Hands-On:
Starting with EUTelescope
- no or passive DUT.

Jan-Hendrik Arling

Cyril Becot, Jan Dreyling-Eschweiler,

Michaela Queitsch-Maitland, Edoardo Rossi

7th Beam Telescopes and Test Beams Workshop

What is EUTelescope?

- EUTelescope is a framework for reconstructing and analysing data taken with pixel beam telescopes.
- Implemented as part of the EUDET-project:
  1. Hardware → Mimosa26 beam telescopes.
  2. DAQ software → EUDAQ.
  3. Reconstruction software → EUTelescope.
- Integrated within the ILCsoft (ILC software) framework.

Outline.

- **Introduction to EUTelescope**

  What is the idea of EUTelescope? What is the base software structure? Who is developing?

- **Reconstruction workflow**

  Which are the parts of an EUTelescope analysis? How is the reconstruction chain?

- **Hands-On: The installation**

  What are the different ways for getting EUTelescope?

- **Hands-On: The GBL noDUT example**

  Reconstructing data from only telescope data using GBL.

- **Hands-On: The GBL SUT example**

  Measuring scattering angles for a passive DUT using GBL.

Caveats.

- We will tackle in this tutorial the basics of EUTelescope by analysing only passive DUTs.
- The next step is then to analyse data testbeam data with an active DUT → next tutorial.
- We try to show you as best as possible how you can use EUTelescope for your own use. If we cannot answer your questions directly, please forgive us :)
- In case of any further question: please use the issue tracker on GitHub:

  https://github.com/eutelescope/eutelescope/issues

Introduction to EUTelescope.

EUTelescope...

- has the main goal to get from raw data to high level objects like tracks crossing through the beam telescope.
- is embedded in the ILCsoft (ILC software) framework.
- is built in the Marlin (Modular Analysis & Reconstruction for the linear collider) software package.
- provides processors for different reconstruction steps, which can be daisy-chained (modular structure).
- uses the LCIO (Linear Collider I/O) data model.
- uses GEAR (Geometry API for Reconstruction) as markup language describing the telescope geometry.
- is a user-driven software project, implementation of new processors for (new) applications/devices by the user groups.

EUTelescope reconstruction workflow.

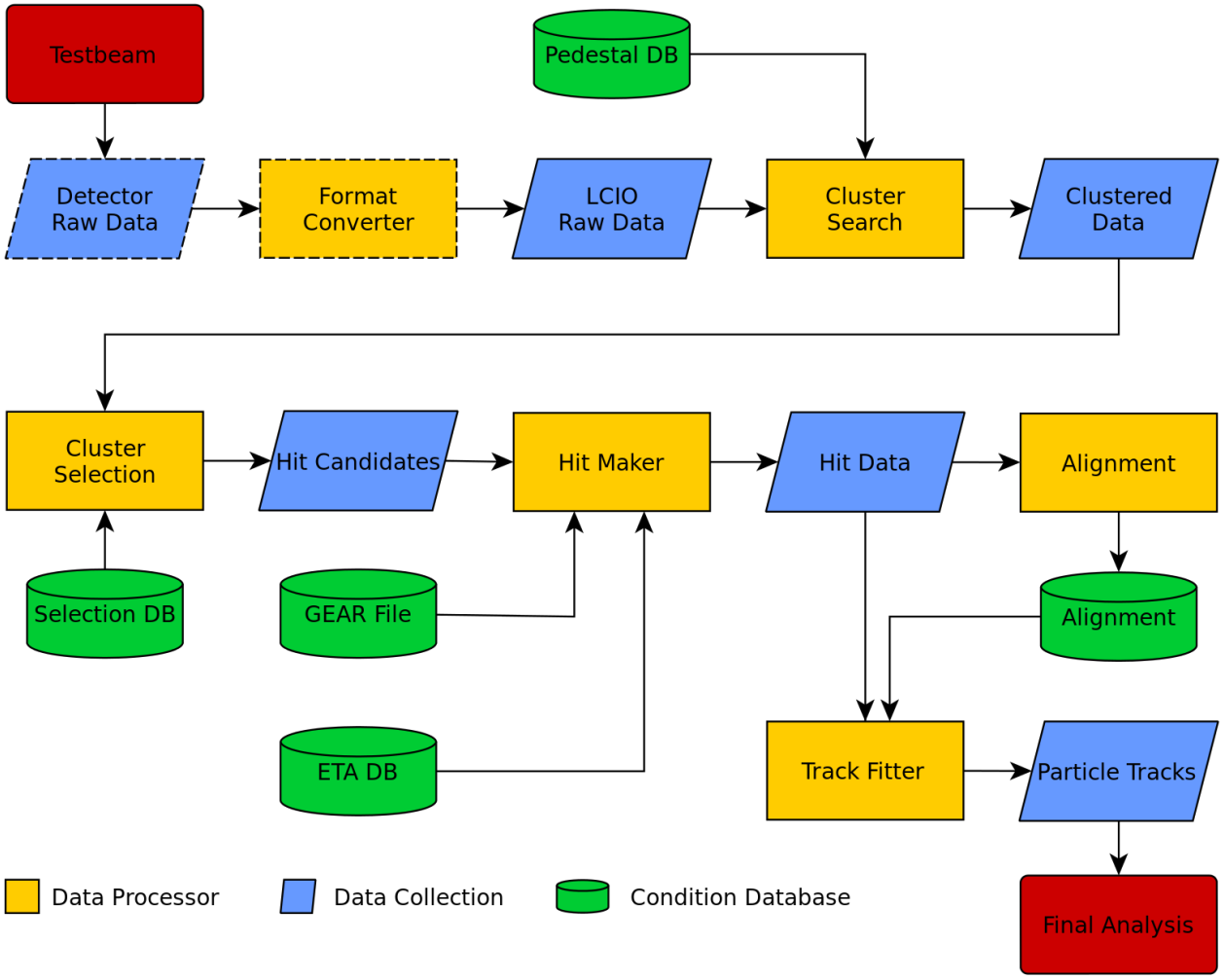
Analysis components.

