

GIANO: software design and acquisition facilities

E. Rossetti^a, P. Montegriffo^a, C. Baffa^b, E. Giani^b, E. Oliva^{b,c}, L. Origlia^a

^aINAF - Osservatorio Astronomico di Bologna (Italy);

^bINAF - Osservatorio Astrofisico di Arcetri - Firenze (Italy);

^cINAF - Fundacion Galileo Galilei - Santa Cruz de La Palma, TF (Spain)

ABSTRACT

We present the general software design and acquisition facilities of GIANO, an ultra-stable IR echelle spectrometer optimized both for low ($R \simeq 500$) and high ($R \simeq 50,000$) resolution, that will be mounted at Nasmyth-B focus of the Telescopio Nazionale Galileo (TNG).

We describe the high-level software structure of the instrument, the user interface characteristics and the control of all subsystems. The management of GIANO sensors and controls of the mechanical movements is indeed a crucial issue that is handled by dedicated tasks. Monitoring of all these parameters is performed by means of separated processes running in background on the control workstation (PC).

In this paper we will also schematically discuss the software for the instrument control, status display and setup, the quick look facility and the pipeline for data reduction.

Keywords: Software control, data reduction, infrared spectrometers.

1. INTRODUCTION

The GIANO project is part of the Second Generation Instrument Plan of the Telescopio Nazionale Galileo (TNG), a 3.58m alto-azimuthal telescope located at Roque de Los Muchachos Observatory (ORM), La Palma, Spain (detailed information on TNG can be found at <http://www.tng.iac.es>).

GIANO is an IR instrument which can perform spectroscopic observations on a wide wavelength range (from $0.9\mu\text{m}$ to $2.5\mu\text{m}$) at different resolution (between 500 and 50,000) combining a set of prisms with an echelle grating. The imaging mode is available only for the centering of an astronomical object in the slit.

GIANO will be installed at the Nasmyth-B focus of TNG, which has a F/11 focal aperture and is equipped with an active optics system. For a general description of the instrument and its performances see Oliva et al.¹ (2006).

One of the primary goals of the GIANO design is to provide the instrument with a good user-friendly management software. As often experience teaches, starting the detailed software design and development at the beginning of an instrumental project is critical to its success. For this reason, before the completion of the optical and mechanical design, we already simulated the data output from the spectrometer detector and a prototype of the Graphical User Interface (GUI) for the instrument. In addition a full simulation (Giano Virtual Instrument) of the different observing modes and the motor movements has been also performed. This strategy allowed us to begin the coding of the final software at an early stage, in order to have it ready during the system integration and test. Software is needed not only for the instrument control and status display, but also for the observing set-up, for “quick-look” spectral analysis at the telescope and for the automated data reduction “pipeline”.

Further author information: send correspondence to
E.R., INAF - Osservatorio Astronomico di Bologna, via Ranzani 1, I-40127 Bologna, Italy;
e-mail: emanuel.rossetti@oabo.inaf.it

