GLORIA is funded by the European Union 7th Framework Programme (FP7/2007-2013) under grant agreement n° 283783

GLObal Robotic telescopes Intelligent Array for e-Science

Data Reduction Methods

Documentation of data reduction methods used by RT in the network and the resulting data combination algorithm.

CODE: DEL-053

VERSION: 01

DATE: January 02, 2013

Authors:	Chris LINTOTT (UOXF) Robert SIMPSON (UOXF) Aleksander Filip ZARNECKI (UWAR)
Collaborators:	
Revised by:	Fernando IBÁÑEZ (UPM)
Approved by:	Fernando SERENA (UPM)

Distribution List:

Name	Affiliation	Date
Fernando IBÁÑEZ	UPM	December 31, 2012
Fernando SERENA	UPM	December 31, 2012

Change Control

Issue	Date	Section	Page	Change Description
01.A	31/12/2012	All	All	Creation

Reference Documents

Nº	Document Name	Code	Version

Index

1. Introduction

The principal data-handling problem posed by the GLORIA network is be the automatic handling of the variety of (and potentially size of) data from online and offline experiments. Users will be dealing with huge amounts of data and large variety of analysis tasks. We need an analysis framework which would be both very efficient and very flexible. Luiza is a package based on Marlin (Modular Analysis and Reconstruction for the LINear collider), a simple modular application framework for development of reconstruction and analysis code for International Linear Collider.

2. Luiza framework

2.1. Basics

In Luiza, every computing task is implemented as a processor (module) that analyzes the data stored in an internal data structure and created additional output is also added to that collection. The advantage of such a modular approach is to keep things as simple as possible. Every single step of the full analysis chain that goes e.g. from raw images to light curves can be processed step by step and the output of each step is still self consistent and can be fed into the next step without any manipulation.

Data reconstruction and analysis should be divided into small, well defined steps, implemented as so called processors. Processors are grouped in modules, dedicated to particular tasks or types of analysis.

The crucial requirement in such an approach is that each step of the analysis has a well defined input and output data structure. By defining universal data structures we make sure that different processors can be connected in a single analysis chain, i.e. exchange data and analysis results.

The dedicated processor manager loads selected processors and calls their corresponding methods for subsequent steps of data analysis.

The Marlin framework turned out to be very efficient and flexible, and is widely used by the ILC community. It is designed to run in a batch mode, without user interaction, with all input streams, analysis tasks, parameters and options specified in the steering file. Therefore we decided to use the same concept in development of the Luiza framework for GLORIA.

The package is developed mostly in C++ and makes wide use of Standard Template Library (STL) classes and methods.

2.2. Data Structures

FITS (Flexible Image Transport System) is the standard astronomical data format endorsed by both NASA and the IAU. FITS is primarily designed to store scientific data sets consisting of multi-dimensional arrays (spectra, images or N-dimensional data cubes) and 2-dimensional tables containing rows and columns of data. When developing Luiza we decided to use the CFITSIO library for reading and writing data files in FITS format.

We have also defined the following basic data classes:

- **GloriaFitsImage**: class for storing 2-dimensional FITS images, with either integer or floating point

pixels. It includes also basic methods for image manipulation (addition, subtraction, multiplication and division)

- **GloriaFitsTable**: class for storing data tables. Different column types are allowed (integers, floats, strings, but also vectors of integers and floats)
- **GloriaFitsHeader**: class for storing FITS header data, includes basic methods for accessing and modifying header information. Both GloriaFitsImage and GloriaFitsTable inherit from this class.
- **GloriaTelescope:** class for storing telescope parameters.

Additional classes can be implemented on this basis, as for example the **GloriaObjectList** class, which defines **GloriaFitsTable** with predefined columns for storing object position on CCD (*CCD_X* and *CCD_Y*) and object brightness (*SIGNAL*). This makes sure that object lists will be exchangeable between processors. User can add additional columns if needed.