- The analysis is controlled by a config file (*.cfg), a runlist csv table (*.csv), and steering file templates (*.xml).
  - The config file specifies the configuration for the individual processors, i.e. setting processor parameters.
  - The csv table contains a list of runs and associated parameters specific to that run, e.g. beam energy.
  - Steering files defines the reconstruction step by controlling which processors/options are executed by Marlin.
- The telescope geometry is provided by a gear file (*.xml) describing the telescope planes as well as the DUTs.
- Execution of reconstruction step by jobsub command.

```
jobsub --config config.cfg --csv runlist.csv [options] PROCESSOR RUNNR
```

Typical reconstruction workflow.

- From raw data to telescope tracks...



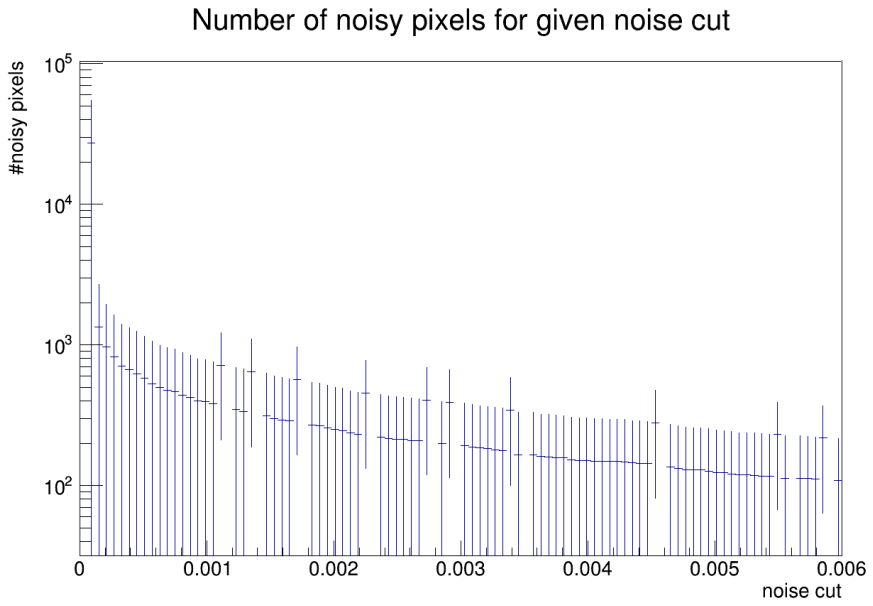
Step: CONVERTER.

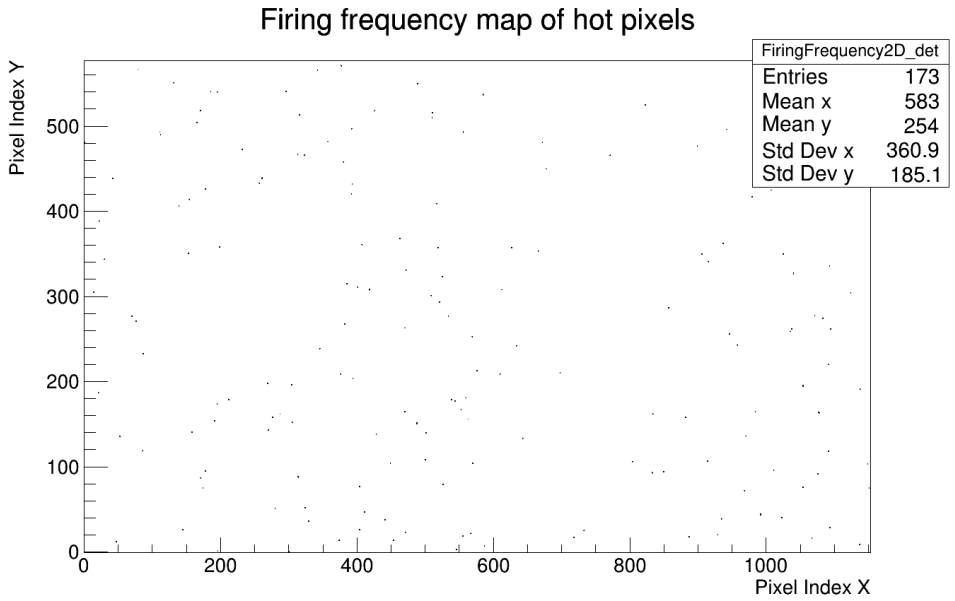
- Convert data from EUDAQ raw-format to LCIO data format.
- Only needed for EUDAQ1 testbeam data.
- Use EUTelNativeReader as interface to EUDAQ.
- Creates LCIO file with zero-suppressed data of telescope planes.

- For EUDAQ2, external converter inside EUDAQ2 package (called euCliConverter). Preprocessing from raw to LCIO.

Step: NOISYPIXEL.

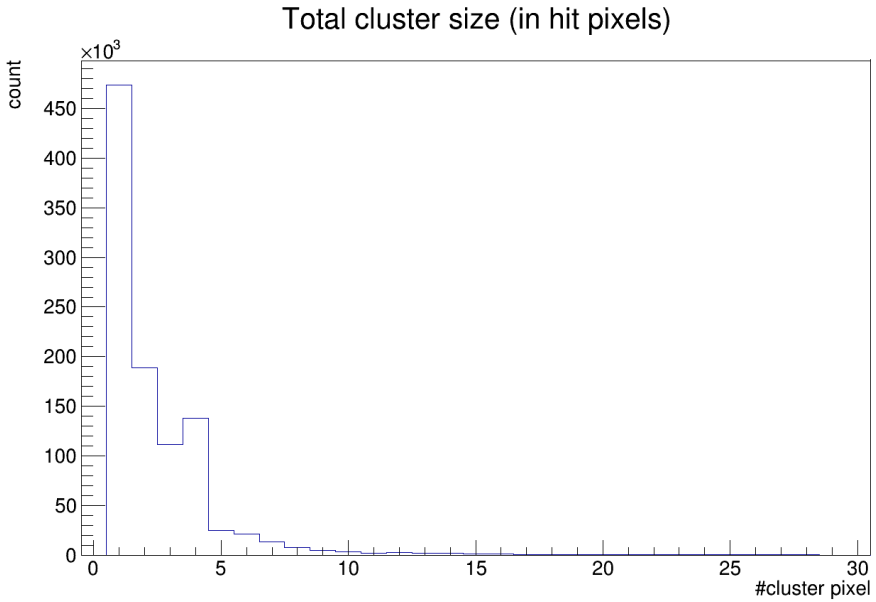
- Identification of noisy pixel by applying cut on firing frequency of individual pixel in the planes.
- Only masking by storing in database file.