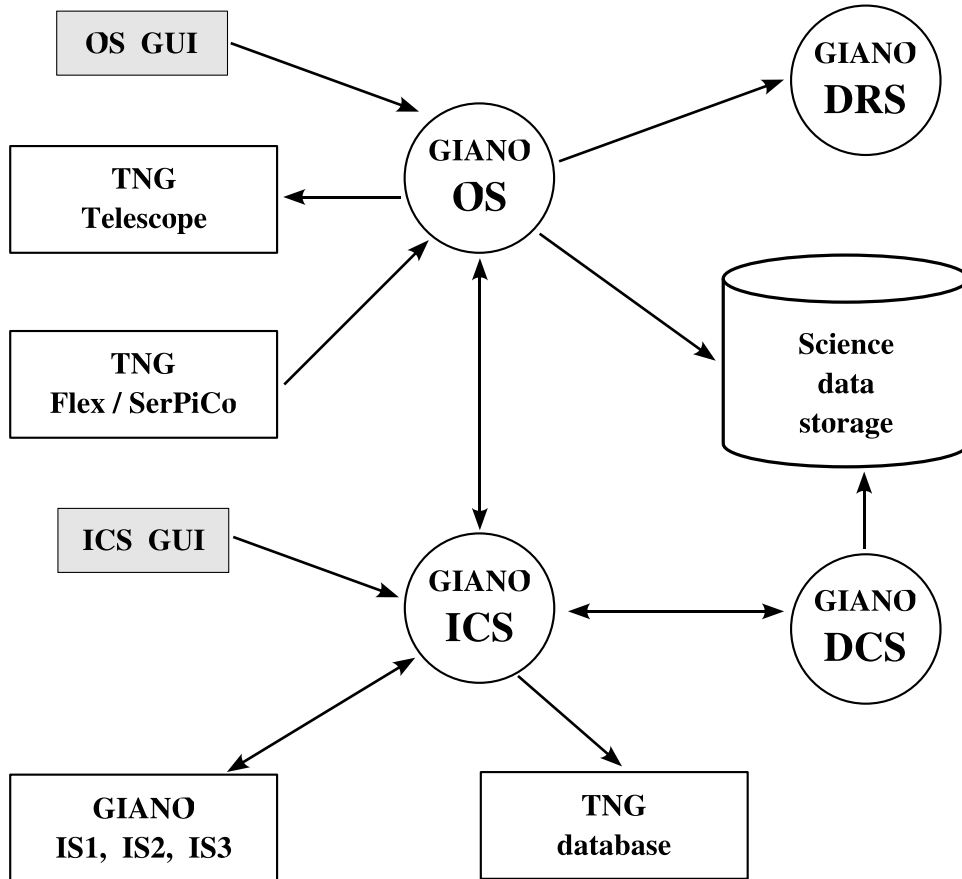


Figure 1. Main architecture blocks for the GIANO control software.

2. GIANO SOFTWARE DESIGN

The GIANO control software is foreseen to run on a dedicated PC-Linux based workstation. This choice is considered robust and reliable compared to other existing PC operating systems. In particular, the use of Linux kernel 2.6 ensures some advantages in terms of general performances, software improvements, networking and system security.

In its general structure the GIANO software system can be subdivided into 3 main levels:

- the low-level software. This level is designed to handle all hardware related functions and IR detector controls. It is physically split in two locations: one inside the GIANO workstation and the other into the embedded processor located at the focal plane electronics.
- the high-level software. This level is intended to fulfill all astronomy-related tasks and also to act as an interface between the low-level software and the astronomer/operator. Hence, it includes several GUIs to provide a full control of all sub-systems.
- the scientific software. It includes the observing block preparation tools and the off-line data reduction pipeline.

Fig. 1 shows the main architecture of the GIANO Control Software. Each software level deals with different aspects of the GIANO functionality and include several blocks, which can be grouped as follows.

- *Detector Control Software* (DCS): it manages the spectrometer array controllers. It is directly interfaced with the hardware and it is responsible for the initial handling of scientific data in real time.
- *Instrument Control Software* (ICS): this block manages the instrument status and also provides a GUI for monitoring purposes and interfaces with specific hardware modules, like the detector, the sensors and the motors. Finally it provides a list of FITS keywords describing the instrument status.
- *Observation Software* (OS): this branch coordinates TNG telemetry command variables, ICS and Flex (the TNG software for handling the observation blocks). It controls all the parameters needed to perform scientific exposures: instrument setup, initializations, array setup, calibration units setting and exposure time count-down. It also provides a GUI and additional frame processing facilities for quick look purposes.
- *Observation Support Software* (OSS): this task provides a tool for building off-line Observation Blocks during the *phase I* and *phase II* preparation. In its final version it will also include a web interface to the GIANO Exposure Time Calculator (ETC).
The final structure and implementation of OSS will be performed after the GIANO integration phase.
- *Data Reduction software* (DRS): it provides the off-line data reduction pipeline.

Each block is devoted to a particular job and is independent from the others. The advantage of this modular structure implies both reduced software complexity and easier maintenance. All the interprocess communications are implemented by means of Unix-like sockets. The communication protocol has been designed to be as uniform as possible along all channels and consists of packets which are composed by a header containing essential information, among which the “magic number”, a destination address, the packet description and reference number, a validity check and the related packet length. All data are in plain ASCII for easy of debugging and maintenance.

2.1. Detector Control Software

The DCS consists of two different parts: the first resides inside the PC-Linux and the second runs on the embedded processor at the focal plane electronics and has direct access to the hardware. The two branches are interfaced by means of a standard *Ethernet* connection, on which data and commands are sent to Unix-like sockets. The software portion inside the embedded processor manages the section of the appropriate waveform for the detector, controls the data flow and can handle special observing modes such as multiple starts and stops. More specifically, it can perform any of the following tasks:

- setting the array read-out mode and detector integration time (DIT);
- starting an integration and, at its end, computing the image as start end (double correlated readout) or ramp integral (multiple non-destructive read-out) and transferring the resulting frame to the instrument workstation;
- optional transferring of each single non-destructive read-out frame (useful for post-processing and defining the best read-out and co-adding strategy for long integrations);
- averaging a number (NDIT) of individual DITs before transferring the image;
- executing a sequence of dummy images (useful to clean the array from persistence effects);
- monitoring and setting the array temperature.

