Seed-based Tracking Algorithm for SiD

Richard Partridge

Brown University

Vancouver Linear Collider Workshop

July 19-22, 2006

Motivation

- The SiD detector concept seeks to provide robust, high precision tracking in a compact volume using a silicon strip vertex detector and a silicon strip outer tracker
- An essential part of the design process is to use realistic simulations to understand the performance trade-offs in making design choices
- The simulation tools need to be flexible, sensitive to design options under consideration, and not require extensive tuning / coding for each design option

A Partial List of Questions

- Do we have enough/too many layers?
- Do we need stereo layers in the barrel strip tracker? Which layers?
- Do we need to fill the gap between the vertex and strip detectors?
- What is the tracking efficiency (overall, forward, core of jets, etc.)?
- What are the rates for fake/mis-measured tracks?
- Can we efficiently find long-lived decays (K_S, Λ, long-lived b decays)?
- What is the impact of inefficiency/fakes on PFA/physics measurements?
- What is the impact of options that add material (stereo, more layers) on PFA/physics measurements?

SeedTracker - Overview

- SeedTracker is being developed as a track finding program in the org.lcsim framework
- Algorithm is largely based on tracking algorithms developed in the hep.lcd framework, with a number of generalizations to assist in performing design studies
  - Utilize geometry information in org.lcsim
  - Allow both inside-out and outside-in algorithms
  - Minimize number of parameters and make them all accessible to the user

SeedTracker Features

- Implements barrel and disk geometries
- Finds tracks in central barrel layers, forward disk layers, or a combination of barrel and disk layers
- Extracts geometry using org.lcsim geometry classes
- Extracts hit resolution from track hit class
  - Future: use geometry classes to determine multiple scattering errors
- Full user control of track-finding "strategies"
- Allow multiple track-finding strategies to be pursued
- To speed calculations:
  - Ignore multiple scattering correlations between hit measurements
  - Currently uses a 2D circle fitter that implements the Karimaki algorithm
  - Ignore correlations between r-ϕ and r-z fits

Strategies

- SeedTracker processes a list of track-finding "strategies"
- A strategy consists of:
  - Detectors / Layers to be used for this strategy
  - Role of each layer (track seed, seed confirmation, additional hits)
  - Cuts on track quantities (minimum pT, maximum DCA)
  - Cuts on pattern recognition quantities (χ^2, minimum number of hits)
- The user defines the strategies to be pursued
  - This allows track finding to be tailored to a particular design study

Finding Seeds

- Track seeds are generated by forming helices from all combinations of hits in three “Seed Layers”
- Seed Layers must provide 3D hit position measurements
  - Pixel detector does this by default
  - Pairs of strip layersStrip detectors with pairPair of strip layers with stereo
- To be a seed, a helix must:
  - Exceed the minimum pT requirement
  - Satisfy a cuts on distance of closest approach in r-ϕ and r-z planes
  - Have a z coordinate of middle layer to be consistent with inner and outer layer hits
- Which layers can be seed layers?
  - Any layer producing 3D hits (either barrel or disk)
  - Now: pixel layers the directly produce 3D hits
  - Future: stereo strip layers (need to add ghost hits)

Confirming the Seed

- In general, hit combinatorics will generate many more seeds than actual tracks
- To quickly reduce the number of track candidates, require the seed to be confirmed in one or more layers
  - Confirmation can be done with 2D or 3D measurements
  - Confirmation layers can be either barrel or disk layers
- To confirm a seed:
  - Check that track seed intersects the confirmation layer
  - Swim the track to find the intersection
  - Search for hits consistent with the seed
  - Require that the minimum number of confirmation hit are found
  - Check that the seed hasn't already been found
- If a new seed is confirmed, the track helix is recalculated using the confirming hits

Extending the Seed

- Look for additional hits that can be added to the track
- Swim track to extension layer and look for hits consistent with the track
- If a new hit is added, re-fit the helix
- If multiple hits are found, follow both branches
- When all extension layers have been searched:
  - Require a minimum number of extension hits
  - Eliminate duplicate/fake tracks when multiple track candidates share hits

SeedTracker Status

- Expect to release first version shortly
  - First version will find and confirm seeds
  - Can be used to with other track-finding software that requires a seed track
- Future updates will provide additional functionality
  - Extend track seed to outer layers
  - Incorporate multiple scattering errors
  - Implement 3D hits from stereo strip layers (including ghost hits)

Summary

- SeedTracker developed in org.lcsim framework to help in the optimization of the SiD tracker design
- Flexible design to allow study of tracker variations without modifying tracking code or re-tuning parameters
- User can control all parameters that direct the tracking strategies
- First version that finds and confirms track seeds will be released shortly