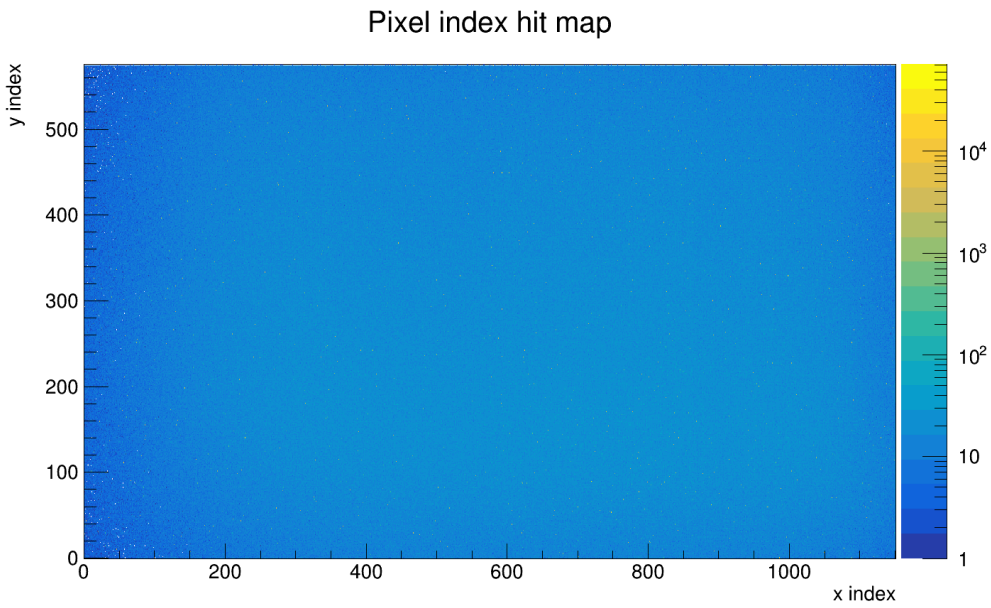




Step: CLUSTERING.

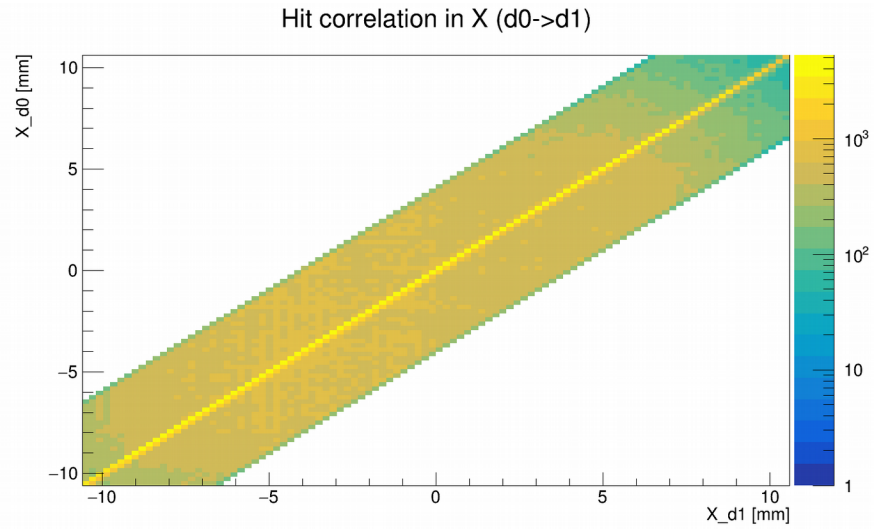
- Applying sparse clustering to find clusters out of adjacent hits.
- Use noisy pixel database to mask and remove noisy clusters from the data.
- Save noise-free clusters in LCIO.

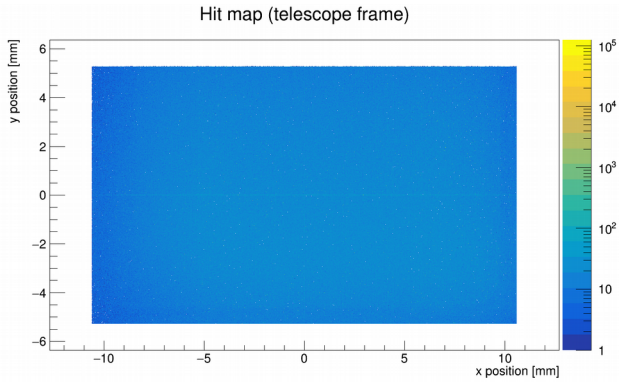


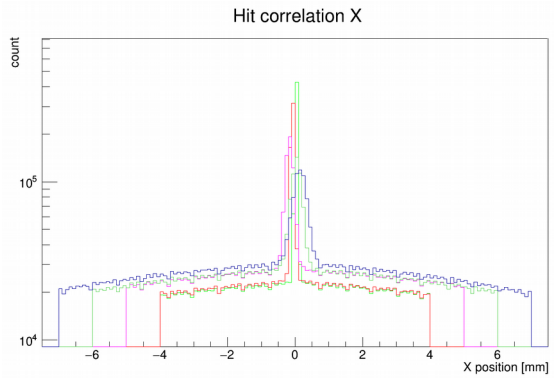


Step: HITMAKER.