In order to communicate with the DCS, which controls electronics and the related firmware, one or more ethernet sockets are used. A series of defined commands are sent to an interpreter and translated into DCS commands. Moreover to interact with the IR detector system, interfaces with DCS are also developed. The array can also perform short frame acquisition of the GIANO slit field of view and positioning for quick-look and/or guiding purposes.

For further details about GIANO DCS and communication protocol (between DCS and ICS) see Baffa et al.² (2006).

2.2. Instrument Control Software

As a middle-ware between the hardware and the scientific functions, the ICS is in charge of collecting and recording instrument status data, forwarding movements and setting commands, and synchronizing the operational flow.

The management of all GIANO sensors and controls of the mechanical movements are a crucial point that must be handled by a dedicated functions called *Instrument Status* (IS). The monitoring of all these parameters are foreseen to be done by means of 3 separated processes that must always run in background on the GIANO PC-Linux system. These tasks can be summarized as follows:

- *IS1*: controls the temperature and pressure values inside the dewar;
- *IS2 (cold movements)*: drives the movements of all stepper motors and switches;
- *IS3 (warm movements)*: commands the movements of the optical derotator, polarimetric unit and calibration lamp switches.

The more practical solution to build each IS block is to use a socket transmission protocol between the PC and the related hardware controller. In case of serial controllers it is possible to use a serial-Ethernet transceiver like e.g. the one based on the TNG Rabbit Module System. Each IS input/output command is driven and addressed to a specific IP number (one for each sub-system).

The temperature and pressure monitor program (*IS1*) reads every few seconds the temperatures of about a dozen sensors inside the criostat and the dewar pressure. Higher reading frequencies are not necessary as the thermal time constants of GIANO are expected to be of the order of hours.

The continuous monitoring and storage of temperature and pressure values are mandatory in order to perform the initialization of the GIANO control system. For this reason the *IS1* background process (at the highest priority) must be always running.

IS1 has a startup procedure which senses the environment and acts accordingly. Obviously, there can not be two *IS1* processes running at the same time. For this reason, at start, the process checks if another *IS1* is already running. In this case it checks the status of the process. If the previous process is running the second *IS1* kills itself. If there is not any previously running daemon, *IS1* performs sensors initialization, publishes data and installs itself as a background daemon.

All sensor values read by *IS1* are sent immediatly to the TNG database by the ICS. If the link to the database is not active/available the values are stored into a local log file and sent to database as soon as the communication channel is ready.

For *IS2* (cold movements) and *IS3* (warm movements) a similar scheme is foreseen. Both daemons monitor continuously the instrument status and communicate all data to the ICS. *IS2* and *IS3* can receive movements commands and transform them into elementary instructions to be executed by the motor controllers. Both motor daemons follow the same intelligent startup procedure of *IS1*. *IS2* and *IS3* provide an on-line status of all motors, and communication of these parameters run in both directions (client-server system-like).

For ICS a main graphical User interface (GUI) is developed. This is a stand-alone panel that allows direct operations at ICS level without involving any OS tasks. The ICS GUI is used mainly for test and maintenance purposes. Its basic functionality can be summarized as follows:

- startup/shutdown ICS;
- set ICS devices under handset control;
- change the instrument configuration;
- set commands like: STANDBY, ONLINE, OFF;
- monitoring and setup of all motors and lamps;
- temperature and pressure monitoring.

As all sensor measurements are stored in the TNG database, they can be viewed or plotted on ICS GUI only by direct inquire to the database TNG workstation. If that machine is down, the GUI sensor monitoring is automatically redirected to read the local log file. As a general rule the values of temperature and pressures are continuously monitored giving an alarm signal when out of a critical range.

2.3. Observation Software

GIANO OS is responsible for the coordination of all control activities related to a “single” exposure, for all subsystems involved (including “external world” communications with other TNG PC/workstations).

The basis for the OS is a keyword system. Communications between the instrument and the observer is based on a small library of keywords whose values can be read or modified. Observers can operate the instrument modifying entries in the OS GUI whilst GIANO superusers/operators can type keywords directly on the command line at the operating system level.

