Technical Report

IRAS – Image Reduction and Analysis Software

Prepared by
Lino Mastrodomenico

December 2008
Version 1.0

Version management

Project manager: S. Fineschi

Contents

Abstract

IRAS (Image Reduction and Analysis Software) is a GUI (Graphics User Interface) program for an easy conversion, calibration and manipulation of the images by the *SCORE* (Sounding CORonagraph Experiment) project cameras.

IRAS is entirely written in the *IDL* (Interactive Data Language) programming language to ease code reuse, since IDL is a popular programming language for astronomy applications, and is compatible with different IDL versions across multiple operating systems.

1.Introduction

1.1.Background

The SCORE experiment is a set of two coronagraphs designed to provide full images of the extended corona in the EUV and visible light. SCORE is part of the scientific payload of the NASA *HERSCHEL* (the Helium Resonant Scattering in the Corona and Heliosphere) sounding rocket mission, which is composed of SCORE and the Extreme Ultraviolet Imaging Telescope (EIT).

Depending on the success of the first launch, a second and a third launch are foreseen upon NASA approval.

1.2.Objectives

The main goal of the project is the development of a simple-to-use integrated conversion and calibration software for images acquired by the EUV and VL cameras of the SCORE project.

IRAS can read FITS files as well as raw images directly produced by the cameras, and can decode any information included by the camera in the headers of the raw images.

The software is also able to do basic image calibration: dark removal, shutterless correction, devignetting, photometric calibration and includes a few utility functions like multiplication and division of two arbitrary images and can build a *pB* (polarized brightness) from three or four polarized images.

Each step should be performed in a specific and documented order to obtain significant calibrated data, but IRAS also offers the flexibility to test each calculation or any subset of them on its own. This capability is often useful for quick tests before the full calibration parameters for the whole work-flow are available.

IRAS can save the final calibrated data as well, as any intermediate result, as a FITS file. It has integrated default values for many calibration parameters of the SCORE cameras, but it can also read them at runtime, at any time, from external text files with a simple syntax. The released source code includes an example parameters file with values identical to the internal defaults.

Moreover IRAS offers the possibility of opening an arbitrary number of images (limited only by available virtual memory on the system) and keeps in memory by default all intermediate results of calculations.

This allows quick tests with intermediate results without restarting a calibration sequence from the beginning when an error is made. Any loaded image or calculation result can be deleted at any time and its corresponding allocated memory is immediately freed, independent of the on which the images where loaded or generated (this is not trivial to accomplish in IDL, see the chapter “Lessons learned” for details about how this has been accomplished).

IRAS also automatically promotes internally the data from 16 bits integers to 32 bits or to floating points as required to keep it accurate based on the exact sequence of operations executed by the end users.

2.Architecture overview

IRAS is written entirely in IDL and the source distribution includes a few utility programs used for its development written in Python or as bash scripts. These utilities are not required by end users running IRAS but they are useful for its development.

Their necessity steams from the requirements that IRAS should be easily usable and be able to run and being modified on IDL implementations by ITT Visual Information Solutions, including installations using trial “personal-use-only” licenses.

Unfortunately these are somewhat conflicting goals: since IRAS is a medium-sized project (the IDL source code alone is near 80 kiB, spread over a few thousand lines of code), it’s highly desirable that its code should be split over several source files, to ease code development and maintenance. But trial IDL versions from ITT VIS don’t offer the possibility of packing multiple source files together for easy distribution. For other IDL programs this usually means that end users have to manually change the current directory to the one containing the source files from within the IDL development environment, possibly recompile them if one has been changed and run an helper script each time the program must be run.

This is clearly suboptimal because it puts an heavy burden on end users for something that ideally should be as simple as clicking a program icon. The overall root of the problem is that apparently there’s no way for an IDL program to automatically determine its source file location on the file system and hence the location of the other program modules and any data file required (e.g. icons or configuration defaults).

IRAS solves this problem by keeping its source code neatly split over multiple files to ease maintenance, but it also includes a small utility *build.py* that merges all files required to run IRAS in a single stand-alone IDL file, *iras.pro*. This file includes all the IRAS source, an helper function (also called *iras*) and also all the program icons, which usually reside in the *data* directory, converted to IDL functions that return 2D (for grayscale images) or 3D (for color icons) matrices identical to the corresponding images as read by IDL.

The end result is a single file, *iras.pro*, that the end user can start from the command line with the command “idl -e iras” or open from the IDL integrated development environment and run right away without the need for changing the current directory or recompiling the project.

IRAS is used by users running multiple ITT VIS IDL versions, including at least IDL 6.3, 7.0 and 7.1, on different operating system (GNU/Linux, Mac OS X and Microsoft Windows). It should be able to run on any combination of these platforms (see the chapter “Compatibility” below for further details on this subject).

3.The choice of the programming language

IDL has been chosen as the programming language for IRAS for a number of reasons;

- it’s the only programming language know by all members of the SCORE project that contributed code or that are expected to be able to read it and potentially change it in the future;
- it’s the language used by previous programs in SCORE and related projects;
- it’s a quite popular language in the astronomy community; while this seems to be increasingly less true, there’s still an huge body of useful astronomy-related functions and libraries available on-line, often usable under free software/open source licenses.

Unfortunately there are a few drawbacks:

- IDL was created in 1977 and the core languages has not evolved much since then, this implies a very verbose syntax that often allows sloppy programming styles; efforts have been made to keep a clean and consistent programming style over IRAS with correct indentation;
- IDL lacks any high level data types, even basic ones offered by all modern programming languages like resizeable lists, lists that can contain arbitrary objects and associative arrays; as a matter of fact one of the more useful long-term results of the IRAS development is probably the implementation of a fast resizeable list that can contain any IDL data type;
- the IDL memory model (including its latest “OOP” additions) is **abysmally awful**, absolutely insufficient for the development of complex GUI programs and barely above the functionality offered by the standard C libraries; it’s probably not a coincidence that IDL is one of the very few languages that don’t have an integrated development environment written in the language itself (the standard IDL IDE is written in Java); see the chapter “Lessons learned” for more details on this subject;
- IDL dumps all functions and classes, including the ones from the standard library, any loaded third-party library and the program itself, in a single global namespace, making name collisions relatively frequent and very dangerous, especially in medium-to-large programs and making forward compatibility to new IDL versions (which can include new functions) more a wishful hope than anything that can be counted on;
- by default IDL uses only 16 bits for integers and, what’s worse, the overflows are ignored and silently return the wrong result, even for simple calculations where the need for more than 16 bits can be detected at compile time;
- the IDL GUI libraries, while in theory cross-platform, have different quirks and small incompatibilities between operating systems; moreover the ITT VIS implementation on GNU/Linux uses an obsolete (and quite ugly) graphic toolkit; see “Compatibility and quirks” below for more details.

Overall the experience with this project suggests that for future development of new programs other languages should be carefully considered. A language that seems to be increasingly popular in astronomy is Python coupled with the SciPy library. Tools exist to convert IDL programs to Python and the Harvard University and the Space Telescope Science Institute offer on-line reference guides with equivalent Python constructs for IDL users.

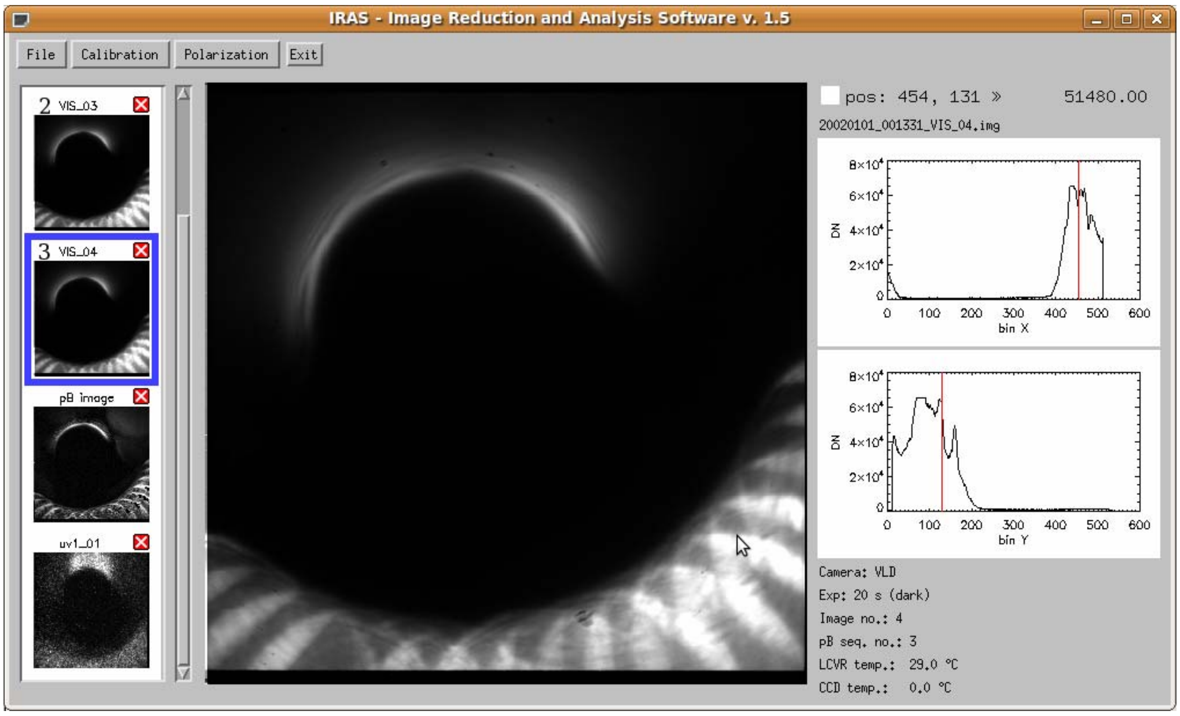
3.1.Compatibility and quirks

There are a number of incompatibilities, small quirks and outright bugs in the ITT VIS IDL implementation on different operating systems. A common example is inconsistent functionality in the graphics widgets on different operating systems. There are a number of workarounds for them in IRAS; they are well documented with comments and can be found grepping the source for “quirk” and “workaround”.

