Status of MEGAlib's Real-Time Analysis Tool Realta

Andreas Zoglauer, Michelle Galloway, Mark Amman, and Steven E. Boggs

***Abstract* – Realta is the real-time analyzer of MEGAlib, a software tool to simulate and analyze the data from soft-to-medium-energy gamma-ray detectors in space and on the ground. Realta encompasses the complete data analysis path which ranges from coincidence determination, Compton scatter and pair creation event reconstruction, event selection, image reconstruction to spectral identification. Realta is highly multi-threaded to fully utilize today's multi-core CPUs and to enable real-time processing. Taking advantage of the versatility of MEGAlib, Realta can be applied to a wide variety of gamma-ray detectors (pixel detectors, strip detectors) and measurement scenarios (near field 2D and 3D, far field). Realta and MEGAlib are open source, freely available, written in C++, and based on ROOT.**

I. Introduction

For astrophysics as well as for terrestrial applications the fully automated real-time analysis of gamma-ray data is of great importance. In astrophysics the real-time localization of transient events such as gamma-ray bursts and their differentiation from similar events such as terrestrial gamma-ray flashes or solar flares is important for initiating immediate follow-up observations in other wave bands. On the ground, homeland security applications require the real-time detection, identification, and localization of nuclear incidents and threats. However, the indirect nature of imaging gamma rays in the energy band from a few tens of keV to a few MeV – the energy band where usually coded masks and Compton telescopes are used – requires sophisticated and thus time consuming event and image reconstruction approaches.

MEGAlib [1] – the medium-energy gamma-ray astronomy library – is a software tool kit which contains all software tools for the simulation and analysis of gamma-ray detectors operating from the X-ray to the pair production regime. A first single-threaded version of a MEGAlib-based real-time analysis tool was originally developed for the 2001 calibration campaign of the MEGA Compton and pair telescope [2]. In the context of this work it has been redesigned to be applicable to a wide range of detector systems and enhanced for high-throughput analysis by utilizing multi-threading.

Since Realta is based on and part of MEGAlib, it is open source, freely available from the MEGAlib website (http://www.mpe.mpg.de/MEGA/megalib.html) through cvs, and based on ROOT [3].

II. Data flow and Data Management

The data of an imaging gamma-ray detector needs to pass through several analysis steps, which are all reproduced within Realta, each in its own analysis thread:

- Data acquisition – here via a TCP/IP connection from the hardware or from a program reading simulated data
- Coincidence determination within a given time window for detector systems without the appropriate hardware
- Spectral identification of isotopes by looking at detected nuclear lines
- Event reconstruction of Compton, pair-creation, and charged particle events
- List-mode image reconstruction in near or far field
- Display of the final results

In a multi-threaded environment the event data has to pass through all these analysis steps in a way that allows concurrent access without blocking other threads. The following scheme has been devised to achieve this goal.

All the data of an event is stored in a single class called MRealTimeEvent (see Fig. 1). This class contains slots (pointers) to other objects presenting different analysis stages of the event. In addition, Boolean flags indicate that the event has passed a certain analysis stage or a special circumstance occurred such as the event has been dropped due to high CPU load or it is in coincidence with another event. Each analysis

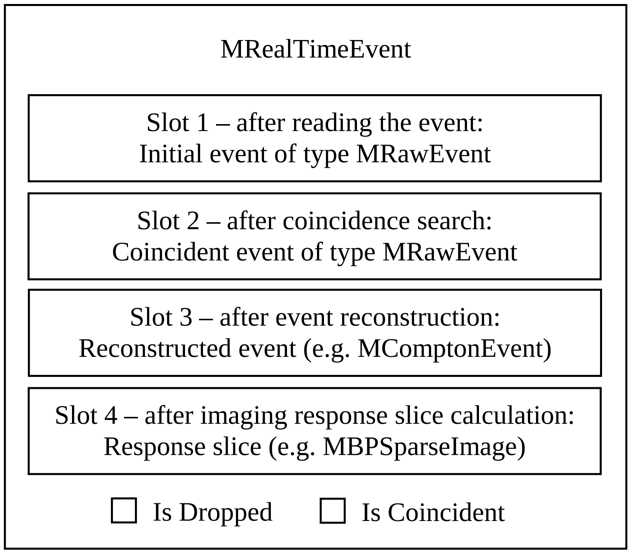

MRealTimeEvent
Slot 1 – after reading the event:
Initial event of type MRawEvent
Slot 2 – after coincidence search:
Coincident event of type MRawEvent
Slot 3 – after event reconstruction:
Reconstructed event (e.g. MComptonEvent)
Slot 4 – after imaging response slice calculation:
Response slice (e.g. MBPSparseImage)
□ Is Dropped
□ Is Coincident