- Creation of hits in local coordinate system out of clusters.
- Transform coordinates in global coordinate system using GEAR file.
- Apply prealignment by fixing one plane, write out prealigned geometry as new GEAR file.
- Evaluate correlation of planes.
- Save local hit collection as LCIO.

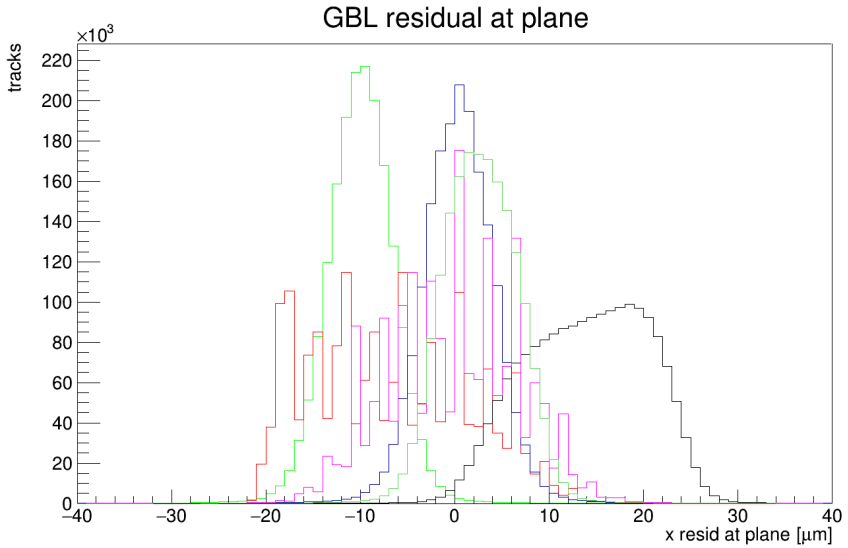


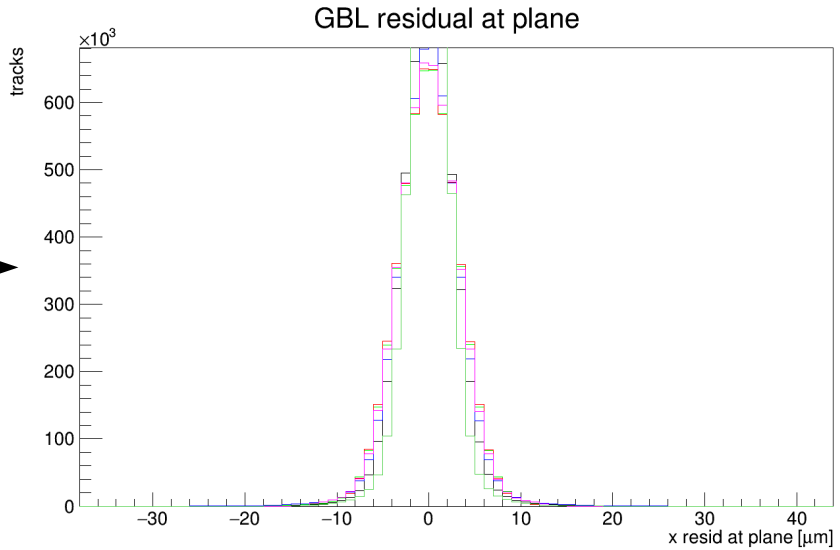




Step: ALIGNGBL.

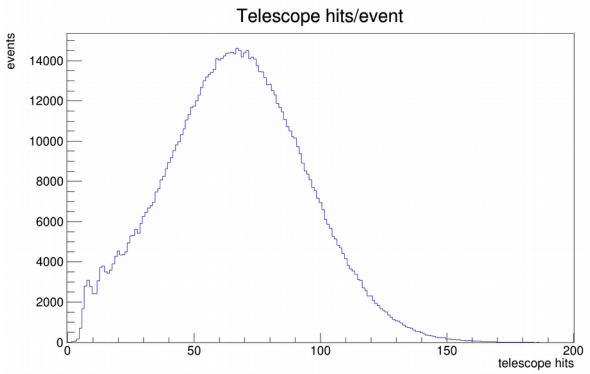
- Transform local hits to global coordinate system by loading GEAR file.
- Setting up MillePedeII for creating new GEAR file.
- Use GBL to perform alignment of planes: build upstream/downstream triplets and best match them; different alignModes for GBL algorithm.
- Write out new GEAR file with alignement parameters.
- Iterate the process for better alignment. Cuts for next iteration suggestible by fitting (triplet) distributions.