An important and often overlooked limitation of IDL 6.3 is a very small maximum line length for source code. If there are lines that go over about 100 characters, they should be carefully tested in IDL 6.3.

4.User manual

4.1.GUI overview



pos: 454, 131 » 51480.00

20020101_001331_VIS_04.img

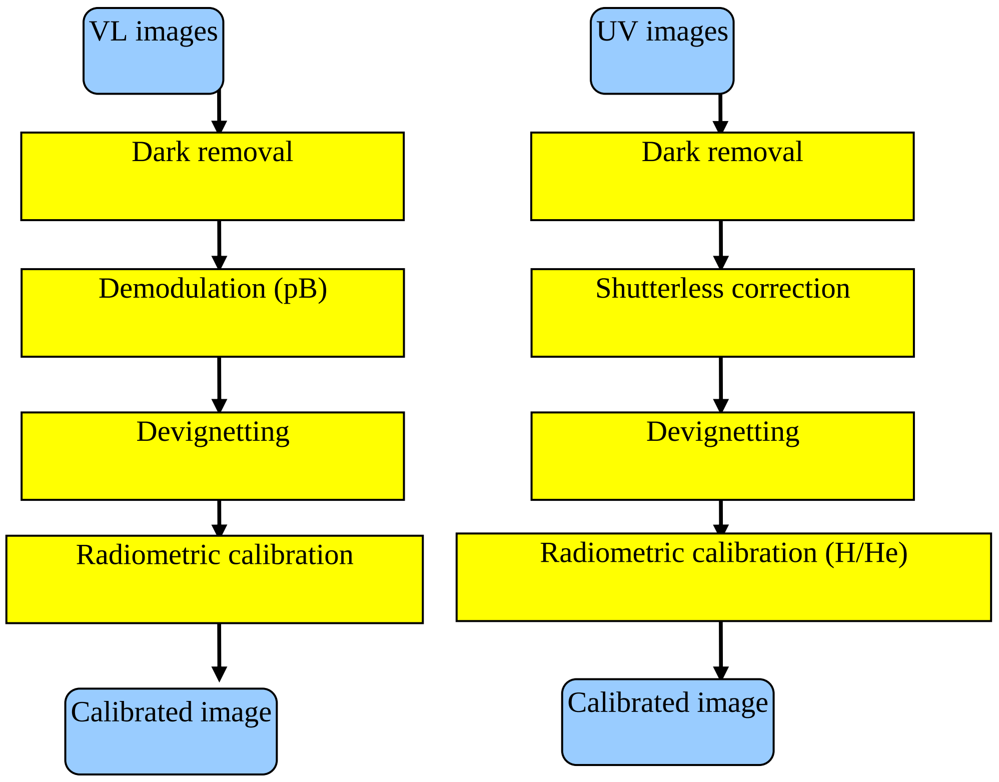
Camera: VLD
Exp: 20 s (dark)
Image no.: 4
pB seq. no.: 3
LCVR temp.: 29.0 °C
CCD temp.: 0.0 °C

The main window in the IRAS GUI is composed by:

1. the menu near the top of the window;
2. the thumbnail column, along the left side of the window;
3. the currently selected image, at the center of the window;
4. the details panel on the right hand side of the window.

An unlimited number of images can be loaded at the same time, the current one can be selected by clicking on the corresponding thumbnail.

4.2.Standard calibration sequences



4.3.Dependencies

IRAS requires a working IDL environment (mostrly tested with ITT IDL 7.06, but IDL 6.3 and 7.1 are also know to work) and a subset of the SolarSoft libraries. They can be installed by unpacking the file solarsoft_iras-1.zip, provided alongside IRAS, in a new directory under the subdirectory "lib" of the IDL installation (e.g. "C:\Programmi\ITT\IDL70" under Windows).

Alternatively they can be installed in any directory as long their current location is added to the !PATH directories in the IDL preferences.

4.4.Running IRAS

Unpack the file iras-x.yz.zip in a new directory.

On Microsoft Windows: start the IDL development environment, open the file *iras.pro* and start IRAS with the command "iras" or by click on the "run" green icon.

On other operating systems its also possible to simply double click on the file *iras.sh*.

4.5.Menus

The menu is composed by three submenus ("File", "Calibration" and "Polarization"), and a "Exit" button that causes to application to exit immediately.

4.5.1.File Menu

The “File” menu contains the following commands:

- “Read RAW”, read raw files (*.img) produced by the SCORE cameras; LCVR sequence numbers associated with images produced by the VL detector are automatically recognized and displayed as a small number (from 1 to 4) near the image thumbnail;
- “Read FITS”, read an image from a FITS file;
- “Read parameters”, read new calibration parameters from a text file; see the file *default_conf.txt* for an example of the correct file format;
- “Save as FITS”, saves the currently selected image as a FITS file;
- “Exit”, exits the program immediately.

4.5.2.Calibration Menu

The “Calibration” menu contains the following commands:

- “Remove dark count”, subtracts the selected image (dark) from the current one (the data frame);
- “Shutterless correction”, apply a shutterless correction for images acquired with the SCORE UV detector to the current image;
- “Remove vignetting”, two different algorithms that apply a devignetting filter to the current image;
- “Photometric calib”, three different calibrations for H, He and VL images;
- “Multiply/divide images” applies the corresponding operation to every pixel of two selected images.

4.5.3.Polarization Menu

The “Polarization” menu contains only a single command:

- “Build polarized image”, creates a pB image, requires three or four images with different polarizations. If only three images are supplied, the angle corresponding to the missing one must be signaled by pressing the red “skip” icon.

5.Lessons learned

Apart from the small IDL graphic limitation and quirks, perhaps the most important detail to pay attention to when building any non-trivial IDL program is the memory allocation. This is due to the very limited IDL memory management capabilities is especially true for most IDL GUI applications and any program that must load variable amount of data or need a flexible work-flow.

Common symptoms of these problems is IDL programs that are limited to a fixed number of data sets loaded in memory at the same time (e.g. earlier IRAS versions where limited to a maximum number of 5 images in memory, making them impractical to use in a real-world scenario) or heavily leak memory due to hard-to-find bugs, often requiring the end user to regularly close and reopen the IDL development environment to free the excess memory used.

Details about the IDL limitations that cause these problems and suggested workarounds and best practices are documented in the following sections.

5.1.The IDL memory model

IDL has two different memory management modes, one for OOP class instances and one for everything else (pretty much basic IDL data types).

They are detailed in the IDL manual, but what's interesting to note is that what they both have in common is that IDL doesn't keep automatically track and frees memory when there's the possibility of more than one reference at the same time to the same physical memory block.

This can be the result of the influence of implementation limitations on the language design. Proper memory management with correct reference counting and/or automatic garbage collection is very hard, and despite the official IDL documentation mentioning "reference counting" apparently IDL only keeps a single "free/used" status bit on all automatically-freed variables and leaves to the programmer the burden to free class instances or objects referenced by pointers.

This leads to a number of undesirable results: all assignments always create a copy of an object, which can be suboptimal if the object is e.g. several megabytes in size, assignments to a slice don't modify the original array, there the possibility of memory leaks or null pointer dereferences if the programmer is not extremely careful when using classes and pointers, which unfortunately are strictly required to overcome the aforementioned limitations of the basic types in non-trivial programs.

5.2.Best practices in IDL OOP and memory allocation

Never, under any circumstance use the command *heap_free* in IDL OOP.

Ever.

This advice is very non-obvious since the official IDL documentation and well-respected sources in books and on-line articles suggest its use to be sure to free all memory used by an instance, even allocated deep in a complex object tree. On first sight, the alternative, *ptr_free*, seems to be more prone to errors since the programmer must carefully call it, in the correct order, for every manually allocated object in a class instance.

But *heap_free* is nonetheless not an option for a very basic design flaw: it completely ignores object boundaries and happily free the memory of all nested even if it belongs to instances of a different class that may not have been properly deallocated yet and/or objects that are still used elsewhere in the program (remember, class instances are the only variables in IDL that can have multiple references to the same object).

This means that *heap_free* completely violates three principles of object oriented programming: abstraction, encapsulation and decoupling. It's very important to understand that this is not only an abstract purity problem, but it has practical consequences: e.g. a change in a program that seems to work perfectly fine can cause unexpected and very hard to debug problems in a completely unrelated part of the code, because *heap_free* has prematurely an inner object that other parts of the program are still using. And a somewhat even worse consequence is that when the cause of the bug is found, the most obvious way to “fix” it is often postponing the call to *heap_free* or removing it altogether. Again this results in a program that apparently “works” but that leaks memory, a bug that is usually very hard to find and fix properly.

Even when used correctly *heap_free* is a bug waiting to happen during future program maintenance.

Experience with many IDL programs suggests that the end result is that the burden of dealing with memory leaks often shifts on the end user that faces programs that use an increasing amount of system memory when it’s restarted multiple times without closing the IDL development environment or even over the course of a single run of the program. End users are often trained to consider this inevitable and to close and reopen IDL each time a program is launched.

The right way to face this problem, short of switching to a high-level programming language, is a careful use of *ptr_free*. As a basic rule, each command that allocate memory (e.g. object that are pointed to by a *ptr_new*, but not the *ptr_new* object itself, unless it’s in turn referenced by another pointer) needs a corresponding *ptr_free*, and each command that allocates a new object (*obj_new*) needs a corresponding *obj_destroy* and the class is then responsible for defining a proper *cleanup* method that in turn usually calls one or more times *ptr_free*.

This is very low-level memory management, similar to the one required by C or similar languages, and it’s very hard to get right even for simple cases without an extremely good understanding of IDL memory management and lots of tests.

So it’s strongly suggested to hide all these details in completely encapsulated classes, leaving to the rest of the code only the lesser problem of calling *obj_destroy* for each allocated class instance.

A good and completely commented, debugged and tested implementation of these principles is the *list* class in the *list.pro* file. It can be used as an example of how to handle a moderately complex and dynamic memory structure and it’s also useful per se since it provides an array that can contain arbitrary IDL variables (including nested lists), can be empty and offers linear-time amortized behaviour (i.e. very fast, O(1), independent of the number and the size of the variables contained in the list) for appends and element removal at the end of the list.