GIANO OS can perform the following main tasks:

- issuing telescope offset commands to the TNG control workstation, reading telemetry from it and setting the autoguide;
- reading and modifying the position of the DOLORES slide, holding the first pre-slit mirror;
- receiving/transferring images and instrument telemetry parameters to the TNG archive;
- monitoring through ICS the detectors status;
- reading from ICS the temperature and pressure values inside the criostat;
- sending through ICS the commands to the motors, lamps and polarimetric unit;
- reading (when available) the Observation Block sequences from TNG *Flex/SerPiCo*.

The OS task can perform any available instrument setup. Since there are no motor collision risks inside GIANO, there are not specific constraints concerning the sequence of individual setup movements inside the spectrograph. Thus, during the setup, all motor units can be moved simultaneously. For cold movements the following motors are foreseen:

- 3+1 in/out movements:
 - Shift High Resolution (SHR): insert/remove a prism;
 - Low Resolution (LR): insert/remove a mirror;
 - Imaging mode (IMA): insert/remove a mirror, used only for object centering and maintenance;
 - SLIT lock: insert/remove mechanical lock, coupled with slit wheel;
- 2 continuous movements:
 - FILTER wheel;
 - SLIT wheel (coupled with the in/out movement for slit lock).

For warm movements (all external to the cryostat) two kinds of motors are foreseen:

- rotatory stages:
 - 1 for de-rotator
 - 2 for beam splitter/dichroics (CCD and IR camera)
 - 4 wheels for polarimetric plates and analyzer

- linear stages for:
 - selecting the polarimetric plates
 - moving the preslit bench and select the various observing modes (normal HR/SHR/LR, absorption-cell and polarimetric).

Also for the OS a main GUI has been developed. This is a stand-alone panel that allows direct operations at the OS level, involving any of the ICS tasks. Both OS and ICS GUIs are developed in Tcl/Tk (for example see Welch³, 2000) and useful Tcl library extensions.

The OS GUI is divided into four main panels: GIANO control (OS), detector control (DCS), telescope control (TCS) and display frame. These GUIs will allow to:

- monitor the devices under ICS control;
- configure the observing mode;
- setup the slit/filter;
- switch on/off calibration units;
- start/stop detector integration;
- monitor exposure count-down;
- display target information (telescope RA, DEC and Epoch);
- display the acquired frames.

In order to view all acquired frames we adopted DS9 as the standard real-time and quick-look display for GIANO. SAOImage DS9 (<http://hea-www.harvard.edu/RD/ds9/>) is a Tcl/Tk GUI with a powerful image display widget, integrating a number of popular libraries for world coordinate system computations, remote image access via HTTP, and other features. This package is commonly used by astronomers to view/handle *fits* files.

In particular, SAOImage XPA messaging system provides an easy communication between DS9 and other Unix/Linux tasks, including X programs and a few scripting languages. It also provides an easy way for users to communicate with DS9 by executing XPA client commands in the shell or by using such commands in scripts. Because XPA works at the programming as well as shell levels, it is a powerful tool to interface different environment packages.

The GIANO OS GUI contains an embedded DS9 panel which is configured to display automatically the acquired frame at the end of each exposure. To avoid undesirable activities (i.e. data reduction) the user can only perform a few essential actions: set the zoom, change scale and color display, enable horizontal/vertical cut graph. Manual loading of *fits* files is disabled.

The GIANO Imaging mode is available only for the centering of an astronomical object in the slit. To allow this operation a direct interface between the embedded DS9 and IRAF (<http://iraf.noao.edu/>) has been added. To center the star the user can choose either *daoedit* or *imexam* tasks. Using the cursor keys it is possible to estimate the center of a displayed object, plot the radial or the surface profile, get some essential statistical parameters. We have adopted the standard graphic interface provided by IRAF.

2.4. GIANO Virtual Instrument

An important aspect during the software development process is ensuring that it fully satisfies all the required specifications. Thus, for a proper testing of the various software and communication tasks, it is crucial to design the logical structure of the software before the instrument completion. For this purpose the GIANO Virtual Instrument (GVI) has been developed, simulating the behaviour of all cryogenic units and related motors. It consists of a TCP/IP interface with a socket transmission protocol so that the GIANO control software can communicate to it as if it were the real instrument. In particular, two distinct procedures have been created: a server interface (it must be installed in the same place where the simulation is performed) and a client interface (the control panel is operated in remote mode by the user).