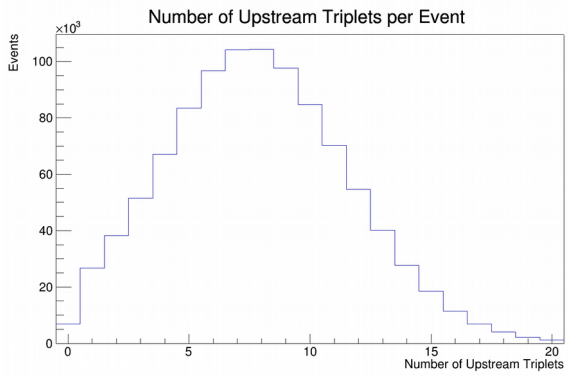


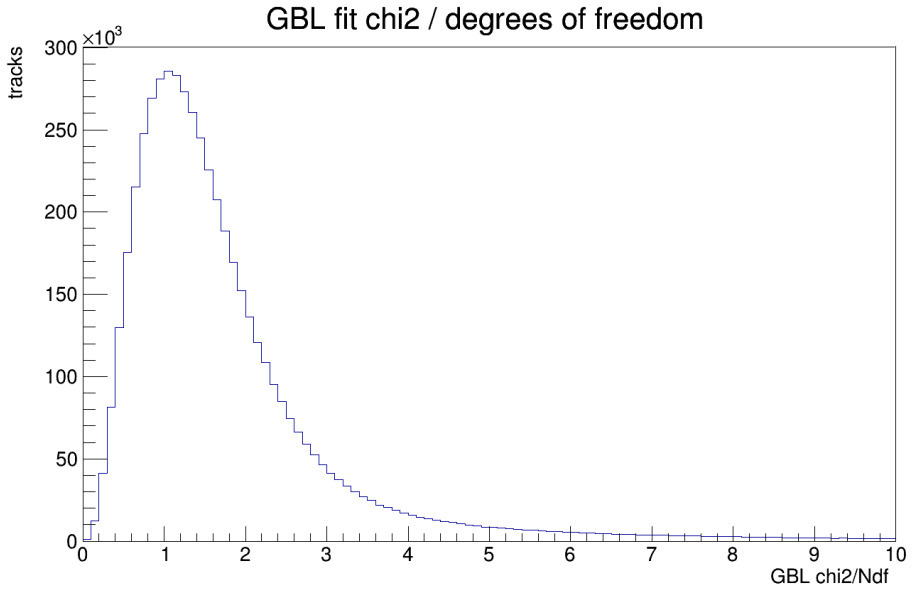


Step: FITGBL.

- Transform local hits to global coordinate system by loading GEAR file.
- Fit telescope tracks by using GBL algorithm.
- Write out tracks and other variables to ROOT ntuple.







Hands-On:
The installation.

How-to installation.

http://eutelescope.web.cern.ch/content/installation

(1) Installation on personal laptop

i. Prerequisites I: packet installation

Install all packages required for the compilation of the ILCsoft packages and of EUTelescope (dependent on distribution).

ii. Prerequisites II: prepare installer

```
export ILCSOFT=/path/where/you/want/to/install
mkdir -p $ILCSOFT
cd $ILCSOFT
git clone -b dev-base https://github.com/eutelescope/ilcinstall
```

iii. Running installation (different releases available)

```
cd $ILCSOFT/ilcinstall
./ilcsoft-install -i examples/eutelescope/release-standalone.cfg
```

iv. Source environment

```
cd $ILCSOFT/v01-19-02/Eutelescope/master
source build_env.sh
```

How-to installation.

https://github.com/eutelescope/eutelescope/wiki/Installation-NAF

(2) Installation on NAF

i. Login to NAF and choose installation location

```
export ILCSOFT=/nfs/dust/atlas/user/USERNAME/ilcsoft
mkdir -p $ILCSOFT
```

ii. Prepare environment

```
scl enable devtoolset-4 rh-git29 bash
```

iii. Get installer

```
cd $ILCSOFT
git clone -b dev-desynaf https://github.com/eutelescope/ilcinstall
```

iv. Run the installation

```
cd ilcinstall
./ilcsoft-install -i examples/eutelescope/release-standalone.cfg
```

v. Source environment

```
cd $ILCSOFT/v01-19-02/Eutelescope/master
source build_env.sh
```

How-to installation.

(1+2)

i. Change branch to BTTB7-version*

```
git fetch origin
git checkout origin/BTTB7 -b BTTB7
```

ii. Rebuild EUTelescope

```
cd $EUTELESCOPE/build
cmake ..
make install
```

*After the tutorial, we will tag the version used in the tutorial, to provide you a good reference back.

How-to installation.

(3) Docker (personal machine)

i. Pull image

```
docker pull eutelescope/bttb_tutorial_jan2019
```

ii. Run image

```
docker run -i -v <INPUTDIRECTORY>:/dummy -v
<OUTPUTDIRECTORY>:/output -t eutelescope/bttb_tutorial_jan2019
```

iii. Environment sourced already

How-to installation.

(4) Singularity (LXplus)

i. Login to LXplus

ii. Make folder to save image (storage space)

mkdir /afs/cern.ch/work/<u>/<user>/.singularity

ln -s /afs/cern.ch/work/<u>/<user>/.singularity

iii. Pull docker image

singularity pull docker://eutelescope/bttb_tutorial_jan2019

iv. Run and bind image

singularity shell --writable --contain --bind /afs:/dummy --bind <OUTPUTDIRECTORY>:/output docker://eutelescope/bttb_tutorial_jan2019

v. Source environment

cd /ilcsoft/v01-19-02/Eutelescope/master/

source build_env.sh

How-to installation.

(5) AFS sourcing (can be slow)

i. Login to machine with AFS access (e.g. NAF or LXplus)