6.License

IRAS includes code contributions from at least Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi, Maurizio Pancrazzi and is copyrighted by at least the Osservatorio Astronomico di Torino, Lino Mastrodomenico and Maurizio Pancrazzi.

IRAS is Free Software (open source) and is currently distributed under the GNU General Public License (version 3 or later). This license has been chosen because it encourages public diffusion of source code for scientific applications.

No restrictions at all are imposed on modified IRAS versions that aren't distributed to third-parties.

7.Conclusions

In this short report, the main features of the IRAS system have been presented. In particular, it has been shown how its architecture gives high flexibility to the software, allowing to load and keep in memory an arbitrary number of images and intermediate results, making it relatively easy to use and resulting in a fast work-flow even for complex scientific calibration sequences.

IRAS can also be used on different operating systems and IDL versions. These result has been obtained by using a number of programming techniques, as illustrated in previous chapters of this report.

8.Appendix A - Source code

8.1.default_conf.txt

```
# IRAS configuration parameters, default values

uv_exp = 40  # UV images exposure time in seconds

# photometric calibration:
# Units: [(ph/cm^2/s/sr)/(DN/s/bin))]
photometric_VL = 1.8e+11  # 09.04.02 data
photometric_H = 5e+9      # estimated from component-level measurements
photometric_He = 7.9e+8   # 09.04.02 data

# Conversion Digital Number (DN)/bin -> ph/cm^2/s/sr.
# E.g.: DN * photometric_He/uv_exp

# occulter center position, (0, 0) is the *lower* left corner:
xc = 230
yc = 268
```

8.2.iras_gui.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

; IRAS (a.k.a. SDDAS - SCORE DDAS - SCORE Data Display and Analysis Software)
;
; Aug 23 2007 version 0.0
; Nov  6 2008 version 0.1
; Nov 27 2008 version 0.2
; Dec 11 2008 version 0.3
; Jan 20 2009 version 0.4
; Jan 29 2009 version 0.5
; Feb 13 2009 version 0.6
; Feb 23 2009 version 0.7
; Mar  6 2009 version 0.8
; Mar 18 2009 version 0.9
; Mar 23 2009 version 0.10
; Mar 23 2009 version 0.11
; Mar 23 2009 version 0.12
; Apr 17 2009 version 0.13
; May  8 2009 version 0.14
; Jun  1 2009 version 0.50
; Jun  5 2009 version 0.51
; Jun  9 2009 version 0.52
; Jun 10 2009 version 0.53
; Jun 12 2009 version 0.54
; Jun 12 2009 version 0.55
; Jun 12 2009 version 0.56
; Jun 17 2009 version 0.57
; Jun 19 2009 version 0.58
; Jun 19 2009 version 0.59
; Jun 25 2009 version 0.60
; Jun 29 2009 version 0.61
; Jul  7 2009 version 0.62
; Jul  8 2009 version 0.63
; Jul 10 2009 version 0.64
; Jul 10 2009 version 0.65
; Jul 10 2009 version 0.66
; Jul 20 2009 version 0.67
; Jul 22 2009 version 0.68
; Oct  1 2009 version 1.0
; Oct  9 2009 version 1.1
; Oct  9 2009 version 1.2
; Nov  6 2009 version 1.3
; Nov 19 2009 version 1.4
; Nov 20 2009 version 1.5

pro iras_gui
    common data_structures, last_path, imgs, widgets

    version = '1.5'  ; keep this updated!
    app_name = 'IRAS - Image Reduction and Analysis Software'

    ; workaround for an IDL 7.0 bug on X (Linux), see:
    ;     http://www.dfanning.com/misc_tips/window_retain.html
    device, retain=2

    device, decomposed=1  ; force 24 bit colors
    !p.background = 'ffffff'x  ; white
    ;!p.color = 0  ; apparently this doesn't do anything

    disp_size = 520  ; main image size in pixels
    thumb_size = 100  ; thumbnails image size
    ;max_images = 0  ; max number of images loaded at the same time
    ;thumbs_total_height = (thumb_size + 26) * max_images + 10
    ;thumbs_total_height = 1
    last_path = '.'

    ; workarounds for an IDL GUI bugs/quirks
    idl_quirk1 = 2
    case !version.os_family of
        'unix': idl_quirk2 = 29
        else: idl_quirk2 = 0
    end

    imgs = {data: list(), digits: list(), warning: get_image('warning'), $
            skip: get_image('skip'), current_img: -1}  ; -1 means no image
    for i = 0, 4 do begin
        imgs.digits->append, get_image('t' + string(i, format='(I1)'))
    end

    window = widget_base(title=app_name + ' v. ' + version, /column)

    ; main menu
    menu = widget_base(window, /row)

    file_menu = widget_button(menu, VALUE='File', /MENU)
    desc = ['0\Read RAW...', '0\Read FITS...', '0\Read parameters...', $
            '0\Save as FITS...', '2\Exit']
    dummy = cw_pdmenu(file_menu, desc, /MBAR, /RETURN_FULL_NAME, UVALUE='menu')

    calibration_menu = widget_button(menu, VALUE='Calibration', /MENU)
    desc = ['0\Remove dark count', $  ; FIXME: add back this: '0\Remove flat
field', $
            '0\Shutterless corrrection', '0\Remove vignetting', $
            '0\Remove vignetting 2', '0\Photometric calib VL', $
            '0\Photometric calib H', '0\Photometric calib He', '0\Multiply
images', '0\Divide images']
    dummy = cw_pdmenu(calibration_menu, desc, /MBAR, /RETURN_FULL_NAME,
UVALUE='menu')

    polarization_menu = widget_button(menu, VALUE='Polarization', /MENU)
    desc = ['0\Build polarized image']
    dummy = cw_pdmenu(polarization_menu, desc, /MBAR, /RETURN_FULL_NAME,
UVALUE='menu')

    ;dummy = widget_button(menu, value='Help', uvalue='Help')
    dummy = widget_button(menu, value='Exit', uvalue='Exit')

    ; main window content, below the menu
    display = widget_base(window, /row)
    col1 = widget_base(display, /column)  ; thumbnails
    col2 = widget_base(display, /column)  ; main image display
    col3 = widget_base(display, /column)  ; misc infos

    ; thumbnails
    ;thumbs = widget_draw(col1, /button_events, uvalue='thumbs_click', $
    ;                     xsize=thumb_size + 10 * 2, ysize=thumbs_total_height, $
    ;                     x_scroll_size=thumb_size + 10 * 2 + idl_quirk1, $
    ;                     y_scroll_size=disp_size - idl_quirk2)
    thumbs = widget_draw(col1, /button_events, uvalue='thumbs_click', $
                        xsize=1, ysize=1, $
                        x_scroll_size=thumb_size + 10 * 2 + idl_quirk1, $
                        y_scroll_size=disp_size - idl_quirk2)

    ; main image area at the center of the window
    main_image = widget_draw(col2, /button_events, /motion_events,
uvalue='main_image_click', $
                             xsize=disp_size, ysize=disp_size)

    ; rightmost column, contains misc infos
    info = widget_base(col3, /row)
    info_icon = widget_draw(info, /button_events, uvalue='info_icon', $
                            xsize=16, ysize=16)

    case !version.os_family of  ; FIXME: test this shit on win/linux/macos
        ; see /usr/share/fonts/X11/misc/fonts.alias for a list of GNU fonts
        'unix': font = '9x15'
        else: font = '15'
    end
    info_bar = widget_label(info, value=' ', /dynamic_resize, font=font)
    img_name = widget_label(col3, value=' ', /dynamic_resize, /align_left)
    ;print, WIDGET_INFO(info_bar, /FONTNAME)

    profile_x = widget_draw(col3, xsize=296, ysize=180)
    profile_y = widget_draw(col3, xsize=296, ysize=180)

    widgets = {camera_info: list(), status: 0, disp_size: disp_size, $
              thumb_size: thumb_size, thumbs: thumbs, prev_image: -1, $
```

```
               prev_name: '', pb1: -1, pb2: -1, pb3: -1, pb4: -1, $
               pb_skipped: 0, main_image: main_image, info_bar: info_bar, $
               info_icon: info_icon, img_name: img_name, $
               profile_x: profile_x, profile_y: profile_y, $
               close_icon: get_image('close_small'), $
               selected: get_image('selected'), $
               not_selected: get_white_rect(116, 132), $
               uv_exp: 0.0, photometric_VL: 0.0, photometric_H: 0.0, $
               photometric_He: 0.0, xc: 0.0, yc: 0.0}

    for i = 1, get_raw_header_info_size() do begin
        ; WARNING: value must be a single space not an empty string, otherwise
        ;     the widget will disappear changing the vertical space it uses,
        ;     which seems undesiderable
        widgets.camera_info->append, widget_label(col3, value=' ',
/dynamic_resize, /align_left)
    end

    widget_control, window, /realize

    ; try to deuglify the white dot
    gui_resize_thumbnails_area, widgets, 1, -1

    ; this is necessary on Windows, otherwise the widget_draws are black
    gui_clean_all, widgets
    gui_select_widget, widgets.info_icon
    erase