The items of this software simulation can be summarized as follows.

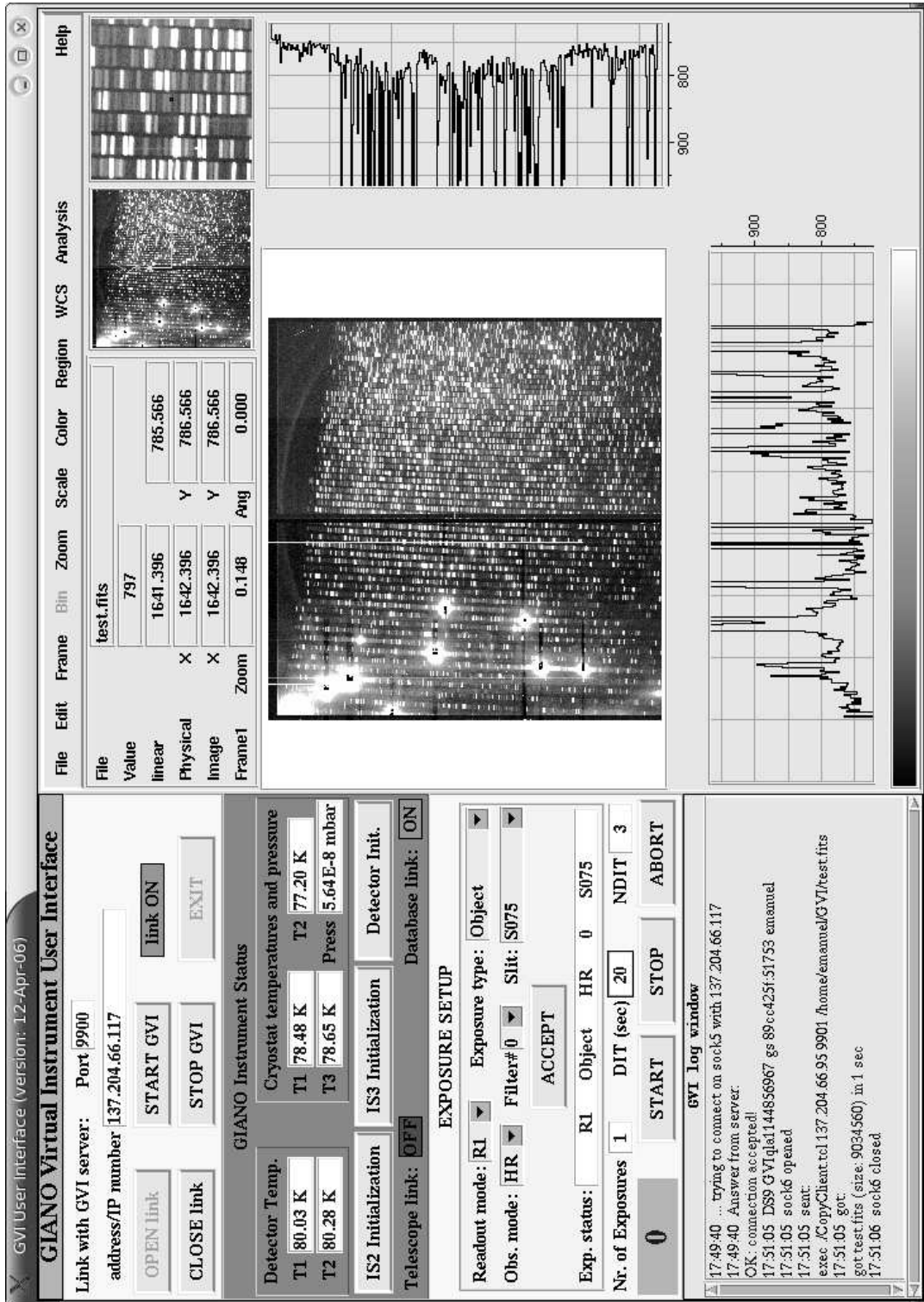


Figure 2. Main panel of GVI User Interface (client side).

- From the server side we are able to:
 - startup/shutdown the GVI socket transmission;
 - setup the 3D visualization procedures;
 - manage the random delay routines simulating the electronic response time.
- From the client side we can:
 - open/close link to the GVI;
 - display cryostat temperature and pressure (simulated);
 - initialize IS2, IS3 and array (simulated);
 - setup the observation mode, the filter, the slit and the exposure time;
 - start/stop/abort the selected exposure;
 - display the *fits* (dummy) file on the embedded DS9 panel sent by the GVI server at the end of each exposure.

