LC-DET-2012-071
http://www-flc.desy.de/lcnotes

Executive Summary: Recent Software Developments for Time Projection Chambers

Christoph Rosemann*, DESY, on behalf of the MarlinTPC working group:
Jason Abernathy[a], Li Bo[b], Christoph Brezina[c], Stefano Caiazza[d], Patrick Conley[a], Klaus Dehmelt[e], Ralf Diener[d], Jan Engels[d], Keisuke Fujii[f], Frank Gaede[d], Christian Grefe[b], Isa Heinze[d], Katsumasa Ikematsu[g], Matthias Enno Janssen[d], Jochen Kaminski[c], Dean Karlen[a], Yuikihiro Kato[h], Martin Killenberg[b], Claus Kleinwort[d], Frederik Klöckner[c], Thorsten Krautscheid[c], Christoph Krieger[c], Felix Müller[d], Astrid Münnich[d], Oliver Schäfer[i], Daizo Sakuragi[h], Steven Turnbull[a], Adrian Vogel[c], Peter Wienemann[c], Ryo Yonamine[f], Simone Zimmermann[c]

[a] *University of Victoria*
[b] *CERN*
[c] *University of Bonn*
[d] *DESY*
[e] *Stony Brook University*
[f] *KEK*
[g] *University of Siegen*
[h] *Kinki University*
[i] *University of Rostock*

Abstract

This note summarizes the developments of reconstruction software and software tools in use for the LCTPC collaboration since the report to the DESY PRC from 2010 [1]. It is intended as an interim note to several detailed notes to be released within 2012.

*christoph.rosemann@desy.de

1 Introduction

In one of the forming workshops of LCTPC, it was decided to have one common software framework for the reconstruction of Large Prototype (LP) data. Main reasons are the compatibility of software developments, the comparability of results and the reduced amount of work in development. The main project took shape in the form of MarlinTPC, and is extended in additional other tools. MarlinTPC is based on the common persistency framework LCIO [2], building upon the reconstruction framework MARLIN [3]. Complemented is the usage of LCCD [4] as the standard way of providing access and storage of conditions data. The geometry description is based on GEAR [5].
The chain of event reconstruction is organized in tiers. At first tier – or lowest level – the raw data is converted to LCIO format and then processed according to the type of readout system. These hardware specific parts rest in the hands of the individual groups. The middle tier is already more generic, and ends with the creation of three dimensional space points in the TPC. Each individual step is characterized by distinct types of LCIO data objects. An example for the different data types is presented in table 1, which also outlines the processing chain. The first two tiers constitute the basic reconstruction. The higher

LCIO Type	Default Collection Name
TrackerRawData	TPCRawData
TrackerData	TPCConvertedData
TrackerPulse	TPCPulses
TrackerHit	TPCHits
Track	TPCTracks

Table 1: The basic data types defined in LCIO for track reconstruction in a tracking device. The order of data types and their (default) name outline the processing chain.

reconstruction follows as last tier, which is concerned with the track level. It consists of finding and fitting tracks, as well as performing alignment, calibration and correction tasks. One of the basic characteristics of using the Marlin event framework, is that the different processing steps are organized linear chain of software modules acting on events. Data that is not event based, but based on collections of events or constant over certain times (like data runs) is provided by the conditions data handling.

2 MarlinTPC

2.1 Basic Reconstruction

Two different main lines of event reconstruction exist, namely one for pad based readout systems and one for pixelized readout systems.
For pad based data two different readout electronics exist. The first one is ALTRO, used at the ALICE experiment at CERN; and the second one is AFTER, based on the readout

system for the T2K experiment in Japan. Both are similar in the regard, that channel wise ADC spectra in time are recorded. In each channel signals are sought and extracted as so called *pulses*, still a charge spectrum associated with a certain channel. These signals are then mapped to the geometry description of the readout plane and then finally combined to a distributed charge deposition in each readout row. The position in time is converted by multiplication with the drift velocity into a position along the drift axis, which is the eponymous part for the time projection chamber. These charge depositions are then the 3D space points with an integrated charge; where all properties are calculated from the individual channel wise signals.
Pixelized readouts can resolve individual charge clusters, which are spread out in the measurement plane. Reconstruction of this clusters require a 2D clustering procedure. An additional complication arises from the fact, that current readout chips only record either charge or time per pixel, which requires interpolation between pixels. As final step the space points are then calculated from the reconstructed clusters.