    ;read_conf, 'default_conf.txt', widgets
    read_default_conf, widgets

    ; FIXME: use the cleanup parameter for xmanager to register a procedure
    ;     that frees all the memory (mostly lists, maybe also things in common)
    ;     to be friendlier for users that use IRAS from IDLDE without IDL
    ;     restarts between IRAS restarts???
    xmanager, 'iras_gui', window, event_handler='handle_gui_event', /no_block
end

pro reset_status, widgets
    widgets.status = 0
    widget_control, widgets.info_bar, set_value=' '
    gui_select_widget, widgets.info_icon
    erase
end

pro handle_gui_event, ev
    common data_structures, last_path, imgs, widgets

    ;help, ev, /struct
    widget_control, ev.id, get_uvalue=uvalue
    redirect = 0

    if uvalue eq 'thumbs_click' then begin
        gui_select_widget, widgets.thumbs
        thumb_size = widgets.thumb_size
        if ev.release eq 1 then begin
            ; unlike the code below (for selecting a new image), when closing
            ; an image we act only on release of the left button, to be really
            ; sure that the user meant it
            close_x = 10 + thumb_size - 14
            ystep = thumb_size + 26
            ;print, !d.y_size, ev.x, ev.y
            if ev.x ge close_x && ev.x lt close_x + 14 then begin
                ; Here we use "!d.y_size - 1", while get_y_from_top uses only
                ; "!d.y_size". This is necessary to use "ge" and "lt" with y
                ; below (otherwise "gt" and "le" would have been necessary).
                ; IOW we need to consider the pixel under the mouse a square
                ; with height 1 to avoid off-by-one errors.
                ; FIXME: maybe simply use get_y_from_top(ev.y, 1) ???
                y = !d.y_size - 1 - ev.y
                n = y / ystep
                y mod= ystep
                if y ge 10 && y lt 24 then begin
                    ;print, ev.x - close_x, y, n
                    if n lt imgs_get_len(imgs) then begin
                        reset_status, widgets
                        imgs_delete_image, imgs, n, widgets
                    end
                end
            end
        end
        if ev.press eq 0 then begin
            ; 1, 2, 4 means press of left, center and right button).
            ; When selecting an image we act on click
            ; instead of release to make things feel a bit snappier (IDL is
            ; veeeeery slow)
            ; FIXME: is this correct?
            return
        end
        ; FIXME: the code for calculating q[0] seems seriously brain-damaged
        nimages=!D.y_size/(thumb_size+26)
        id=INDGEN(nimages)
        ypositions=!D.y_size-(thumb_size+26)*(id+1)

        q=WHERE(ypositions lt ev.y AND (ypositions+thumb_size) gt ev.y, cnt)
        if cnt ne 0 then begin
            n = q[0]
            if n lt imgs_get_len(imgs) then begin
                imgs_select_image, imgs, n, widgets
                if widgets.status eq 1 then begin
                    redirect = 1
                    uvalue = 'menu'
                    ev = {value: 'Remove dark count'}
                end else if widgets.status ge 2 && widgets.status le 5 then begin
                    pb_skip = 0
                    redirect = 1
                    uvalue = 'menu'
                    ev = {value: 'Build polarized image'}
                end else if widgets.status eq 6 then begin
                    redirect = 1
                    uvalue = 'menu'
                    ev = {value: 'Multiply images'}
                end else if widgets.status eq 7 then begin
                    redirect = 1
                    uvalue = 'menu'
                    ev = {value: 'Divide images'}
                end
            end
        end
        if redirect eq 0 then begin
            return
        end
    end

    if uvalue eq 'info_icon' then begin
        if ev.release eq 1 && widgets.status ge 2 && widgets.status le 5 &&
widgets.pb_skipped eq 0 then begin
            pb_skip = 1
            redirect = 1
            uvalue = 'menu'
            ev = {value: 'Build polarized image'}
        end else begin
            return
        end
    end

    if uvalue eq 'main_image_click' then begin
        ;if ev.release eq 1 then begin
        if ev.press eq 1 then begin
            ; see the description above about ev.press
            if imgs_get_len(imgs) ne 0 then begin
                draw_profile, widgets, imgs_get_current_image(imgs), ev.x, ev.y
            end
        end
        ; this is a click or a motion event
        if widgets.status eq 0 && imgs_get_len(imgs) ne 0 then begin
            sz = size(imgs_get_current_image(imgs))
            w1 = sz[1] - 1
            h1 = sz[2] - 1
            disp_size1 = widgets.disp_size - 1
            x = round(float(ev.x) * w1 / disp_size1)
            y = round(float(ev.y) * h1 / disp_size1)
            ; yes, the following check may file (e.g. during drag&drop)
            if x ge 0 && x le w1 && y ge 0 && y le h1 then begin
                ; FIXME: make this (and other string format elsewhere) simpler
                right_guillemet = string(byte(187))  ; implicit Latin-1
encoding???
                s = ('pos:' + string(x, format='(I4)') + ',' + $
                     string(y, format='(I4)') + ' ' + right_guillemet + ' ' + $
                     string(float((imgs_get_current_image(imgs))[x, y]),
format='(G13)'))
                widget_control, widgets.info_bar, set_value=s
            end
        end
        return
    end

    if redirect eq 0 then begin
        reset_status, widgets
    end

    if uvalue eq 'Help' then begin
        ; FIXME: add inline help?
        return
    end

    if uvalue eq 'Exit' then begin
        widget_control, ev.top, /destroy
        return
    end

    if uvalue eq 'menu' then begin
        if ev.value eq 'Exit' then begin
            widget_control, ev.top, /destroy
            return
        end

        if ev.value eq 'Read RAW...' then begin
            bak_path = last_path
            file = dialog_pickfile(filter=['*.img', '*.bin', '*.raw'], $
                                   /read, path=last_path, get_path=last_path)
            name = file_basename(file)
            short_name = strmid(name, strlen(name) - 10, 6)
            if file eq '' then begin
                last_path = bak_path  ; workaround for an IDL bug
                return
            end
            image = raw_read_image(file)
            ;header = get_vld_header(file)
            ;camerainfo = decode_raw_header(file)
            img_info = decode_raw_header(file)
            ;camerainfo = img_info.camerainfo
            ;help, header, img_info.seqnum

            ; convert the image from unsigned 16 bit to signed 32 bit, to avoid
            ; errors in later calculations (e.g. dark removal)
            ; FIXME: this should really be done by each calculation routine
            ;     that needs it!
            image = long(image)
            imgs_append, imgs, image, name, short_name, widgets, 0, img_info
            return
        end

        if ev.value eq 'Read FITS...' then begin
            bak_path = last_path
            file = dialog_pickfile(filter=['*.fits', '*.fit'], /read, $
                                   path=last_path, get_path=last_path)
            name = file_basename(file)
            short_name = strmid(name, strlen(name) - 11, 6)
            if file eq '' then begin
                last_path = bak_path  ; workaround for an IDL bug
                return
            end
            ; FIXME: do something with the FITS header?
            fits_read, file, image, hdr

            imgs_append, imgs, image, name, short_name, widgets, -1  ; FIXME:
read exposure from FITS header?
            return
        end

        if ev.value eq 'Read parameters...' then begin
            bak_path = last_path
            file = dialog_pickfile(filter=['*.txt'], $
                                   /read, path=last_path, get_path=last_path)
            if file eq '' then begin
                last_path = bak_path  ; workaround for an IDL bug
                return
            end
            read_conf, file, widgets
            return
        end

        if ev.value eq 'Save as FITS...' then begin
            ;if not ia_image_exists(images, current_image) then begin
            if imgs_get_len(imgs) eq 0 then begin
                dummy = dialog_message('No image loaded', /error)
                return
            end
            ;image = ia_get_image(images, current_image)
            image = imgs_get_current_image(imgs)
            bak_path = last_path
            file = dialog_pickfile(filter=['*.fits', '*.fit', '*.*'], /write, $
                                   path=last_path, get_path=last_path)
            if file eq '' then begin
                last_path = bak_path  ; workaround for an IDL bug
                return
            end
            fits_write, file, image
            return
        end

        if ev.value eq 'Shutterless corrrection' then begin
            ;if not ia_image_exists(images, current_image) then begin
            if imgs_get_len(imgs) eq 0 then begin
                dummy = dialog_message('No image loaded', /error)
                return
            end
            ;image = ia_get_image(images, current_image)
            image = imgs_get_current_image(imgs)
            exptime = 4.0
            shutless2, image, exptime, imageout

            name = 'shutterless of (' + imgs_get_current_name(imgs) + ')'
            short_name = 'shutterless'
            imgs_append, imgs, imageout, name, short_name, widgets,
imgs_get_current_exp(imgs)
            return
        end
        if ev.value eq 'Remove dark count' && widgets.status eq 0 then begin
            if imgs_get_len(imgs) eq 0 then begin
                dummy = dialog_message('No image loaded', /error)
                return
            end
            widgets.status = 1
            widget_control, widgets.info_bar, set_value='Select dark'
            widgets.prev_image = imgs.current_img
            widgets.prev_name = imgs_get_current_name(imgs)
            gui_select_widget, widgets.info_icon
            tv, imgs.warning, true=1
            return
        end

        if ev.value eq 'Remove dark count' then begin
            reset_status, widgets

            ;if not ia_image_exists(images, 0) then begin
            if 0 then begin  ; FIXME!!!
                dummy = dialog_message('Image not loaded', /error)
                return
            end
            ;if not ia_image_exists(images, 1) then begin
            if 0 then begin  ; FIXME!!!
                dummy = dialog_message('Dark not loaded', /error)
                return
            end
            ;image = imgs_get_image(imgs, 0) - imgs_get_image(imgs, 1)
            image = imgs_get_image(imgs, widgets.prev_image) -
imgs_get_current_image(imgs)

            name = 'dark (' + imgs_get_current_name(imgs) + ') removed from (' +
widgets.prev_name + ')'
            short_name = 'dark removed'
            imgs_append, imgs, image, name, short_name, widgets,
imgs_get_exp(imgs, widgets.prev_image)
            return
        end

        if ev.value eq 'Multiply images' && widgets.status eq 0 then begin
            if imgs_get_len(imgs) eq 0 then begin
                dummy = dialog_message('No image loaded', /error)
                return
            end
            widgets.status = 6
            widget_control, widgets.info_bar, set_value='Select multiplier'
            widgets.prev_image = imgs.current_img
            widgets.prev_name = imgs_get_current_name(imgs)
            gui_select_widget, widgets.info_icon
            tv, imgs.warning, true=1
            return
        end

        if ev.value eq 'Multiply images' then begin
            reset_status, widgets

            ;if not ia_image_exists(images, 0) then begin
            if 0 then begin  ; FIXME!!!
                dummy = dialog_message('Image not loaded', /error)
                return
            end
            ;if not ia_image_exists(images, 1) then begin
            if 0 then begin  ; FIXME!!!
                dummy = dialog_message('Dark not loaded', /error)
                return
            end
            ;image = imgs_get_image(imgs, 0) * imgs_get_image(imgs, 1)
            image = float(imgs_get_image(imgs, widgets.prev_image)) *
imgs_get_current_image(imgs)
```