A crucial point in the GVI simulation is the 3D graphic visualization of the GIANO components inside the cryostat. Since Tcl/Tk 8.4 does not have a toolkit to accomplish a direct 3D display we adopted the GDHE library (<http://www.laas.fr/matthieu/gdhe>). This tool, developed at CNRS/LAAS laboratories, has been designed to visualize robotics applications. It is fully programmable by using the Tcl/Tk scripting language and allows to build a 3D representation of a given geometrical model of an environment and its temporal evolution.

To make possible creation of complex 3D models, GDHE provides a set of predefined primitive objects that can be used to build drawing procedures. The basic objects are: boxes, cylinders, polygons, prisms, spheres, ellipsoids and disks. With a suitable combination of them, it is possible to make more complex objects like gears and holed disks.

At the GVI startup, on server machine, a standalone GDHE interface panel is opened: it displays the main 3D components inside the GIANO cryostat. Any motion commands sent by the remote client is viewed in real time. Then it is possible to follow the motion of any motors. The user positioned at the server side can use the GDHE control panel to set the distance between the observer and the scene point of view. Moreover, it is possible to control the elevation, the azimuth and the lightening of the scene.

The GVI is mainly used as a diagnostic tool for the GIANO ICS.

3. DATA REDUCTION SOFTWARE (DRS)

3.1. Programming Language and environment

The Giano Data Reduction Software (hereafter DrG) has been designed to run on different Unix/Linux platforms and relies on utilities available under the Unix public domain software. All data flow will be through FITS and ASCII formats. The DrG *back end* will be entirely written in C99; the *front end* will be developed in C99 using Gnu GTK libraries and, at some stage, with the popular *DS9* display program via XPA access point. We also plan to release a command-line version working in unattended automatic mode.

The DrG package will be provided with makefiles generated by Gnu *automake* and a *configure* script generated by Gnu *autoconf*.

A small number of external libraries will be needed:

- glib-1.2.10 - required by gtk-1.2.10
- gtk-1.2.10 - the Gimp Tool Kit library
- gsl-1.5 - the Gnu Scientific Library
- cfitsio-2.5 - the NASA's FITS File subroutine library
- fftw-3.0.1 - Fastest Fourier Transform in the West - *only if needed*

3.2. DrG back end

GIANO is an instrument with very few observing modes, each of them producing a quite constant and repeatable distribution of light on the array. This allows us to develop a highly optimized code: 2 main setup will be available for the high and low resolution, respectively.

The DrG modules will be grouped in five major tasks:

- Calibration Tools
- Pre-reduction Tools
- Reduction Tools
- Analysis Tools
- Utilities Tools

3.2.1. Calibration Tools

These tasks work on flat field and calibration lamps exposures and create tables with calibration coefficients. Given the stability of the instrument, the package will provide library calibration tables which can be used as standard calibration for scientific exposures. Calibration tools have been designed to work with a high degree of automation. Many tests have been carried out on high resolution spectra from different instruments such as SARG@TNG, UVES@VLT, NIRSPEC@KECK, FEROS@ESO-MPI2.2M. By providing a configuration file with the instrumental parameters such as the camera focal length, detector pixel size, gain and RON, the echelle groove density, the nominal echelle blaze angle, a pipeline can be run to quickly produce calibrated spectra and some quality check in a fully automatic mode.

- **Order identification and geometry characterization:** for each order it finds the polynomial which best fits the order center, the order width as a polynomial function of the x-coordinate and the range covered along the dispersion direction (x-axis). A check is also made to guarantee the completeness of the order detection. The task creates an output table with fitting coefficients.
- **Order line geometry:** for each order it models the geometry variation of lines across the spectrum, first fitting the shape of each line from the calibration lamps with a polynomial, then looking for a global solution to model the line shape as a function of x-position and order number. Finally, for each order a reference grid is built and the geometric transformation to a linear space is saved in an output table.
- **Wavelength calibration:** it creates an output table with computed coefficients for each order dispersion relation in a full automatic fashion.

3.2.2. Pre-reduction Tools

The aim of these tasks is to transform a science exposure into a FITS file with straightened orders ready for the final extraction of monodimensional spectra. These tasks are carried out in fully automatic mode.