Status of Basic Reconstruction The different algorithms have been extensively tested and many changes and improvements have been applied. All measurement values can be attributed with errors, including the final error propagation to the properties of the space points. This part is under constant review, modification and extension. It is used within all working groups and constitutes the main working horse within MarlinTPC.

2.2 Higher Reconstruction

The higher reconstruction deals mainly with the track building. This consists of track finding and fitting, with the input of the space points. The goal is to extract the best estimation of track parameters.
Due to the more complex nature of track finding and fitting, the development of dedicated finding and fitting algorithms within MarlinTPC had been abandoned. Some effort was initially put into the implementation of track finding and fitting for single track events, which is still available. It was common agreement, that the best practice is to use and incorporate external packages for the actual track reconstruction.
For track finding two options present themselves:

1. PathFinder[6], a dedicated global pattern recognition program for tracks with or without magnetic field. It is based on Hough Transformations.

2. Clupatra[7], a local pattern recognition program. It is based on searching for nearest neighbours.

For track fitting two different choices are currently available:

1. For helicoidal tracks:
   The KalTest [8] library, which is based on ROOT [9] and provides basic C++ classes

for track fitting with Kalman filter technique. The library allows to handle measurement layers of various shapes and coordinate systems in a unified way, while minimizing user-implemented code. The library is also in use for the ILD reconstruction, based on simulation.

2. For straight line and helicoidal tracks:
   General Broken Lines [10], which has recently been added. It does a refit of previously found tracks and provides a direct interface to Millipede-II [11], which can be used for extraction of (mis-)alignment and calibration parameters and distortions.

A dedicated processor exists, which lies in the middle between track finding and fitting. It combines the KalTest library with a combinatorial Kalman Filter.

Status of Higher Reconstruction The different software packages are in the hands of the respective developers. They all are used in different contexts as well, and all teams are still actively developing. PathFinder has been used for reconstruction of test beam data from the LP, as well as in other circumstances. Clupatra has been heavily tested within the standard reconstruction chain of the ILD simulation. KalTest is used in the ILD simulation and for the track reconstruction of LP data.
The extraction of (mis-)alignment, or the application of correction due to distortions is not yet publicly available within the framework.

2.3 Laser calibration

The large prototype has a prepared aluminium cathode with a certain point pattern, where the copper anodization is removed. An UV laser setup can release electrons, which drift through the field of the whole TPC from cathode to anode. Data taken with and without magnetic field can be used to extract field distortions. A set of special reconstruction modules, including a simulation had been written to perform the task.

Status of Laser calibration Unfortunately the initial work was neither finished nor properly published, apart from a work report [12]. The software parts that were written are still available within MarlinTPC. Work recently has restarted to analyze laser calibration data and extract the distortions.

2.4 Conditions Data

Another large contribution concerns the non-event related data. The MARLIN framework has an abstract interface (LCCD) to an implementation, which runs in reality as a MySQL database. Within the processing chain the data can be accessed by any software module via a special method. The data are identified by time stamps, so that most of the matching

between event and non-event data is done automatically by comparing times. In MarlinTPC reconstruction it is agreed, that this is the standard way to provide all non-event related data like field settings, drift velocity, readout frequency etc. Of course, in private programs or test runs this way of data access can usually be overridden by explicit setting of values, which also allows processing without database access.
The actual data objects which are stored, need to be user defined and are built from basic variable types like integers, floats and strings. The set of data which is needed for event reconstruction in the LP is defined in MarlinTPC, which means that these objects are formally described in classes which reside in the MarlinTPC code.. Conditions objects often require some kind of processing, but in any case the user needs to push them actively to the database.