Fig. 1. Simplified layout of the class MRealTimeEvent

Manuscript received November 15, 2011. This work was supported in part by the U.S. Department of Homeland Security, Domestic Nuclear Detection Office, under Interagency Agreement HSHQDC-08-X-00832 and by the U.S. Department of Energy, Office of Science, under Contract DE-AC02-05CH11231.

A. Zoglauer is with the Space Sciences Laboratory, University of California at Berkeley, CA 94720, USA (telephone: +1-510-643-7735, e-mail: zog@ssl.berkeley.edu).

M. Galloway and S. E. Boggs are with the Space Sciences Laboratory, University of California at Berkeley, CA 94720, USA.

M. Amman is with the Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA.

thread only adds newly generated information to the event into the slot associated with the analysis thread. For example, the image reconstruction thread adds a new response slice and sets a flags indicating that this event has been processed by this analysis step. The next thread waits until this flag is set before it reads and processes the data of this event.

In addition, all events are stored in a double-linked list, which must allow for concurrent adding events in the front, sorting events for ordering them via time, deleting the events after they are no longer needed at the back of the list, and parallel accessing the events. Since the double-linked list in the standard library (std::list) is thread safe it fulfills all requirements. The individual analysis threads all contain an iterator of that list which points to the next event which should be analyzed, or to the first event which shall be used for histogramming or spectral analysis.

This setup achieves efficient, multi-threaded data handling since no mutex is required to guard the data itself, only the final data products such as the spectrum, image, and the detected isotope list.

III. The Analysis Pipeline

The individual modules of the analysis pipeline all utilize existing MEGAlib classes. These classes only required slight modification both to their application programming interface and to ensure reentrance for multi-threaded access from within Realta.

Internally the analysis is managed by the class MRealTimeAnalysis. This class takes care of starting and stopping the threads, managing all user options for event, spectral, and image reconstruction, and of safely passing the final data products to the GUI.

All Realta analysis steps require a geometry and detector description. Within MEGAlib the Geomega [4] library provides a universal detector library which can represent a wide variety of different setups. A simple ASCII file contains the detector description for volumes, materials, detectors, and triggers. The file is keyword based and reflects the object-oriented structure of Geomega. For example, a volume consists of the following parameters: shape, material, position, rotation, mother volume, and visualization attributes. Geomega can also handle advanced features such as constants, math functions, for-loops, if conditions, matrices, etc. to simplify constructing a geometry. Several widely used detector types such as strip detectors with and without depth resolution and voxel detectors are built into Geomega, along with a sophisticated detector effects engine which enables realistic energy, position, time resolutions, noise and trigger thresholds, ADC overflows, etc. which are required for simulations.

Fig. 2 shows the geometry which was used for the simulations and reconstructions. The detection system consists of two half-spherical detector modules, which are built of 1-cm^3 sized CZT elements. The two modules are mounted on a helicopter platform for long-range standoff detection. This detector system is an envisioned advanced version of HEMI, the high-efficiency multi-mode imager [5].

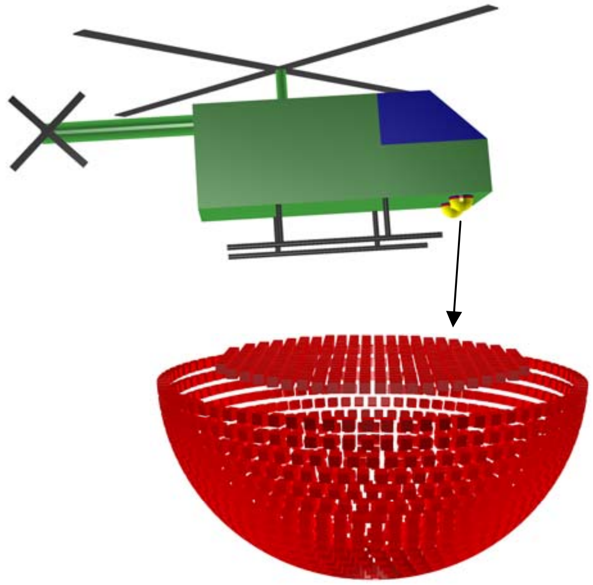


Fig. 2. Detector models: The top section shows a helicopter model with the two 1500-element half-spherical HEMI instruments mounted, and the bottom section shows the arrangement of the CZT elements (passive surrounding material is not shown) in a backplane and a half-spherical structure.