```
            name = 'multiply (' + imgs_get_current_name(imgs) + ') by (' +
widgets.prev_name + ')'
            short_name = 'multiply'
            imgs_append, imgs, image, name, short_name, widgets,
imgs_get_exp(imgs, widgets.prev_image)
            return
        end

        if ev.value eq 'Divide images' && widgets.status eq 0 then begin
            if imgs_get_len(imgs) eq 0 then begin
                dummy = dialog_message('No image loaded', /error)
                return
            end
            widgets.status = 7
            widget_control, widgets.info_bar, set_value='Select divisor'
            widgets.prev_image = imgs.current_img
            widgets.prev_name = imgs_get_current_name(imgs)
            gui_select_widget, widgets.info_icon
            tv, imgs.warning, true=1
            return
        end

        if ev.value eq 'Divide images' then begin
            reset_status, widgets

            ;if not ia_image_exists(images, 0) then begin
            if 0 then begin  ; FIXME!!!
                dummy = dialog_message('Image not loaded', /error)
                return
            end
            ;if not ia_image_exists(images, 1) then begin
            if 0 then begin  ; FIXME!!!
                dummy = dialog_message('Dark not loaded', /error)
                return
            end
            ;image = imgs_get_image(imgs, 0) * imgs_get_image(imgs, 1)
            image = float(imgs_get_image(imgs, widgets.prev_image)) /
imgs_get_current_image(imgs)

            name = 'divide (' + imgs_get_current_name(imgs) + ') by (' +
widgets.prev_name + ')'
            short_name = 'divide'
            imgs_append, imgs, image, name, short_name, widgets,
imgs_get_exp(imgs, widgets.prev_image)
            return
        end

        if ev.value eq 'Remove vignetting' then begin
            ; FIXME: ask the operator the center
            if imgs_get_len(imgs) eq 0 then begin
                dummy = dialog_message('No image loaded', /error)
                return
            end
            ;if not imgs_image_exists(imgs) then begin
            ;    dummy = dialog_message('No image loaded', /error)
            ;    return
            ;end
            ;image = remove_vignetting(ia_get_image(images, current_image), 512,
467, 531)
            image = remove_vignetting(imgs_get_current_image(imgs), 512, 467,
531)

            name = 'vignetting removed from (' + imgs_get_current_name(imgs) +
')'
            short_name = 'vignetting'
            imgs_append, imgs, image, name, short_name, widgets,
imgs_get_current_exp(imgs)
            return
        end
        if ev.value eq 'Remove vignetting 2' then begin
            ; FIXME: ask the operator the center
            if imgs_get_len(imgs) eq 0 then begin
                dummy = dialog_message('No image loaded', /error)
                return
            end
            ;if not imgs_image_exists(imgs) then begin
            ;    dummy = dialog_message('No image loaded', /error)
            ;    return
            ;end
            ;image = remove_vignetting2(ia_get_image(images, current_image), 512,
467, 531)
            ;image = remove_vignetting2(imgs_get_current_image(imgs), 512, 467,
531)
            image = remove_vignetting2(imgs_get_current_image(imgs), 512,
widgets.xc, widgets.yc)

            name = 'vignetting removed from (' + imgs_get_current_name(imgs) +
')'
            short_name = 'vignetting'
            imgs_append, imgs, image, name, short_name, widgets,
imgs_get_current_exp(imgs)
            return
        end

        if ev.value eq 'Photometric calib VL' then begin
            if not imgs_image_exists(imgs) then begin
                dummy = dialog_message('No image loaded', /error)
                return
            end
            exp = imgs_get_current_exp(imgs)
            help, exp
            image = imgs_get_current_image(imgs) * (widgets.photometric_VL / exp)
            name = 'VL calib. of (' + imgs_get_current_name(imgs) + ')'
            short_name = 'VL calib.'
            imgs_append, imgs, image, name, short_name, widgets, -1
            return
        end

        if ev.value eq 'Photometric calib H' then begin
            if not imgs_image_exists(imgs) then begin
                dummy = dialog_message('No image loaded', /error)
                return
            end
            image = imgs_get_current_image(imgs) * (widgets.photometric_H /
widgets.uv_exp)
            name = 'H calib. of (' + imgs_get_current_name(imgs) + ')'
            short_name = 'H calib.'
            imgs_append, imgs, image, name, short_name, widgets, -1
            return
        end

        if ev.value eq 'Photometric calib He' then begin
            if not imgs_image_exists(imgs) then begin
                dummy = dialog_message('No image loaded', /error)
                return
            end
            image = imgs_get_current_image(imgs) * (widgets.photometric_He /
widgets.uv_exp)
            name = 'He calib. of (' + imgs_get_current_name(imgs) + ')'
            short_name = 'He calib.'
            imgs_append, imgs, image, name, short_name, widgets, -1
            return
        end

        if ev.value eq 'Build polarized image' && widgets.status eq 0 then begin
            if imgs_get_len(imgs) eq 0 then begin
                dummy = dialog_message('No image loaded', /error)
                return
            end
            widgets.status = 2
            widgets.pb_skipped = 0
            deg = string(byte(176))  ; implicit Latin-1 encoding???
            widget_control, widgets.info_bar, set_value='pB - select 1st image,
0' + deg
            widget_control, widgets.img_name, set_value='Click SKIP above to
ignore this angle'
            gui_select_widget, widgets.info_icon
            tv, imgs.skip, true=1
            return
        end

        if ev.value eq 'Build polarized image' && widgets.status ge 2 &&
widgets.status le 4 then begin
            deg = string(byte(176))  ; implicit Latin-1 encoding???
            if pb_skip eq 0 then begin
                n = imgs.current_img
            end else begin
                n = -1
                widgets.pb_skipped = widgets.status - 1
                gui_select_widget, widgets.info_icon
                tv, imgs.warning, true=1
            end
            case widgets.status of
                2: begin
                    s = '2nd image, 45' + deg
                    widgets.pb1 = n
                end
                3: begin
                    s = '3rd image, 90' + deg
                    widgets.pb2 = n
                end
                4: begin
                    s = '4th image, 135' + deg
                    widgets.pb3 = n
                end
            end
            widgets.status += 1
            widget_control, widgets.info_bar, set_value='pB - select ' + s
            if pb_skip eq 1 then begin
                widget_control, widgets.img_name, set_value=' '
            end else if widgets.pb_skipped eq 0 then begin
                widget_control, widgets.img_name, set_value='Click SKIP above to
ignore this angle'
            end
            return
        end

        if ev.value eq 'Build polarized image' then begin
            reset_status, widgets
            if widgets.pb_skipped ne 1 then begin
                tmp0 = imgs_get_image(imgs, widgets.pb1)
            end else begin
                tmp0 = 0
            end
            if widgets.pb_skipped ne 2 then begin
                tmp1 = imgs_get_image(imgs, widgets.pb2)
            end else begin
                tmp1 = 0
            end
            if widgets.pb_skipped ne 3 then begin
                tmp2 = imgs_get_image(imgs, widgets.pb3)
            end else begin
                tmp2 = 0
            end
            if pb_skip eq 0 then begin
                tmp3 = imgs_get_current_image(imgs)
            end else begin
                tmp3 = 0
                widgets.pb_skipped = 4
            end
            image = build_pB(tmp0, tmp1, tmp2, tmp3, widgets.pb_skipped)

            name = 'pB image'
            short_name = 'pB image'
            imgs_append, imgs, image, name, short_name, widgets,
imgs_get_exp(imgs, widgets.pb1)
            return
        end