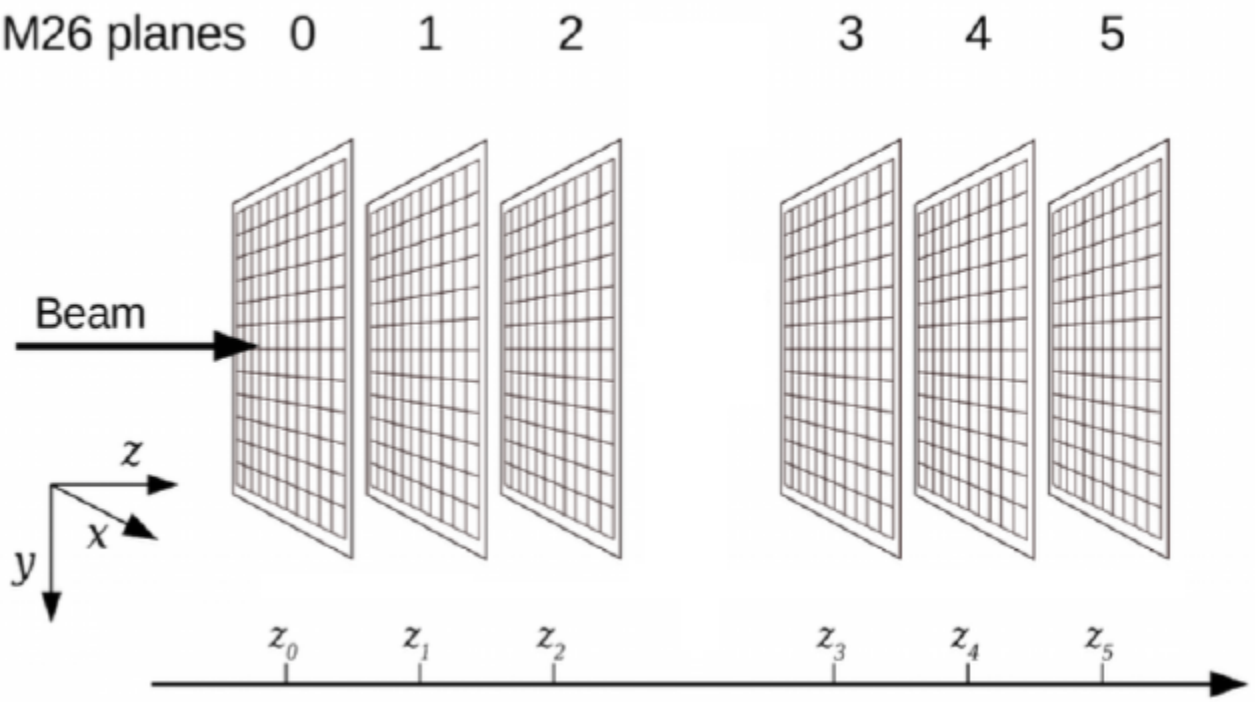
ii. Source EUTelescope environment

source /afs/desy.de/user/a/arling/public/ilcsoft/v01-19-02/Eutelescope/master/build_env.sh

Hands-On:
The GBL noDUT example.

Empty telescope.

- Telescope consisting of six Mimosa26 planes, no DUT inserted.
- Use General Broken Line (GBL) algorithm for alignment and fitting.



Setting up analysis.

- Create analysis folder:

  mkdir -p <USERPATH>/analysis_noDUT
  cd <USERPATH>/analysis_noDUT

- Copy example files:

  cp -r $EUTELESCOPE/jobsub/examples/GBL_noDUT/* .

- Change in config-file:

  BasePath = <USERPATH>/analysis_noDUT
  NativePath = <PATHTOINPUTFILES> (if no AFS access)

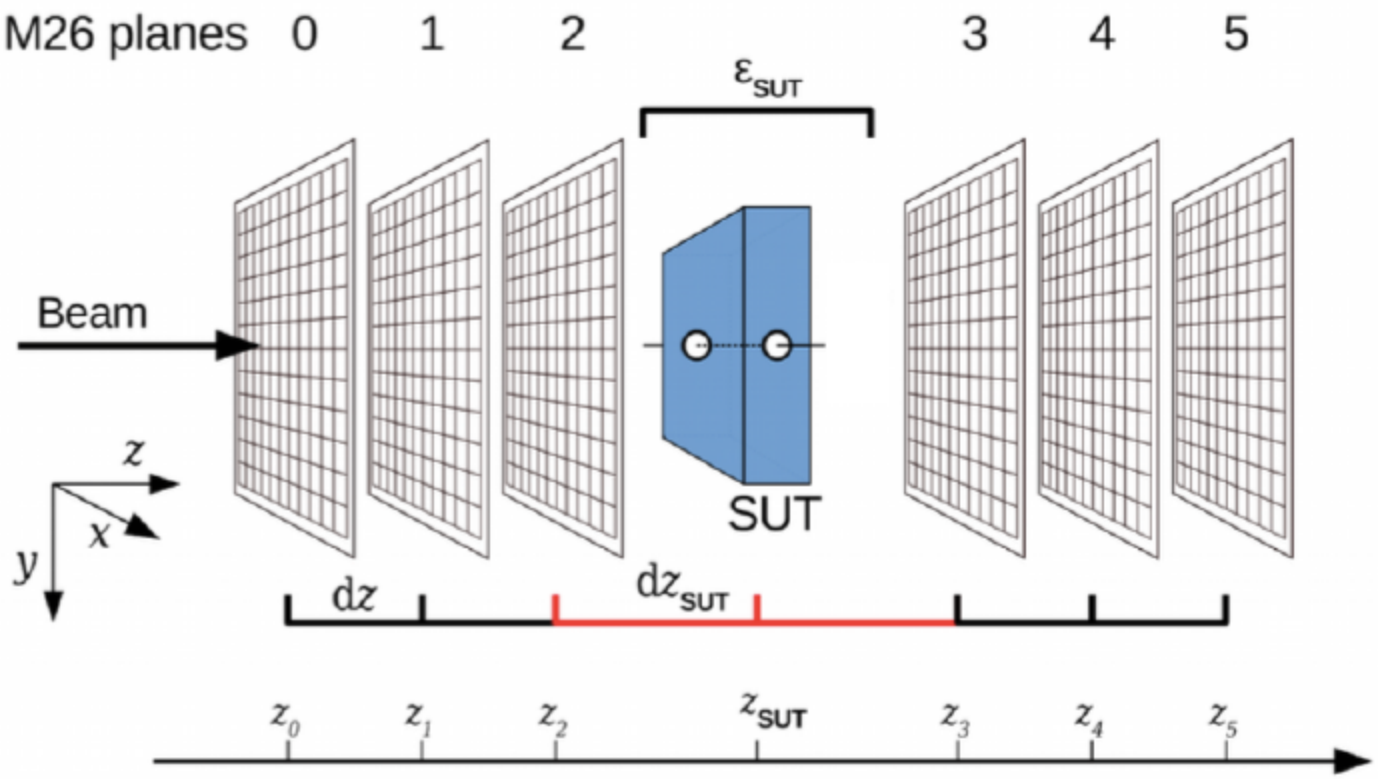
- Run step-by-step reconstruction:

  jobsub -c config_DESY2018.cfg -csv runlist.csv -g noisypixel 701
  jobsub -c config_DESY2018.cfg -csv runlist.csv -g clustering 701
  jobsub -c config_DESY2018.cfg -csv runlist.csv -g hitmaker 701
  jobsub -c config_DESY2018.cfg -csv runlist.csv -g alignGBL 701
  jobsub -c config_DESY2018.cfg -csv runlist.csv -g alignGBL2 701
  jobsub -c config_DESY2018.cfg -csv runlist.csv -g alignGBL3 701
  jobsub -c config_DESY2018.cfg -csv runlist.csv -g fitGBL 701

Hands-On:
The GBL SUT example.