A. Data input

In order to decouple data acquisition and data analysis, and to enable remote analysis, Realta receives its input data via a TCP/IP interface. The user only has to specify a host (or an IP address) and a port. The data is transferred in a simple ASCII format containing position, energy, and time of the event. Since data calibration is usually very specific to a given detector system and Realta aims to be a universal analysis tool, the transferred data has to be calibrated. This first analysis step also makes sure that the events are ordered by time, and identifies and ignores potentially corrupted data sets.

Internally this step creates an MRealTimeEvent object and fills its first data slot with an object MRawEvent containing the event time as well as energy and position of all associated hits. This object is then added to the front of the event list. The MRawEvent class can be processed by the following event reconstruction steps.

B. Coincidence Determination

Coincidence detection is one of the simplest tasks in Realta and is included because not all detection systems provide it in hardware. In this analysis step, all events which occur within the coincidence time interval (e.g. 2 µs for HEMI) are combined into one event. This step can be deactivated in the GUI if desired.

Internally this step uses the coincidence engine of MEGAlib's Revan library. It takes the MRawEvent from the first step and creates a new MRawEvent object, which contains all hits of all coincident events. It is added to the second slot in the first coincident (or only) MRealTimeEvent object. All following coincident events are flagged as "coincident" and therefore excluded from any further analysis.

C. Spectral identification

Nuclear threat detection applications require the instantaneous identification of radioactive elements. Realta uses MEGAlib's Spectralyzer [6] library to accomplish this.

The starting point is the events after coincidence determination. All events within a user defined time window from the front of the event list are combined into a spectrum and handed to the spectral analyzer.

The first step of the spectral analysis is peak identification. Utilizing ROOT's TSpectrum class, the input spectrum is smoothed, the background is subtracted, and the peaks are determined with their mean peak energy, their source and their background count rate.

In a second step, those peaks are validated based on their width relative to the detector energy resolution (e.g., for HEMI it must not be smaller than 0.8, but not larger than 1.8 to still detect multiples), their total counts (e.g., at least 20 required), and the significance of the peak above background (e.g., 5σ).

In a third step, the peak positions are used to look up matching isotopes in a user-defined table using 1 sigma of the estimated line energy as acceptance window.

Since each peak may have multiple isotope candidates, a down selection to one final isotope per peak is performed based on the number of found lines per isotope and their expected relative intensity.

The output of the spectral analyzer is a list of found isotopes, which is used by the display thread to mark the found lines in the spectrum.

D. Event Reconstruction

This step searches for charged particle, pair creation, and Compton event signatures by utilizing the energies and positions of the hits along with the kinematics of the events as well as the geometry via absorption probabilities along the potential paths of the particles. The final results are the parameters of the found event type such as the energy of the recoil electron and the scattered gamma-ray along with first and second interaction position for Compton events. For details about event reconstruction within MEGAlib see [7].

This step also uses MEGAlib's Revan library, which besides others provides a wide variety of different reconstructions options for Compton scatter events such as different variants of the classic Compton sequence reconstruction and Bayesian event reconstruction, Compton electron tracking options as well as pair creation and charged particle reconstruction options. All of the options are accessible via the Settings menu.

After the event reconstruction is performed, the third slot in MRealTimeEvent is filled with a new classification of the event derived from the base class MPhysicalEvent, describing the event as single-site (MPhotoEvent), Compton scatter (MComptonEvent), pair creation (MPairEvent), muon (MMuonEvent), or unidentifiable event (MUnknownEvent). These classes can determine all high level parameters of the underlying events, such as Compton scatter angle or pair opening angle. This object type can then be used by the high-level data analysis steps described in the following section.