        print, 'WARNING:', ev.value, ' is not yet implemented'  ; FIXME remove
this
    end
end
```

8.3.gui_low_level.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

function get_white_rect, w, h
    return, make_array(w, h, value=255, /byte)
end

function get_y_from_top, y, height
    return, !d.y_size - y - height
end

function get_thumb_size, widgets
    return, widgets.thumb_size
end

pro gui_resize_thumbnails_area, widgets, prev_size, delta
    ; FIXME: prev_size can be calculated from !d.y_size (but first do a
    ;     "gui_select_widget, widgets.thumbs" !!!)
    ystep = widgets.thumb_size + 26
    hstart = 10
    if prev_size + delta eq 0 then begin
        w = 1  ; IDL doesn't like zero sizes
        h = 1
    end else begin
        w = widgets.thumb_size + 10 * 2
        h = ystep * (prev_size + delta) + hstart
    end

    ; Backup the contents of thumbs and restore them after resize because the
    ; IDL GUI libraries from ITT VIS are nothing more than a big pile of
    ; rotting waste and they sometimes corrupt the widget contents during the
    ; resize (e.g. IDL 7.0.6 on Linux when a part of the window is hidden).
    ; Yes, this happens even with "device, retain=2": the documentation sucks
    ; too.
    gui_select_widget, widgets.thumbs
    bak_w = min([!d.x_size, w])
    bak_h = min([!d.y_size, h])
    bak = tvrd(0, !d.y_size - bak_h, bak_w, bak_h, true=1)
    widget_control, widgets.thumbs, draw_xsize=w, draw_ysize=h
    tv, bak, 0, !d.y_size - bak_h, true=1

    if delta gt 0 then begin
        if prev_size eq 0 then begin
            ystep += hstart
        end
        tv, get_white_rect(widgets.thumb_size+10*2, ystep)
    end
    if h eq 1 then begin
        ; try to deuglify the white dot; 192 (0xc0) is the color used for the
        ; background in IDL 7.0 on GNU/Linux on my computer. Yeah, I know...
        erase, 'c0c0c0'x
    end
end

pro gui_clean_all, widgets
    gui_select_widget, widgets.main_image
    erase
    gui_select_widget, widgets.profile_x
    erase
    gui_select_widget, widgets.profile_y
    erase
    widget_control, widgets.img_name, set_value=' '
    for i = 0, widgets.camera_info->get_len() - 1 do begin
        ; use a single space as value, not an empty string
        widget_control, widgets.camera_info->get_item(i), set_value=' '
    end
end

pro display_thumbnail, widgets, image, seqnum, n, selected, name
    gui_select_widget, widgets.thumbs
    thumb_size = widgets.thumb_size  ; FIXME: fix this if thumbs are not square
anymore!
    ystep = thumb_size + 26
    y = n * ystep
    x = 10
    close_x = x + thumb_size - 14
    if selected then begin
        tv, widgets.selected, 2, get_y_from_top(y + 2, 132), true=1
    end else begin
        tv, widgets.not_selected, 2, get_y_from_top(y + 2, 132)
    end
    tvscl, image, x, get_y_from_top(y + 26, thumb_size)
    ofs_num = 2
    tv, seqnum, x + ofs_num, get_y_from_top(y + 10, 16)
    tv, widgets.close_icon, close_x, get_y_from_top(y + 10, 14), true=1

    xt = x + ofs_num + 16  ; FIXME: 16 is the width of seqnum
    yt = y + 10
    wt = close_x - xt
    ht = 16
    ; add a bit of spacing:
    xt += 4
    wt -= 6
    ;tv, get_white_rect(wt, ht) * 0.75, xt, get_y_from_top(yt, ht)  ; FIXME
remove this
    clip = [xt, get_y_from_top(yt + ht - 1, 1), xt + wt - 1, get_y_from_top(yt,
1)]
    idl_quirk_baseline_offset = 4
    ; , color='0000ff'x  ; FIXME remove this
    xyouts, xt, get_y_from_top(yt - 4, ht), name, /device, clip=clip, noclip=0,
color=0
end

pro delete_thumbnail, widgets, n
```

```
    gui_select_widget, widgets.thumbs
    thumb_size = widgets.thumb_size
    ystep = thumb_size + 26
    y = n * ystep
    tv, widgets.not_selected, 2, get_y_from_top(y + 2, 132)
end

pro display_image, widgets, image, camerainfo, name
    gui_select_widget, widgets.main_image
    tvscl, image
    widget_control, widgets.img_name, set_value=name

    for i = 0, widgets.camera_info->get_len() - 1 do begin
        s = camerainfo[i]
        if s eq '' then begin
            ; with an empty string the widget will disappear changing the
            ; vertical space it uses, which seems undesiderable, so use a
            ; single space
            s = ' '
        end
        widget_control, widgets.camera_info->get_item(i), set_value=s
    end
end

pro draw_profile, widgets, im, xp, yp
    x = xp
    y = yp

    s = size(im)
    w = s[1]
    h = s[2]

    disp_size = widgets.disp_size
    if x eq -1 && y eq -1 then begin
        x = w / 2
        y = h / 2
    end else if x lt 0 || x ge disp_size || y lt 0 || y ge disp_size then begin
        return
    end else begin
        x = x * (w - 1) / (disp_size - 1)
        y = y * (h - 1) / (disp_size - 1)
    end

    gui_select_widget, widgets.profile_x
    ;plot,image(*, 256),xtitle='X
pixel',ytitle='Counts',thick=1.8,color=black,background=white
    plot, im[*, y], xtitle='bin X', ytitle='DN', color=0, background='ffffff'x
    plots, [x, x], [!y.crange(0), !y.crange(1)], line=0, /data, color='0000ff'x
; red

    gui_select_widget, widgets.profile_y
    plot, im[x, *], xtitle='bin Y', ytitle='DN', color=0, background='ffffff'x
    plots, [y, y], [!y.crange(0), !y.crange(1)], line=0, /data, color='0000ff'x
; red
end

; why IDL doesn't provide this in its standard library?
pro gui_select_widget, widget
    widget_control, widget, get_value=id
    wset, id
end
```

8.4.list.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

; IDL_List version 0.3
;
; Variable-sized dynamic array that can contain values of any type and can have
; size zero, heavily ispired by the Pyhon list class.
;
; Each time you put a value in a list a copy of the value is created and only
; the copy itself is stored in the list (the only exception are "object"
; variables, i.e. class instances, since IDL never makes copies of objects).
; FIXME: maybe support the "temporary" function to avoid copies of arguments
;     passed to apppend and set_item, by changing set_item to something like:
;         copy = value
;         (*self.items)[i] = ptr_new(copy, /no_copy)
; FIXME: document that copies of lists are never created, e.g.:
;         a = list()
;         b = a
;     simply creates two references (aliases) to the same list. This is
;     guaranteed as part of the list public interface.
;
; IMPORTANT: basic rules of thumb for memory management in this implementation:
;     0) never use heap_free
;     1) ptr_new() without arguments in list__define doesn't affect memory
;        allocation and can be ignored;
;     2) there's a ptr_new (in ::init) each time a list is created and a
;        corresponding ptr_free (in ::cleanup); ::cleanup also has a ptr_free
;        for any remaining item in the list
;     3) ::_resize is roughly equivalent to creating a new list and deleting
;        the old one, so it has both ptr_new and ptr_free
;     4) each time an item is added to the list there's a ptr_new, and each
;        time an item is removed there's a ptr_free
;     5) ::set_item is equivalent to a removal followed by an insertion, so it
;        has both ptr_new and ptr_free
;     6) never use heap_free

pro list__define
    dummy = {list, len: 0, items: ptr_new()}
end

function list::init
    ; yes, this double-pointer madness is necessary to make this work in IDL...
    self.items = ptr_new(ptrarr(1))
    return, 1  ; success
end

pro list::cleanup
    ; Both the following statements are required: the first frees any remaining
    ; items in the list (including null pointers and dangling pointers in the
    ; overallocated area at the end of *self.items, but apparently IDL doesn't
    ; mind), the second frees *self.items itself (the ptrarr).
    ptr_free, *self.items
    ptr_free, self.items
end

function list::get_len
    return, self.len
end
```

```
function list::_decode_index, index
    if index lt -self.len or index ge self.len then begin
        ; FIXME: this should raise an exception or something (maybe simply call
        ;     exit?)
        ; FIXME: make the error message more user friendly!
        print, 'IndexError: list index out of range', index, self.len, $
            n_elements(self.items)
    end
    i = index  ; don't change the original value
    if i lt 0 then begin
        i += self.len
    end
    return, i
end

pro list::_resize, delta_len
    ; this gives constant-time amortized speed for append, algorithm stolen
    ; from the CPython list implementation (but their code is bug-free)
    new_len = self.len + delta_len
    allocated = n_elements(*self.items)
    if allocated lt new_len or new_len lt allocated / 2 then begin
        ; mild over-allocation, not exactly identical to CPython since we start
        ; from 1, not from 0, and this code is less sophisticated
        ; FIXME: this formula doesn't do well when doing del_item and len < 10
        new_allocated = new_len + new_len / 8 + 4
        if new_len eq 0 then begin
            new_allocated = 1  ; IDL sucks
        end
        new_items = ptrarr(new_allocated)
        min_len = min([new_len, self.len])
        ;print, min_len, n_elements(*self.items), n_elements(new_items)

        if min_len ne 0 then begin
            ; FIXME: the next line is probably wildly inefficient, maybe use 2
            ;     codepaths for shrink and enlarge???
            new_items[0:min_len - 1] = (*self.items)[0:min_len - 1]
        end

        ; WARNING: without the following ptr_free IDL leaks memory, but at the
        ;     same time this shouldn't be a heap_free because otherwise IDL
        ;     will free all the objects (e.g. another list) that are contained
        ;     *within* this list!
        ptr_free, self.items

        self.items = ptr_new(new_items, /no_copy)
    end
    self.len = new_len
end

pro list::_debug_dump
    print, 'allocated:', n_elements(*self.items)
    print, 'len:', self.len
    for i = 0, self.len - 1 do begin
        print, i, *(*self.items)[i]
    end
end

function list::get_item, index
    return, *(*self.items)[self->_decode_index(index)]
end

pro list::set_item, index, value
    i = self->_decode_index(index)
    ptr_free, (*self.items)[i]
    (*self.items)[i] = ptr_new(value)
end

pro list::del_item, index
    i = self->_decode_index(index)
    ptr_free, (*self.items)[i]
    if i lt self.len - 1 then begin
        (*self.items)[i:self.len - 2] = (*self.items)[i + 1:self.len - 1]
    end
    self->_resize, -1
end

pro list::append, value
    self->_resize, 1
    ; don't use "self->set_item, -1, value", since set_item assumes that the
    ; previous value is valid and must be deallocated and this may introduce
    ; very nasty bugs in rare situations (e.g. if "_resize, -1" leaves in
    ; items[self.len] a copy of items[self.len - 1])
    (*self.items)[self.len - 1] = ptr_new(value)
end