Telescope with SUT.

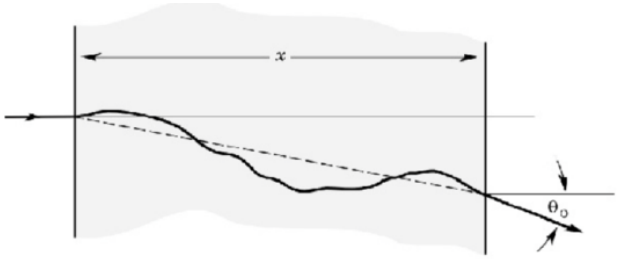
- Six Mimosa26 planes with an inserted scatterer (SUT).
- Use General Broken Line (GBL) algorithm for alignment and fitting.

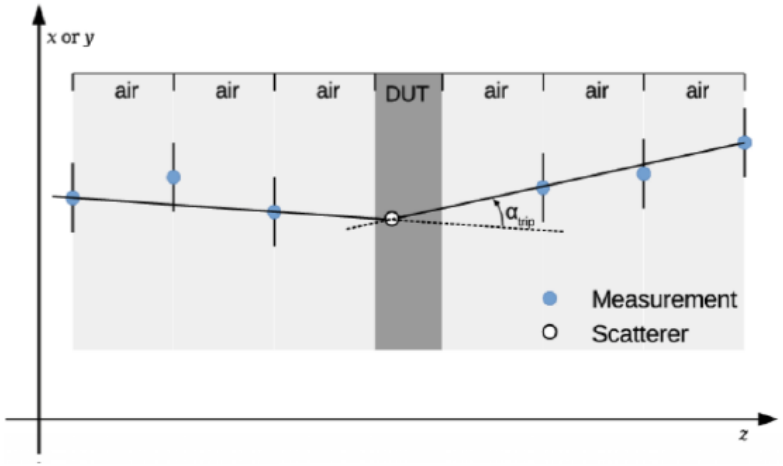


Idea: Material budget imaging.

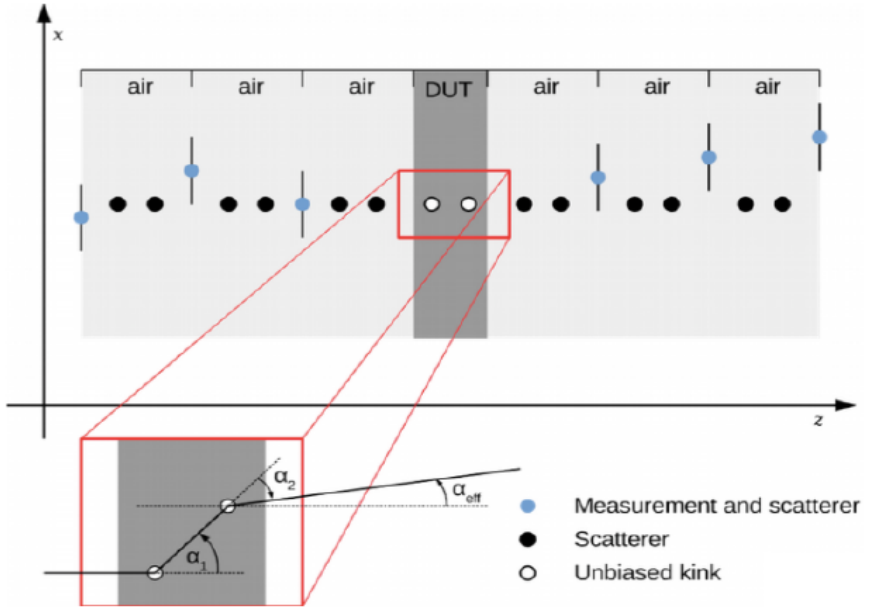
- Multiple scattering inside material dependent on material budget (X/X0) results in effective kink angle for each particle track.

  → Reconstruct kink angles as estimator of X/X0





GBL



Setting up analysis: Alu sample.

- Create analysis folder:

  mkdir -p <USERPATH>/analysis_SUT_Alu
  cd <USERPATH>/analysis_SUT_Alu

- Copy example files:

  cp -r $EUTELESCOPE/jobsub/examples/GBL_SUT/* .

- Change in config-file:

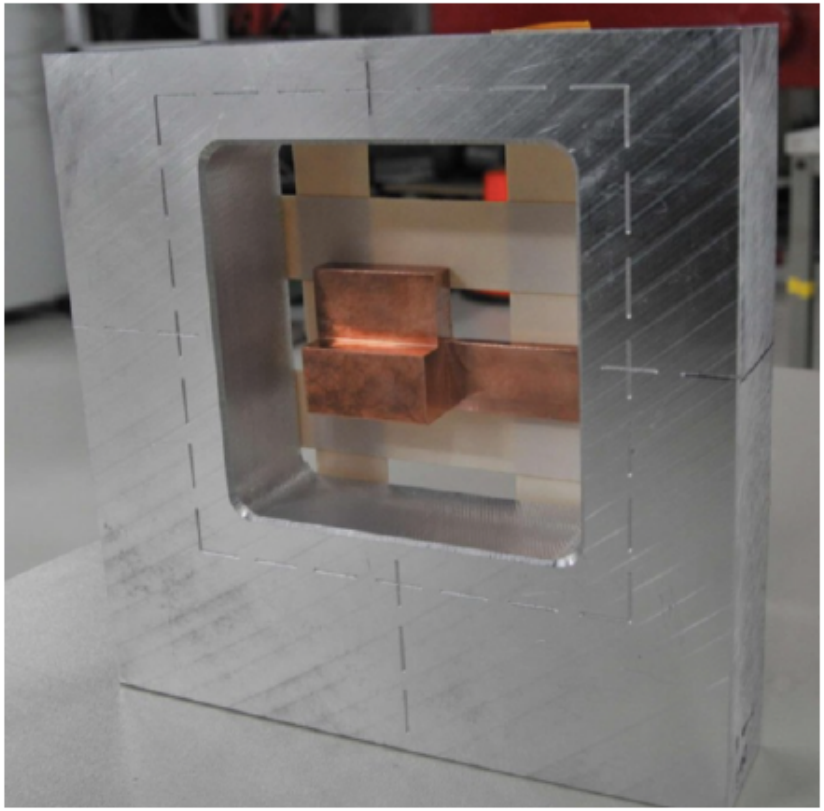
  BasePath = <USERPATH>/analysis_SUT_Alu
  NativePath = <PATHTOINPUTFILES> (if no AFS access)

