfaces
  - some intelligence required

Detector

- Currently use compact.xml to create a tracking Detector composed of surfaces, along with interacting propagators to handle track vector and covariance matrix propagation, as well as energy loss and multiple scattering.
  - Silicon pixel and microstrip wafers modeled as either xyplane or zplane.
  - TPC modeled as cylindrical layers (corresponding to pad rows).
  - Currently using thin multiple scattering approximations.
  - Using pure solenoidal field propagators
    - Runge-Kutta propagators available when needed.

Recent Developments

- Java version in org.lcsim being used by HPS for real data from the test run.
- Magnetic Field Map handling being optimized
- Runge-Kutta stepper being optimized
  - adaptive step size
- Support being added for tilted planes.
  - for misalignment
  - intentional tilt to account for Lorentz angle
- Alignment software being implemented
  - Not within trf framework

Summary

- trf toolkit provides full infrastructure for defining detectors, hits & tracks as well as propagators, interactors and fitters.
  - Currently working on ~generic interface between compact detector description and trf tracking Detector.
  - Effort being devoted to “smart” propagator.
- Available in Java (org.lcsim) as well as C++ (standalone).
- ftf pattern recognition based on 2-D measurements on surfaces was implemented as Marlin Processor.
  - Fast, with high efficiency.
- Lots of work ahead to interface with the geometry system.