; FIXME: move the following comment at the top of the module:
; this must be called by the user, failing to do so results in a memory leak
pro list_destroy, self
    obj_destroy, self
end

function list
    return, obj_new('list')
end
```

8.5.decode_raw_header.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

; return the size of the array returned by decode_raw_header
function get_raw_header_info_size
    return, 6
end

function get_empty_img_info, exp
    img_info = {camerainfo: strarr(6), seqnum: 0, exp: exp}
    return, img_info
end

function decode_raw_header, filename
    ; NAME:
    ;   decode_raw_header
    ;
    ; PURPOSE:
    ;   Load the raw images of the UVCI visible-light detector (VLD)

    ; CALLING SEQUENCE:
    ;   decode_raw_header(filename)
    ;
    ; INPUTS:
```

```
    ;   header:  header of the image
    ;   image:  The image to be loaded

    ; OUTPUTS:
    ;   40-byte long vector
    ;
    ; COMMON BLOCKS:
    ;   None.
    ;
    ; SIDE EFFECTS:
    ;   None.
    ;
    ; RESTRICTIONS:
    ;   None.
    ;
    ; PROCEDURE:
    ;   Straightforward.
    ;
    ; EXAMPLE:
    ;   camerainfo = decode_raw_header('file.img')
    ;
    ; MODIFICATION HISTORY:
    ;   vers. 0.  S. Fineschi, 22 November, 2008.
    ;   vers. 1.  M. Pancrazzi, S. Fineschi, 2 March, 2009
    ;   vers. 2.  Lino Mastrodomenico, 3 June, 2009

    camerainfo = strarr(6)  ; human-readable camera information

    fi = file_info(filename)
    size = fi.size
    if size ne 551046 then begin
        camerainfo[0] = 'Camera: UVD'
        img_info = {camerainfo: camerainfo, seqnum: 0, exp: -1}
        return, img_info
    end

    openr, lun, filename, /get_lun
    hdr=bytarr(40); store the first 40-byte vector of raw image
    ;             Image name  = 32 byte
    ;             VLD Header = 8 byte
    readu, lun, hdr
    free_lun, lun  ; close, lun  ; FIXME: which is the correct one???

    ; Decode camera information stored in the header
    filename=string(hdr[0:31])
    camerainfo_B=hdr[32:37]; camera information in Bytes

    ; 1st info: camera ID
    if camerainfo_B[0] ge 128 then camerainfo[0]='Camera: UVD'else
camerainfo[0]='Camera: VLD'

    ; 2nd info: aquisition mode
    aqmode = camerainfo_B[0]
    aqmode = aqmode and 112
    aqmode = aqmode / 16
    case aqmode of
        3: begin
            camerainfo[1] = 'Exp: 10 s'
            exp = 10
        end
        4: begin
            camerainfo[1] = 'Exp: 20 s'
            exp = 20
        end
        5: begin
            camerainfo[1] = 'Exp: 10 s (dark)'
            exp = 10
        end
        6: begin
            camerainfo[1] = 'Exp: 20 s (dark)'
            exp = 20
        end
    end

    ; 3rd info: No. of images after "start" comand
    nima=camerainfo_B[0] and 15
    if nima lt 10 then begin  ; FIXME!!!
        camerainfo[2]='Image no.: '+string(nima, format='(I1)')
    end else begin
        camerainfo[2]='Image no.: '+string(nima, format='(I2)')
    end

    ; 4th info: LCVR Temperature
    msb=camerainfo_B[1]; most significant Byte
    lsb=camerainfo_B[2]; least significant Byte
    if msb gt 128 then begin
        msb=msb and 127
        msb=-1*msb
    end
    temp=(float(msb)*256.+float(lsb))/10.

    deg = string(byte(176))  ; implicit Latin-1 encoding???
    camerainfo[4] = 'LCVR temp.: ' + string(temp, format='(F5.1)') + ' ' + deg +
'C'

    ; 5th info: pB sequence no.
    seqnum = camerainfo_B[3]
    camerainfo[3] = 'pB seq. no.: ' + string(seqnum, format='(I1)')

    ; 6th info: CCD temperature
    msb=camerainfo_B[4]; most significant Byte
    lsb=camerainfo_B[5]; least significant Byte
    if msb gt 128 then begin
        msb=msb and 127
        msb=-1*msb
    end
    temp=(float(msb)*256.+float(lsb))/10.
    camerainfo[5] = 'CCD temp.: ' + string(temp, format='(F5.1)') + ' ' + deg +
'C'

    ;print,'Image name: ', string(hdr[0:31])
    ;print,'Camera header: ', camerainfo[0:5]
    img_info = {camerainfo: camerainfo, seqnum: seqnum, exp: exp}
    return, img_info
end
```

8.6.get_selected.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

function get_selected
```

```
    im = bytarr(3, 116, 132)
    im[0:1, *, *] = 64
    im[2, *, *] = 255
    im[*, 6:109, 6:125] = 255
    return, im
end
```

8.7.image_utils.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

; this file contains misc image and data bookkeeping utils

function imgs_create_thumbnail, imgs, image_orig, img_info, thumb_size
    image = congrid(image_orig, thumb_size, thumb_size)  ; FIXME: keep aspect
ratio
    seqnum = imgs.digits->get_item(img_info.seqnum)
    thumbnail = {image: image, seqnum: seqnum}
    return, thumbnail
end

pro imgs_display_thumbnail, imgs, n, selected, widgets
    img_data = imgs.data->get_item(n)
    display_thumbnail, widgets, img_data.thumbnail.image, $
        img_data.thumbnail.seqnum, n, selected, img_data.short_name
end

function fix_image_for_display, image_orig, disp_size
    ; "normalize" the image for display:
    s = size(image_orig)
    w = s[1]  ; FIXME: maybe swap w and h???
    h = s[2]
    if w ne disp_size or h ne disp_size then begin
        ; FIXME: maybe should keep the aspect ratio?
        image_b = congrid(image_orig, disp_size, disp_size)
        ; make a copy to avoid changing the original image:
    end else begin
        image_b = image_orig
    end

    ; The following two lines are necessary otherwise histogram() below freaks
    ; out with "Array has too many elements" for images with high dynamic range
    ; (e.g. after photometric calibration). Did I mention that IDL sucks for
    ; scientific calculations?
    image_b -= min(image_b)
    image_b *= 1000.0 / max(image_b)
    nans = where(finite(image_b, /nan))
    if (size(nans))[0] ne 0 then begin
        ; in IDL 7.0 histogram() below craps out on NaN (yes, even with "/nan")
        image_b[nans] = 0
    end
    hist = histogram(image_b[16:disp_size - 16, 16:disp_size - 16], omin=min_v,
omax=max_v)
    cut_black = (disp_size - 32) ^ 2.0 / 1000
    cut_white = (disp_size - 32) ^ 2.0 * 999 / 1000
    start_v = min_v
    end_v = max_v
    ; FIXME: this looks slow
    c = 0
    for j = 0l, n_elements(hist) - 1 do begin
        c += hist[j]
        if c ge cut_black then begin
            start_v = min_v + j
            break
        end
    end
    c = 0
    for j = 0l, n_elements(hist) - 1 do begin
        c += hist[j]
        if c ge cut_white then begin
            end_v = min_v + j
            break
        end
    end
    image_b[where(image_b le start_v)] = start_v
    image_b[where(image_b ge end_v)] = end_v

    return, image_b
end

function imgs_get_len, imgs
    return, imgs.data->get_len()
end

function imgs_image_exists, imgs
    return, imgs.data->get_len() ne 0
end

function imgs_get_image, imgs, index
    return, (imgs.data->get_item(index)).image
end

function imgs_get_current_image, imgs
    return, (imgs.data->get_item(imgs.current_img)).image
end

function imgs_get_current_name, imgs
    return, (imgs.data->get_item(imgs.current_img)).name
end

function imgs_get_current_short_name, imgs
    return, (imgs.data->get_item(imgs.current_img)).short_name
end

function imgs_get_exp, imgs, index
    return, (imgs.data->get_item(index)).img_info.exp
end

function imgs_get_current_exp, imgs
    return, (imgs.data->get_item(imgs.current_img)).img_info.exp
end

pro imgs_select_image, imgs, n, widgets
    if imgs.current_img ne -1 then begin
        imgs_display_thumbnail, imgs, imgs.current_img, 0, widgets
    end
    imgs_display_thumbnail, imgs, n, 1, widgets
    img_data = imgs.data->get_item(n)
    display_image, widgets, img_data.disp_image, img_data.img_info.camerainfo,
img_data.name
```

```
    imgs.current_img = n
    draw_profile, widgets, imgs_get_current_image(imgs), -1, -1
end

pro imgs_delete_image, imgs, n, widgets
    ; FIXME: this can be much faster, but pay attention to all corner cases if
    ;     you change it!
    num_images = imgs.data->get_len() - 1
    delete_thumbnail, widgets, num_images
    imgs.data->del_item, n
    redraw = 0
    if n eq imgs.current_img then begin
        redraw = 1
        if n eq num_images then begin
            imgs.current_img -= 1
        end
    end else if n lt imgs.current_img then begin
        imgs.current_img -= 1
    end
    for i = n, num_images - 1 do begin
        if i ne imgs.current_img then begin
            imgs_display_thumbnail, imgs, i, 0, widgets
        end
    end
    if num_images eq 0 then begin
        gui_clean_all, widgets
    end else if redraw then begin
        imgs_select_image, imgs, imgs.current_img, widgets
    end else begin
        ; this is necessary here because redrawing a nearby thumbnail may
        ; overwrite a part of the highlight for the current image
        imgs_display_thumbnail, imgs, imgs.current_img, 1, widgets
    end
    ; remove space for one more thumbnail:
    gui_resize_thumbnails_area, widgets, imgs.data->get_len() + 1, -1
end

pro imgs_append, imgs, image, name, short_name, widgets, exp, img_info
    if n_elements(img_info) eq 0 then begin
        img_info = get_empty_img_info(exp)
    end
    disp_image = fix_image_for_display(image, widgets.disp_size)
    thumb_size = get_thumb_size(widgets)
    thumbnail = imgs_create_thumbnail(imgs, disp_image, img_info, thumb_size)
    img_data = {image: image, disp_image: disp_image, thumbnail: thumbnail, $
                img_info: img_info, name: name, short_name: short_name}
    imgs.data->append, img_data
    ; add space for one more thumbnail:
    gui_resize_thumbnails_area, widgets, imgs.data->get_len() - 1, 1
    imgs_select_image, imgs, imgs.data->get_len() - 1, widgets
    ; scroll to the end of the thumbnails, if necessary (FIXME: should be in
gui_low_level.pro)
    widget_control, widgets.thumbs, set_draw_view=[0, 0]
end
```

8.8.iras_run.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
```

```
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

; FIXME: this path should really be configured in the preferences
!path = expand_path('+/home/lino/osservatorio/idl/solarsoft/all') + ':' + !path