For internal storage of all data being processed a dedicated class **GloriaDataContainer** was implemented. It stores vectors, so called collections, of images or tables. Each collection has a unique name, which can be used to access its elements. Multiple collections can be stored in memory, each with multiple images or tables. A pointer to **GloriaDataContainer** is passed to each Luiza processor in the data processing loop.

2.3. Data Processing

We assume that every computing task can be implemented as a processor (module) that analyzes the data stored in a **GloriaDataContainer** structure and created additional output is also added to that structure.

A user defines the analysis chain at run time, by specifying list of active processors in an XML steering file. The idea is to develop a large number of processors for GLORIA, doing many different tasks, so user is always able to find a set which matches his/her needs.

The core engine of Luiza is the processor manager (**ProcessorMgr** class). It is used by Luiza to create a list of active processors (after parsing an XML file), and setting values to parameters required by these processors (given in the same XML file). The same processor types (e.g. the processor reading FIT images - **FitsImageReader**) can be used many times. Each instance has its own set of parameters, so for example, one instance of image reader can be used to read dark frames used for calibration and another instance to read actual images.

The current version of Luiza includes already some basic tools for image processing:

- An image viewer (based on the CERN ROOT package from Marlin)
- An image normalization processor, allowing for dark/bias subtraction and flat correction
- A processor for image stacking or averaging
- A processor for simple geometry operations: image cropping and rotations
- An object finding algorithm based on the particle identification algorithm developed for silicon pixel detectors...
- ...and another one based on Python library Mahotas
- An astrometry algorithm based on Astrometry.net (still under tests)

2.4. Continuous development

We plan to continue development of basic tools for image manipulation and analysis (astrometry, photometry, light curve reconstruction). General purpose algorithms, which should be flexible enough to cope with data from all GLORIA telescopes, are not supposed to be the most precise ones. They can be used as examples and starting points for future improvements and development of more advanced tools, dedicated to particular studies.

Dedicated processors will also be development as a part of GLORIA offline experiments. Our current plans include development of:

- An interface to star catalogues and external databases
- An interface to Virtual Observatory resources
- A processor for smart image stacking (correcting for image shifts and rotations)
- A processor for frame quality analysis
- A processor for high quality aperture and profile photometry
- A processor for light curve determination and variability analysis
- A processor to search for optical bursts, such as flares.

Thanks to simple and modular structure of Luiza, individual GLORIA users will also be able to contribute to software development. New packages can be compiled as independent libraries and loaded dynamically at run time without the need to change anything in Luiza
or other modules.

It is therefore possible for users to develop ‘private’ Luiza modules and libraries, deducted to for their particular analysis, which can be later included in Luiza as a separate package after proper testing.

3. Online documentation

Luiza is being developed using the Doxygen package for framework documentation. HTML and LaTeX documentation is created automatically from class header files, based on simple tags used in the comments included in the code. Additional work needed to keep the documentation up to date is minimal, assuming developers do put comments in the code anyway.

Documentation for the current public Luiza release is available on the dedicated web page at:
http://hep.fuw.edu.pl/u/zarnecki/gloria/luiza/doc/html/index.html

GLORIA Partners

UPM
Universidad Politécnica de Madrid
SPAIN

AUAV
Astronomical Institute, Academy of Sciences of the Czech Republic
CZECH REPUBLIC

CSIC
Consejo Superior de Investigaciones Científicas
SPAIN

INAT & CVUT
Czech Technical University in Prague
CZECH REPUBLIC

IP-ASCR
Institute of Physics of the Academy of Sciences of the Czech Republic
CZECH REPUBLIC

IAC
Instituto de Astrofísica de Canarias
SPAIN

INAF
Istituto Nazionale di Astrofisica
ITALY

SAO
Special Astrophysical Observatory of Russian Academy of Sciences
RUSSIA

UCDNUID
University College Dublin
IRELAND

UC
University of Chile
CHILE

UMA
University of Málaga
SPAIN

UOXF
University of Oxford
UNITED KINGDOM

UNIWARSAW
Uniwersytet Warszawski
POLAND