- Bad pixel and cosmic ray identification and flagging;
- Sky Subtraction: two different setups: 'telescope *nodding on slit*' and 'telescope *offsetting*';
- Flat Field correction: it optionally corrects for high frequency intensity variation across a flat field frame and normalizes the order mean values to unit;
- Frame Resampling (order straightening): a flux conservative resampling which maps the data to a geometric space where the echelle orders are parallel to an image axis and the wavelength dispersion is linear within each order.

3.2.3. Reduction Tools

- Object Aperture (OA) definition: it defines the y-range covered by the observed target inside each order;
- 1-D spectrum extraction;
- Optional correction for low frequency variations within the field of view;
- Merging of spectra from different orders.

3.2.4. Analysis Tools

These tasks will work on extracted 1-D spectra and will provide some standard features such as continuum normalization, flux calibration, signal filtering, line profile analysis tools, spectra cross-correlation (to derive relative velocities between spectra), as well as a spectral line rest wavelength reference, suitable to allow a fast access to spectral features of interest, especially for the *on-line* data quality assessment.

The GUI will look like the popular *DS9* display program: it will support multiple frame buffers, scaling and colormap manipulation, arbitrary zoom, pan and rotation, and obviously all the DrG tools to give the user the opportunity to check interactively each step in the reduction and analysis process.

3.2.5. Utilities Tools

Some other utilities will be provided, such as general purpose command-line FITS calculator to allow the user to easily do arithmetic computation of arbitrarily complexity among any wished number of FITS images, by also including computation of averages, weighted averages, medians *etc.*

3.3. Automatic Wavelength calibration

Wavelength calibration of echelle spectra usually demands to manually identify a few lines from a calibration lamp exposure on at least three orders well distributed along the echellogram. By using a suitable atlas of spectral lines it is possible to compute a starting guess for the dispersion relation and iteratively refine it by adding more and more lines to the fitting process. Then a global dispersion formula can be fitted to combine the whole set of orders (in this way it is possible to get a more robust fit where only a few lines are visible).

At present the astronomical community seems still lacking a general software able to perform a wavelength calibration of echelle spectra without predefined solutions or user interaction. This puts serious limits to the development of data reduction pipelines working in a completely unattended mode. With the GIANO DRS we have developed a program that uses a starting guess for the order dispersion model, accurate enough to achieve a good calibration without any user action. This is possible by making use of WCSLIB, a library developed by Mark Calabretta at the Australia Telescope National Facility (<http://www.atnf.csiro.au/mcalabre/index.html>) to handle coordinate systems within FITS standards. Accordingly to the definitions in Greisen et al.⁴ (2006), we set:

- p: Pixel coordinate (abscissa)
- q: Intermediate pixel coordinate
- q1: Corrected intermediate pixel coordinate
- x: Intermediate world coordinate
- s: World coordinate

where:

$$q = p - CRPIX_m$$

$$q1 = c_0 + c_1 \times q + c_2 \times q^2 + c_3 \times q^3 + \dots$$

$$x = q1 \times CDELTA_m$$

and $CRPIX_m$, $CDELTA_m$ are the standard FITS keyword. Transformation from x to s (and back) is computed using `spcx2s()` and `spcs2x()` WCSLIB modules. These routines compute a suitable physical relation for the