- Run step-by-step reconstruction:

  jobsub -c config_AluSample.cfg -csv runlist.csv -g noisypixel 699
  jobsub -c config_AluSample.cfg -csv runlist.csv -g clustering 699
  jobsub -c config_AluSample.cfg -csv runlist.csv -g hitmaker 699
  jobsub -c config_AluSample.cfg -csv runlist.csv -g alignGBL 699
  jobsub -c config_AluSample.cfg -csv runlist.csv -g alignGBL2 699
  jobsub -c config_AluSample.cfg -csv runlist.csv -g alignGBL3 699
  jobsub -c config_AluSample.cfg -csv runlist.csv -g fitGBL 699

Setting up analysis: Cu sample.

- Repeat the same steps as before with another data set, a copper sample as SUT.
- Use now config_CuSample.cfg and the corresponding run number 681.



What else?

- Increase of events:

  So far, we were running only on a part of the data. You can increase the number of events, but especially the alignment takes then more time, but here it is sufficient to align with only a fraction.

- Check alignment:

  Use the cuts suggestion to optimize your alignment. Is it worth to add more iterations of the GBL alignment?

- Advanced computing:

  Instead of running locally, you can send your job to the batch cluster via HTCondor. This is especially helpful for parallelizing the reconstruction on more data samples. For this, check out examples/condor_submission.

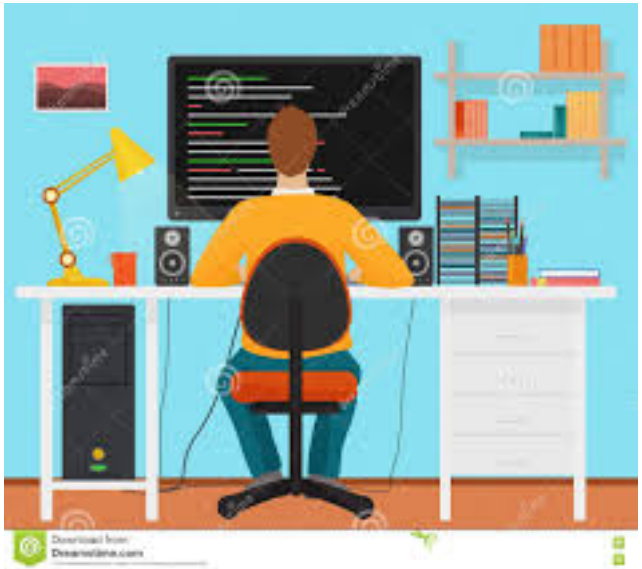
- Last but not least:

  Go active and join the next EUTelescope tutorial :)

Conclusion.

What you take home...

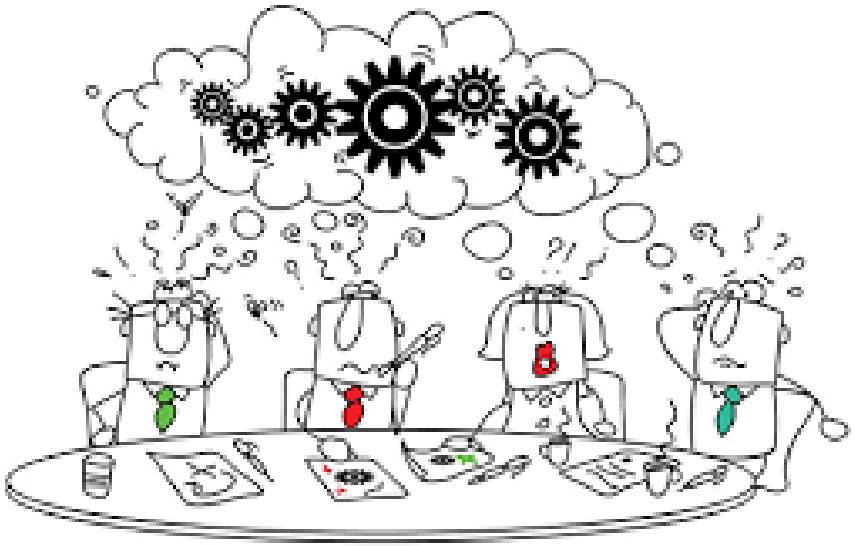
- You learned about the basic philosophy of reconstructing telescope data using the EUTelescope framework.
- You were able to install/run EUTelescope on your preferred system.
- You tested the analysis flow for an empty telescope using GBL.
- You were able to retrieve kink angle maps of upto two different SUTs.
- You familiarized yourself with the processor workflow of EUTelescope.



Source: https://de.dreamstime.com/stock-abbildung-bemannen-sie-das-hintere-arbeiten-des-programmierers-seinem-pc-computer-kodierung-und-programmierung-image80997205

What we hope...

- You think EUTelescope is usable for your testbeam analysis.
- You are eager to contribute by adding own processors for your specific use cases.
- By doing so you help to rebase EUTelescope to the standard framework for testbeam analysis using EUDET-type telescopes and even beyond. This goal we started now, but need help from anyone willing to help :)



Source: https://www.matrixmanagementinstitute.com/blog/collaborative-work-style-bring-leaders-on-board