Event reconstruction, especially if it is utilizing absorption probabilities, can be very time consuming, and can limit the overall throughput. Therefore measures have to be taken to prevent this analysis step from falling behind and queueing up too many unanalyzed events. To accomplish this, the CPU core load is continually monitored by looking at the wait time for new events. If the load is at maximum, a self-determined fraction of the events is randomly not reconstructed and thus flagged as "dropped". The fraction of the dropped events is dynamically adjusted to achieve a maximum load of around 90% of the used CPU core.

E. Image reconstruction

The reconstructed events can now be imaged. Realta uses MEGAlib's imaging library Mimrec [8] for this task and allows full access to all options available in Mimrec via the Settings menu. This includes performing event selections on all performance-relevant parameters of single-site, Compton, and pair events (e.g. energies, interaction distances, scatter or opening angles, etc.) as well as list-mode likelihood image reconstruction in spherical, Galactic, and Cartesian coordinates (2D, 3D) including different response calculation approaches.

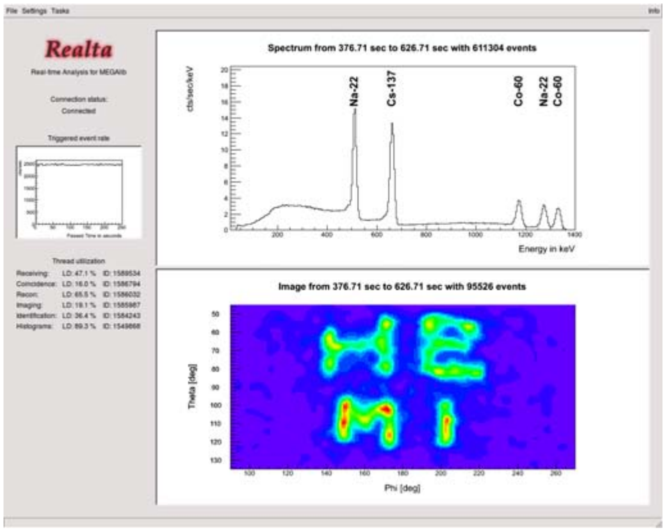
Internally this is a two thread process. The first thread only calculates the response slices (the back projections, e.g., the event cones for Compton events) for each event and adds them to the last slot in the MRealTimeEvent object. Since this step can also be quite CPU intensive, a similar approach as for the event reconstruction thread is used. Events are dropped as necessary to prevent the CPU core utilization from reaching 100%.

A second thread then uses an (ordered-subset) maximum-likelihood expectation-maximization approach to deconvolve the events into the final image. The output is an image, a TH2D histogram, which is brought to the screen by the display thread.

F. Display

The final task is to display the data. Figure 3 shows an example using the HEMI geometry from Fig. 1. In the simulation, the HEMI helicopter is situated 25 m above ground. The source photons originate from ^{137}Cs, ^{22}Na, and ^{60}Co sources. They build the word HEMI in a 9x7 dot matrix of point sources per letter. For the analysis only good Compton events with an interaction distance of at least 10 cm in the detector have been preselected to be sent to Realta.

The main GUI consists of three main sections. The section on the left shows the connection status, the number of received events as a histogram, and a table giving the CPU-core load of the individual threads along with the ID of the



Receiving:	LD: 47.1 %	ID: 1589534
Coincidence:	LD: 16.0 %	ID: 1586794
Recon:	LD: 65.5 %	ID: 1586032
Imaging:	LD: 19.1 %	ID: 1585987
Identification:	LD: 36.4 %	ID: 1584243
Histograms:	LD: 89.3 %	ID: 1549668

Fig. 3. The main Realta user interface with the three main histograms: The rate of triggered events, the detected spectrum, and the reconstructed image. In addition, a table giving the utilization of the individual threads can be seen. The full set of MEGAlib's event reconstruction, image reconstruction, and event selection options is accessible via the Settings menu, while the Tasks menu allows starting, stopping, and resetting the analysis.

latest analyzed event. The top right histogram shows the spectrum from the last 250 seconds (user set) along with all identified lines. The bottom right slot shows the reconstructed image, where the letters HEMI are clearly visible, albeit slightly distorted due to the projection of the letter from a distant Cartesian plane onto far-field spherical coordinates. The full set of MEGAlib's event reconstruction, image reconstruction, and event selection options is accessible via the "Settings" menu. However, all changes require a reset of the real-time analysis - with the exception of the time range over which the spectrum and image are generated. The "Tasks" menu allows starting, stopping, and resetting the analysis.