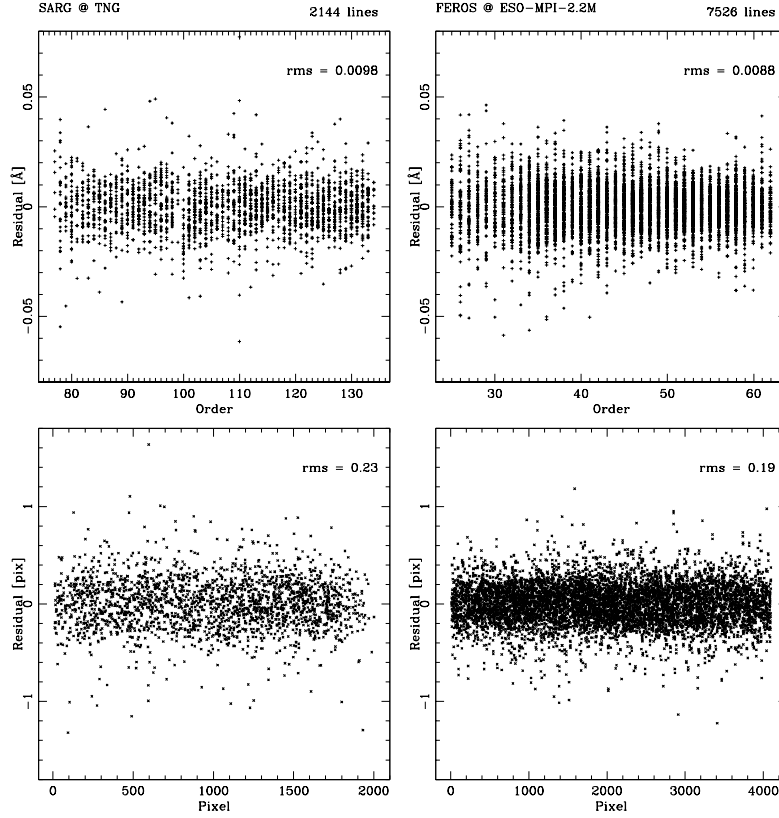


Figure 3. Left panels show the residual of the wavelength calibration for SARG@TNG echelle spectra in units of \AA (upper panel) and pixel (lower panel): 2144 lines have been identified, finding $rms \approx 0.01 \text{ \AA} \approx 0.23 \text{ pixel}$. Right panels show the same quantities for the FEROS@ESO-MPI-2.2M echelle spectra: 7526 lines are identified with a resulting $rms \approx 0.009 \text{ \AA} \approx 0.19 \text{ pixel}$.

dispersers commonly used in astronomical spectrographs, in order to define a world coordinate function and derive spectral coordinates. The relation applies to the simple case of a single disperser and under the assumption that the radiation enters perpendicular to it. The requested physical parameters for such a transformation are the grating ruling density σ , the order number m , the angle of incidence β and the $CRVAL_m$ (λ value at reference point). For each order the following starting conditions are adopted:

$$q1 = q, \quad (1)$$

$$CRVAL_m = \frac{2.0 \times \sigma \times \sin(\beta) \times \cos(\gamma)}{m}, \quad (2)$$

$$CDELTA_m = \frac{\cos(\beta) \times \cos(\gamma)^2 \times \sigma \times \text{pix_size}}{f_{cam} \times m}, \quad (3)$$

where pix_size and f_{cam} denote the detector pixel size and the camera focal length, respectively. The 1-st relation comes from the expected blaze wavelength, the 2nd from the expected linear dispersion at the central wavelength. At a first stage, for each order the task looks for the $CRPIX_m$ value which maximizes the number of matches between observed and library lines. Linearity of $m \times CRVAL_m$ vs. m and $m \times CDELTA_m$ vs. m relations is used to distinguish each time well fitted orders from bad ones. In the second stage, the $CRVAL_m$ vs. m relation is finally fixed and the task proceeds to find $CRPIX_m$ values and c_0, c_1, \dots, c_n polynomial coefficients order by order, progressively involving lines from the end of the orders, where deviations from linearity are more severe,

and increasing the degree of polinomals. Much care has been put to avoid the incidence of false line matches as much as possible. Orders correctly modelled are used to derive a global dispersion relation which is suitable to derive an accurate dispersion relation for all missing orders.

As an example, Fig. 3 shows the residuals of the wavelength calibrations for two different spectrographs, namely SARG and FEROS. In both cases the $rms \approx 0.2 \div 0.23 \text{ pixel}$, corresponding to $rms \approx 0.009 \div 0.01 \text{ \AA}$.

REFERENCES

1. Oliva E., Origlia L., Baffa C., Biliotti V., Bruno P., D'Amato F., Donati S., Falcini G., Gennari S., Ghinassi F., Giani E., Gonzalez M., Leone F., Lolli M., Lodi M., Maiolino R., Mannucci F., Marcucci G., Mochi I., Montegriffo P., Rossetti E., Scuderi S., Sozzi M., "The GIANO at TNG spectrometer", this SPIE Conference, paper 6269-46.
2. Baffa C., Biliotti V., Gennari S., Giani E., Oliva E., Origlia L., Rossetti E., "GIANO: the versatile acquisition system", this SPIE Conference, paper 6274-33.
3. Welch Brent B., "Practical Programming in Tcl/Tk", *Prentice Hall*, 2000.
4. Greisen, E.W., Calabretta, M.R. Valdes, F.G., and Allen, S.L., "Representations of spectral coordinates in FITS," *Astronomy & Astrophysics*, **446**, pp. 747-771, 2006.