Table of Contents

Tracking strategies used in LHCb

Introduction

Tracks are found by different tracking strategies. A tracking strategy is mainly concerned with how tracks are found in the different subdetectors in general, while tracking algorithms specify how tracks are found in detail. This page attempts to give an overview of the different tracking strategies, the algorithms that implement them and the types of tracks resulting from their application. It also lists the input required by algorithms implementing a strategy.

The order in which strategies are discussed respects dependencies among the strategies, so we will not discuss a strategy before we explain where its input comes from. This means that one has to start with the strategies and algorithms that reconstruct tracks from hits rather than those which already need tracks as input. Furthermore, all strategies have been placed in different categories, depending on the type of tracks they produce.

Once a track is found, it is fitted with a Kalman filter. See LHCbTrackFitting for details.

As a track may be found by different algorithms, or segments of the same track might be available in different track containers on the TES, so there is a clone killing stage which collects tracks from the various locations, selects the best candidate if there are several (whatever that may mean in practice), and writes the resulting combined collection to an output container, per default Rec/Track/Best. This is usually what one uses in the end for physics analyses. The clone killer is described in more detail in section Clone Killer near the end of this page.

This page is organized as follows:

Track types

Track types available in LHCb are:

- VeloR track (2D Velo track in RZ projection)
- Velo track (full 3D Velo track)
- T track
- Upstream track
- Downsteam track
- Long track

The following sketch gives you a more graphical impression:

LHCb track types

Strategies and Algorithms

Velo Tracking

This category contains strategies to find tracks in the Velo (See also this page.).

Velo tracking in RZ

Method: Starting from hits in Velo R sensors, form 2D Velo tracks by assuming all tracks originate in the same point.

Input: Velo R hits

Output: RZ Velo tracks

Default Location in TES: Rec/Track/RZVelo

Algorithms:

- PatVeloRTracking (see also VeloRTracking)

Velo space tracking

Method: Starting from 2D Velo tracks in RZ projection and hits in the Phi sensors of the Velo, this strategy will produce full 3D Velo tracks, again under the vertex assumption.

Input: RZ Velo tracks, Velo Phi hits

Output: 3D Velo tracks

Default Location in TES: Rec/Track/Velo

Algorithms:

- PatVeloSpaceTracking (see also VeloSpaceTracking)

General Velo tracking

Method: This strategy aims to find 3D tracks in the Velo without assuming anything about the origin of the tracks. While being more general, algorithms in this category tend to be slower than the combination of Velo tracking in RZ and Velo space tracking.

Input: Velo hits in both R and Phi sensors

Output: 3D Velo tracks

Default Location in TES: Rec/Track/Velo

Algorithms:

- PatVeloOpenTracking: special algorithm for reconstructing tracks without assumptions about track parameters (e.g. to reconstruct with the Velo in open position), requires hits in consecutive sensors

- PatVeloGeneric: general purpose reconstruction, no assumptions about track parameters, produces very clean track sample for alignment and test beam studies
- PatVeloGeneralTracking: can run on unused tracks after PatVeloRTracking and PatVeloSpaceTracking to recover some K_s, halo and beam gas tracks
- FastVelo: the velo tracking used since 2011

Standalone T station reconstruction

This category discusses how to find T station tracks without using information from other subdetectors.

Standalone T station reconstruction (combinatorial approach)

Method: Track reconstruction works be choosing strategically combinations of very few T station hits from which a track hypotesis is derived. Then, algorithms collect hits in a window around this hypothesis. The resulting track candidates have to satisfy a number cuts before a hypothesis is accepted as track.

Input: T station hits

Output: T station tracks

Default Location in TES: Rec/Track/Seed

Algorithms:

- PatSeeding (default standalone T station pattern reco since 2011, has special tunings for magnet off, HLT usage, finding cosmics or to run with misaligned detector)
- TsaSeeding (used to be the default for data taken in 2010 and before, has been in LHCb software for a long time, tunings for magnetic field off or HLT usage)
- TrackSeedFind (seems to be rarely used nowadays, can't say much)

Long track reconstruction

This category describes strategies to find long tracks.

Forward tracking

Method: Starting from seeds in the Velo, tracks are searched in the T stations by parametrizing the expected position in T as function of the Velo seed track parameters and the position of a single hit in T. Further T station hits in a window around the expected position in different stations and layers are picked up. If the combination of a Velo seed and some T station hits satisfies some quality cuts, it is promoted to a long track. Hits in TT are picked up if they are close enough to a track through Velo and T station hits.

Input: 3D Velo tracks as seeds, hits in T and TT

Output: Long tracks

Default Location in TES: Rec/Track/Forward

Algorithms:

- PatForward

Track matching

Method: Starting from a set of tracks reconstructed in the Velo and a second set reconstructed in the T stations, track matching attempts to match the tracks in the two sets to one another. This is done by extrapolating both Velo and T station track segments to the bending plane of the magnet and evaluating various quantities (e.g. position in the bending plane, slope change, number of compatible hits in TT) to determine if a Velo and a T track segment do actually belong together. TT hits close to the resulting tracks are added afterwards.

Input: 3D Velo tracks, T station tracks, TT hits

Output: Long tracks

Default Location in TES: Rec/Track/Match

Algorithms:

- TrackMatchVeloSeed (focus is on efficiency and few ghost tracks)
- PatMatch (focus is mainly on speed)

Downstream tracking

Method: Using tracks in the T stations, algorithms implementing this strategy search for matching TT hits.

Input: T station tracks, TT hits

Output: Downstream tracks

Default Location in TES: Rec/Track/Downstream

Algorithms:

- PatDownstream

Upstream tracking

Method: Starting from seed tracks in the Velo, upstream tracks are constructed by adding matching TT hits

Input: 3D Velo tracks, TT hits

Output: Upstream tracks

Default Location in TES: Rec/Track/VeloTT

Algorithms:

- PatVeloTT

Clone Killer

Method: By looking at tracks more closely, a clone killer can check if two tracks or segments of tracks are duplicates of each other and chose the "better" track to append it to its output.

Note: some of the algorithms in this section will copy non-clone tracks to an output container, some will only flag clones without creating track containers or modifying the track content of existing containers

Input: various track types

Output: various track types

Default Location in TES: Rec/Track/Best

Algorithms:

- TrackEventCloneKiller: This is the standard clone killer run in our reconstruction jobs used until Reco13. It removes clones by looking at the hit content of tracks; among two clones, the track with more hits is saved to the output location.
- TrackBestTrackCreator: This is the standard clone killer run in our reconstruction jobs since Reco13a. FIXME link to wouter's slides.
- TrackBuildCloneTable/TrackCloneCleaner: The combination of these algorithms flags clones by comparing track parameters and flagging the lower quality ones as clones for subsequent analysis by adding an info field to the track. This allows to e.g. apply successively harder cuts on clones during an analysis.
