Efficient Analysis of Test-beam Data with Corryvreckan

Vertex 2020 Conference
October 6th, 2020

Jens Kröger, Heidelberg University & CERN
on behalf of the CLICdp collaboration and the Corryvreckan Developers
Challenge:

LHC upgrades + future HEP experiments:
- stringent detector requirements
- pushing limits of technology

→ R&D on vast range of pixel detector technologies
- different readout concepts
- highly specialized to each use case

Test-beam Analysis for Pixel Sensors

Our Motivation:

- maximize synergies
  - avoid numerous similar frameworks
  - "one framework fits all"

- flexible reconstruction
  - combine detectors with different read-out schemes
  - different analysis objectives

- easy to use, understand, contribute
The Corryvreckan Framework

- modular structure
  - framework core
  - (user) modules for specific tasks

- highly flexible and configurable
  - TOML style = easy to read
  - support of physical units (e.g. 25um)

- clean code & documentation
  - modern C++, code reviews, CI
  - comprehensive user manual (> 100 pages!)

Some Notable Features:

- 4D tracking (spatial + time cuts)
- various alignment methods (track $\chi^2$, Millepede)
- General Broken Line (GBL) track reconstruction (multiple Coulomb scattering)
- **EUDAQ** integration [https://github.com/eudaq/eudaq/](https://github.com/eudaq/eudaq/)
  - include AIDA TLU as auxiliary device
  - process data recorded with EUDAQ2 DAQ
- job submission tool (HTCondor etc.)
- read in simulated data from **Allpix Squared** [https://cern.ch/allpix-squared](https://cern.ch/allpix-squared)
Combining Different Readout Schemes

- **event building:**
  arrange data from different devices in "time slices" (events) for reconstruction/analysis

- **flexible:**
  combine devices with different readout schemes
  - frame-based,
  - data-driven,
  - triggered,
  - ...

→ full analysis chain event-by-event

---

**Example 1:** frame-based + data-driven detectors

**Example 2:** triggered + data-driven detectors

**Example 3:** frame-based + triggered + data-driven detectors
Corryvreckan in short

reconstruction and analysis tool
for pixel detector test beam data

• highly flexible/configurable
  – separate modules for each reconstruction/analysis step
  – many different event building options

• comprehensive documentation
  + beginner-friendly tutorials

• growing number of users
  + contributors

Learn more:

Visit our website:
https://cern.ch/corryvreckan

Browse through our manual:
→ Get the latest version here

Try our tutorials:
→ Get Started (no prior experience required)
→ Advanced (more complex use cases)

Check out the repository:
https://gitlab.cern.ch/corryvreckan/corryvreckan

Discuss in the forum:
https://corryvreckan-forum.web.cern.ch/

Contact us:
corryvreckan.info@cern.ch
https://mattermost.web.cern.ch/corryvreckan
Backup

more details and examples...
Example Publications
for which Corryvreckan has been used

- Florian Pitters:
  *Time Resolution Studies with Timepix3 Assemblies with Thin Silicon Pixel Sensors*
  JINST 14 (2019) 05, P05022
  DOI: 10.1088/1748-0221/14/05/P05022

- Mathieu Benoit:
  *Pixel detector R&D for the Compact Linear Collider*
  JINST 14 (2019) 06, C06003
  DOI: 10.1088/1748-0221/14/06/C06003

- Magdalena Munker:
  *Vertex and tracking detector R&D for CLIC*
  DOI: 10.1016/j.nima.2020.164475

- Morag Williams:
  *R&D for the CLIC Vertex and Tracking detectors*
  JINST 15 (2020) 03, C03045
  DOI: 10.1088/1748-0221/15/03/C03045

- Jens Kröger:
  *Silicon Pixel R&D for the CLIC Tracking Detector*
  JINST 15 (2020) 08, C08005
  DOI: JINST 15 (2020) 03, C03045

PhD Theses

- Florian Pitters:
  *Silicon Detector Technologies for Future Particle Collider Experiments*
  https://cds.cern.ch/record/2714709?ln=en

- Thorben Quast:
  *Qualification, Performance Validation and Fast Generative Modelling of Beam Test Calorimeter Prototypes for the CMS Calorimeter Endcap Upgrade*
  https://cds.cern.ch/record/2725040?ln=en

- Morag Williams:
  *Evaluation of Fine-Pitch Hybrid Silicon Pixel Detector Prototypes for the CLIC Vertex Detector in Laboratory and Test-Beam Measurements*
  (work-in-progress)
Configuring Corryvreckan

- TOML style = easy to read
- support of physical units (e.g. 25um)
- need 2 files:
  - configuration file: analysis parameters
  - geometry file: detector description
More Event Building Examples

- event building:
  arrange data from different devices in “time slices” (events) for reconstruction/analysis

- flexible:
  combine devices with different readout schemes
  - frame-based, data-driven, triggered, ...

→ full analysis chain event-by-event

**Example 1:** data-driven detector

**Example 2:** frame-based + data-driven detectors

**Example 3:** triggered + data-driven detectors

**Example 4:** two triggered detectors

**Example 5:** frame-based + triggered + data-driven detectors
Documentation

- **online documentation** in repo
  https://gitlab.cern.ch/corryvreckan/corryvreckan
  - every module has a README

- extensive user manual
  - full description of framework
  - installation instructions
  - “Getting started”, FAQs
  - module descriptions (fetched from repo)
  - published as CLICdp note:
    https://cds.cern.ch/record/2703012

- Doxygen code reference
  https://cern.ch/corryvreckan/reference/
  - more details on code
Documentation

- online documentation in repo
  https://gitlab.cern.ch/corryvreckan/corryvreckan
  - every module has a README

- extensive user manual
  - full description of framework
  - installation instructions
  - “Getting started”, FAQs
  - module descriptions (fetched from repo)
  - published as CLICdp note:
    https://cds.cern.ch/record/2703012

- Doxygen code reference
  https://cern.ch/corryvreckan/reference/
  - more details on code
Documentation

- **online documentation in repo**
  https://gitlab.cern.ch/corryvreckan/corryvreckan
  - every module has a README

- **extensive user manual**
  - full description of framework
  - installation instructions
  - “Getting started”, FAQs
  - module descriptions (fetched from repo)
  - published as CLICdp note:
    https://cds.cern.ch/record/2703012

- **Doxygen code reference**
  https://cern.ch/corryvreckan/reference/
  - more details on code
## GitLab Continuous Integration

- ensures compilation, formatting, functionality (all stages explained in backup)
- pipeline runs through for every commit

---

<table>
<thead>
<tr>
<th>Compilation</th>
<th>Testing</th>
<th>Formatting</th>
<th>Documentation</th>
<th>Packaging</th>
<th>Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmp.cc7-gcc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cmp.cc7-gcc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cmp.cc7-llvm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cmp.mac1014-cl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cmp.slc6-gcc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cmp.slc6-llvm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

_for every commit_  

_for every merge request or tag_
GitLab Continuous Integration – all stages

- **Compilation**
  - compile source code on Scientific Linux 6, CentOS7, and Mac OS X with GCC, Clang, and AppleClang

- **Testing**
  - analyse test data sets and compare output to pass conditions

- **Formatting**
  - check format against defined syntax rules (e.g. tabs ↔ whitespaces) to avoid changes caused e.g. by different indentation, and apply linting

- **Documentation**
  - compile user manual from LaTeX sources and generate Doxygen code reference

- **Packaging**
  - generate release tarballs

- **Deployment**
  - publish new version of CVMFS, new docker image in registry, new user manual and code reference on the website and release tarballs