IV. Performance Considerations

The performance of Realta depends on many parameters such as detector setup, event complexity, energy, selected reconstruction options, and operating system. Therefore a prediction of the throughput for a given instrument is not possible.

Realta can be compiled on Linux, Mac, and Windows. However it has only been fully optimized for Linux. As a consequence, its performance on Windows 7 is roughly a factor two worse than on Ubuntu 10.4 for the above HEMI setup.

In addition, the performance depends on the complexity of the average event and thus on the overall detector setup. A simple calorimeter, which only measures energies, has a much higher throughput than a Compton camera, which requires resource intensive Compton sequence reconstruction and image reconstruction. If the Compton telescope is also capable of tracking the electrons, then the complexity of the event reconstruction step is an additional factor of 2-3 times higher. A similar argument holds for the chosen energy range. At lower energies, the number of (Compton) interactions and thus the number of triggered channels is lower, and therefore the event reconstruction is much easier since the Compton sequence reconstruction scales with N! where N is the number of interactions.

Finally, the selected reconstruction options have a strong influence on the reconstruction throughput. For example, using an event or image reconstruction approach which requires the calculation of absorption probabilities is very CPU intensive. In addition, the imaging speed critically depends on the number of selected image bins.

As seen in Fig. 3, using the given HEMI geometry and *only* Compton events, a maximum throughput of roughly ~3800 Compton events per second can be achieved per core. The limiting analysis step is in this case the event reconstruction ("Recon"). However, an event reconstruction approach was used which takes into account absorption probabilities. Using a more simple approach would further increase the throughput. Although in Fig. 2 the histogramming thread is running at almost 90%, it is not the limiting factor, because it was just set to generate and update the histograms as fast as possible.

V. Future Developments

While Realta has been successfully used for HEMI measurements and simulations, it is still under development. Some of the envisioned features for the next version include:

- Automatically splitting the event reconstruction and image reconstruction tasks into multiple threads in order to increase the throughput for these two of the most time-consuming analysis tasks
- Improving the user interface to also show only an image of the detected lines and only a spectrum of the detected peaks in the image
- Enabling more sophisticated techniques to drop events in the case of high CPU-core load, such as eliminating events by their complexity (number of hits) instead of simply randomly
- Adding an additional binary data format to the TCP/IP receiver to reduce the required network bandwidth compared to the current ASCII format

Nonetheless, already in its present state Realta is a highly versatile tool for real-time gamma-ray detector analysis, which can be applied to a wide variety of detection systems.

References

[1] A. Zoglauer, R. Andritschke, F. Schopper, "MEGAlib – the Medium Energy Gamma-ray Astronomy Library", *New Astronomy Reviews*, vol. 50, no. 7-8, pp. 629-632, 2006

[2] G. Kanbach et al., "Development and calibration of the tracking Compton/Pair telescope MEGA", NIM A, vol. 541, pp. 310-322, 2005

[3] R. Brun, F. Rademakers, "ROOT – An Object Oriented Data Analysis Framework", *NIM A*, vol. 389, pp. 81-86, 1997

[4] A. Zoglauer, G. Weidenspointner, S.E. Boggs, M. Galloway, C.B. Wunderer, "Cosima – the Cosmic Simulator of MEGAlib", *Nuclear Science Symposium Conference Record, 2009 IEEE*

[5] M. Galloway et al., “Status of the High Efficiency Multimode imager”, *Nuclear Science Symposium Conference Record, 2011 IEEE*

[6] M. Galloway, A. Zoglauer, M. Amman, S.E. Boggs, P.N. Luke, “Spectral analysis for the High Efficiency Multimode Imager”, *Nuclear Science Symposium Conference Record, 2010 IEEE*

[7] A. Zoglauer, "First Light for the next Generation of Compton and Pair telescopes", Doctoral thesis, TU Munich, 2005

[8] A. Zoglauer, M. Galloway, M. Amman, S.E. Boggs, P.N. Luke, “Design, Implementation, and Optimization of MEGAlib’s image reconstruction tool Mimrec”, *NIM A*, vol. 626, pp. 568-571, 2011