Status of Conditions Data A complete test system of database servers is set up at DESY, which is already in use for several reconstructions. The definition of condition objects has been deeply revised and extended. Several processing steps have been improved, that they automatically push the calculated values and conditions objects into the database. One of the biggest contributions of practical use is the magnetic field map of the PCMag, which is stored as conditions object.

3 Tools for Test beam campaigns

Several smaller programs and tools within and elsewhere of MarlinTPC have been built. The EUDET data acquisition (eudaq [13]) package provides the direct writing of LCIO events from the raw data acquisition; which is implemented (and in use) for the pixelized readout with TimePix. The conversion from ALTRO raw format to LCIO has become part of MarlinTPC, available for all groups that use the ALTRO readout electronics. CEDViewer[14], the event display application for the ILD simulation, has been extended to display also the raw (LCIO) data of ADC spectra. Thus it is possible to overlay all levels of data (raw, pulses, space points, tracks, etc.) and view them at the same time. A program to create the mapping files from hardware channel to geometry ids is available. And an extensive test beam analysis package for pad based data is available.

4 General Organization and Information

In total about 30 people contributed so far to MarlinTPC, recently active are 10 developers. The code itself resides in a svn repository [15], and provides working branches for the developers and the common development (called *trunk*). It also provides the means for software releases, which are included in the general ILCSoft installation [16].
There is a project homepage [17], which is a wiki page that allows constant editing and updating of information. In addition there is a mailing list [18] and a discussion forum [19]. To complete the software communication, there is also a bug tracker [20]. There are

also telephone/EVO meetings, where current issues are discussed and recent developments presented with regular participation from Europe, Asia and Canada.

References

[1] LC-DET-2012-067 Report to the Desy PRC 2010, *LCTPC Collaboration*

[2] LCIO Homepage: http://lcio.desy.de

[3] MARLIN Homepage: http://ilcsoft.desy.de/marlin

[4] LCCD Homepage: http://ilcsoft.desy.de/lccd

[5] GEAR Homepage: http://ilcsoft.desy.de/gear

[6] http://www-flc.desy.de/flc/flcwiki/PATHFINDER

[7] https://svnsrv.desy.de/public/marlinreco/Clupatra

[8] KalTest Homepage: http://www-jlc.kek.jp/subg/offl/kaltest/

[9] RooT Homepage: http://root.cern.ch

[10] General Broken Lines as advanced track fitting method. C. Kleinwort, NIM A, 673 (2012), 107-110, doi:10.1016/j.nima.2012.01.024
http://www.wiki.terascale.de/index.php/GeneralBrokenLines

[11] Millepede-II. V. Blobel, NIM A, 566 (2006), pp. 5-13
http://www.wiki.terascale.de/index.php/Millepede_II

[12] http://ilcagenda.linearcollider.org/getFile.py/access?contribId=1 /
&resId=0&materialId=slides&confId=4794

[13] http://projects.hepforge.org/eudaq/

[14] http://ilcsoft.desy.de/portal/software_packages/cedviewer/

[15] MarlinTPC repository (public access): https://svnsrv.desy.de/public/marlintpc/

[16] ilcinstall Homepage: http://ilcsoft.desy.de/ilcinstall

[17] MarlinTPC Homepage: https://znwiki3.ifh.de/MarlinTPC/

[18] MarlinTPC mailing list: ilcsoft-marlintpc@desy.de

[19] Linear Collider Forum, sub forum *Tracking & Vertexing*:
http://forum.linearcollider.org/index.php?t=threadt&frm_id=39

[20] Software issue tracker: https://jira.slac.stanford.edu/browse/MAR