; WARNING: the order of the following statements is important: import
;     first the dependencies and then the code that uses them!
.run read_conf
.run list
.run get_selected
.run get_close_small
.run get_skip
.run get_t0
.run get_t1
.run get_t2
.run get_t3
.run get_t4
.run get_warning
.run resources
.run decode_raw_header
.run misc_utils
.run gui_low_level
.run raw_images
.run vignetting
.run image_utils
.run iras_gui

iras_gui
```

8.9.misc_utils.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

; Moore-Penrose pseudoinverse of a matrix
function pinv, m
    ; Why in the hell IDL doesn't provide this out-of-the-box?!?!
    ; FIXME: check that this is correct
    return, invert(transpose(m) ## m) ## transpose(m)
end

; build polarized brightness image
function build_pB, a, b, c, d, image_to_skip
    if image_to_skip eq 0 then begin
        print, 'pB with 4 images'  ; FIXME: delete this line
        ; value of a^(-1) = p
        ; first row, intensity (currently unused)
        p11 = 0.267627
        p12 = 0.246785
        p13 = 0.233534
        p14 = 0.252055
        ; second row, q
        p21 = 0.516075
        p22 = -0.0235569
        p23 = -0.480393
        p24 = -0.0121246
        ; third row, u
        p31 = 0.0636487
        p32 = 0.495361
        p33 = -0.0472306
        p34 = -0.511779

        q = a * p21 + b * p22 + c * p23 + d * p24
        u = a * p31 + b * p32 + c * p33 + d * p34
    end else begin
        print, 'pB with 3 images, skip ' + string((image_to_skip - 1) * 45,
format='(I3)') + ' degrees'  ; FIXME: delete this line
        angles = [-0.8, -50.2, -89.8, -136.2]
        for i = 0, 3 do begin
            angles[i] *= !dtor
        end
        case image_to_skip of  ; FIXME: there has to be a better way to do this
            1: begin
                angles = [angles[1], angles[2], angles[3]]
                im0 = b
                im1 = c
                im2 = d
            end
            2: begin
                angles = [angles[0], angles[2], angles[3]]
                im0 = a
                im1 = c
                im2 = d
            end
            3: begin
                angles = [angles[0], angles[1], angles[3]]
                im0 = a
                im1 = b
                im2 = d
            end
            4: begin
                angles = [angles[0], angles[1], angles[2]]
                im0 = a
                im1 = b
                im2 = c
            end
        end
        p = [[1, cos(2 * angles[0]), -sin(2 * angles[0])], $
             [1, cos(2 * angles[1]), -sin(2 * angles[1])], $
             [1, cos(2 * angles[2]), -sin(2 * angles[2])]] / 2
        ;print, invert(p)
        ;p = pinv(p)
        ;print, p
        p = invert(p)
        q = im0 * p[0, 1] + im1 * p[1, 1] + im2 * p[2, 1]
        u = im0 * p[0, 2] + im1 * p[1, 2] + im2 * p[2, 2]
    end
    pB = sqrt(u ^ 2 + q ^ 2)
    return, pB
end

function sc_inverse, n, diag, below, above
    IF (above EQ 0) THEN above=1.0d-200

    wt_above=double(above)/diag
    wt_below=double(below)/diag
    wt_above_1=wt_above-1
    wt_below_1=wt_below-1

    power_above=dblarr(n-1)
    power_below=dblarr(n-1)

    power_above(0)=1
    power_below(0)=1

    for row=1,n-2 do begin
        power_above(row)=power_above(row-1)*wt_above_1
        power_below(row)=power_below(row-1)*wt_below_1
    end

    v= [ 0 , wt_below *(power_below*reverse(power_above))]
    u= [ 0 , wt_above *(power_above*reverse(power_below))]

    d = -u(1)/wt_above - (total(v)-v(N-1))

    f=1/(d+wt_above*total(v))

    u[0]=d
    v[0]=d
    u=u*f
    v=reverse(v)*f
    p=dblarr(n,n,/nozero)
    p[0,0]=u
    for row=1,n-2 do begin
        p[0,row]=v[n-row-1:n-2]
        p[row,row]=u[0:n-row-1]
    end
    p(0,n-1)=v

    return, p
end

; shutterless image correction
pro shutless2, imagei, exptime, imageo
    image = float(imagei)
    sz = size(image)
    cleartime = 0.272248
    exp_eff = exptime + cleartime
    line_ro = 1262e-6
    line_clr = 124e-6
    fixup = sc_inverse(sz(2), exp_eff, line_ro, line_clr)
    imageo = (fixup##image)
end
```

8.10.raw_images.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

function raw_read_image, filename
    fi = file_info(filename)
    size = fi.size
    if size eq 9245031 then begin
        image = read_binary(filename, data_dims=[2150, 2150], data_start=31, $
                            data_type=12, endian='big')
    end
    if size eq 551046 then begin
        image = read_binary(filename, data_dims=[536, 514], data_start=36, $
                            data_type=12, endian='little')
        image = transpose(65535 - image)
    end
    ;if size eq 32 + 1088 * 1024 * 2 then begin  ; IDL sucks really hard
    if size eq 2228256 then begin
        image = read_binary(filename, data_dims=[1088, 1024], data_start=32, $
                            data_type=12, endian='big')
    end
    return, image
end

function get_vld_header, filename
    fi = file_info(filename)
    size = fi.size
    if size eq 551046 then begin
        openr, lun, filename, /get_lun
        header = bytarr(40)
        readu, lun, header
        free_lun, lun
        return, header[35]  ; pB sequence number
    end
    return, 0
end
```

8.11.read_conf.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

pro read_default_conf, widgets
    widgets.uv_exp = 40
    widgets.photometric_VL = 1.8e+11
    widgets.photometric_H = 5e+9
    widgets.photometric_He = 7.9e+8
    widgets.xc = 230
    widgets.yc = 268
end

pro read_conf, filename, widgets
    ; FIXME: a better way to do this?
    line = bytarr(1000)
    line[*] = 64
    line = string(line)

    openr, lun, filename, /get_lun
    for i = 1, file_lines(filename) do begin  ; FIXME: test with missing last
newline
        readf, lun, line
        n = strpos(line, '#')
        if n ne -1 then begin
            line = strmid(line, 0, n)
        end
        line = strtrim(line)
        if line eq '' then begin
            continue
        end
        tmp = strsplit(line, '=', /extract)
        name = strtrim(tmp[0])
        value = float(tmp[1])
        ;help, name, value
        case name of
            'uv_exp': widgets.uv_exp = value
            'photometric_VL': widgets.photometric_VL = value
            'photometric_H': widgets.photometric_H = value
            'photometric_He': widgets.photometric_He = value
            'xc': widgets.xc = value
            'yc': widgets.yc = value
        end
    end
    free_lun, lun
end
```

8.12.resources.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

function get_image, name
    ; FIXME: is this ok?
    ;return, read_png('data/' + name + '.png')
    case name of
        'close_small': return, get_close_small()
        'selected': return, get_selected()
        'skip': return, get_skip()
        't0': return, get_t0()
        't1': return, get_t1()
        't2': return, get_t2()
        't3': return, get_t3()
        't4': return, get_t4()
        'warning': return, get_warning()
    end
end
```

8.13.vignetting.pro

```
; IRAS - Image Reduction and Analysis Software
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

function remove_vignetting, image, radius, xc, yc
    ; FIXME: these are preliminary values
    xy = [[0.00, 0.000000], [33.56, 0.000000], [67.76, 0.000000], $
          [101.63, 0.000000], [135.51, 0.000000], [169.39, 0.000000], $
          [203.27, 0.000000], [237.15, 0.000000], [271.02, 0.009464], $
          [304.90, 0.037855], [338.78, 0.078864], [372.66, 0.126183], $
          [406.53, 0.157729], [440.41, 0.211356], [474.29, 0.255521], $
          [508.17, 0.268139], [542.05, 0.293375], [575.92, 0.305994], $
          [609.80, 0.299685], [643.68, 0.287066], [677.56, 0.280757]]
    x = xy[0, *]
    y = xy[1, *]
    m = shift(dist(radius * 2), xc, yc)
    m = cspline(x, y, m)
    ;m[where(m lt 0.001)] = 1
    m[where(m lt 0.05)] = 1
    if 1 then begin  ; FIXME: this is an ugly hack!
        s = size(image)
        w = s[1]
        h = s[2]
        m = congrid(m, w, h)
    end
    ;print, min(m), max(m)
    tmp = image / m
    ;print, min(tmp), max(tmp), min(image), max(image)
    return, tmp
    ;return, image / m < 65535
end

function remove_vignetting2, image, radius, xc0, yc0
    ; FIXME: these are completely uncalibrated values
    xy = [[0, 1.000000], [1, 1.000000], [2, 1.000000], [3, 1.000000], $
          [4, 1.000000], [5, 1.000000], [6, 1.000000], [7, 1.000000], $
          [8, 1.000000], [9, 1.000000], [10, 1.000000], [11, 1.000000], $
          [12, 1.000000], [13, 1.000000], [14, 1.000000], [15, 1.000000], $
          [16, 1.000000], [17, 1.000000], [18, 1.000000], [19, 1.000000], $
          [20, 1.000000], [21, 1.000000], [22, 1.000000], [23, 1.000000], $
          [24, 1.000000], [25, 1.000000], [26, 1.000000], [27, 1.000000], $
          [28, 1.000000], [29, 1.000000], [30, 1.000000], [31, 1.000000], $
          [32, 1.000000], [33, 1.000000], [34, 1.000000], [35, 1.000000], $
          [36, 1.000000], [37, 1.000000], [38, 1.000000], [39, 1.000000], $
          [40, 1.000000], [41, 1.000000], [42, 1.000000], [43, 1.000000], $
          [44, 1.000000], [45, 1.000000], [46, 1.000000], [47, 1.000000], $
          [48, 1.000000], [49, 1.000000], [50, 1.000000], [51, 1.000000], $
          [52, 1.000000], [53, 1.000000], [54, 1.000000], [55, 1.000000], $
          [56, 1.000000], [57, 1.000000], [58, 1.000000], [59, 1.000000], $
          [60, 1.000000], [61, 1.000000], [62, 1.000000], [63, 1.000000], $
```

```
[64, 1.000000], [65, 1.000000], [66, 1.000000], [67, 1.000000], $
[68, 1.000000], [69, 1.000000], [70, 1.000000], [71, 1.000000], $
[72, 1.000000], [73, 1.000000], [74, 1.000000], [75, 1.000000], $
[76, 1.000000], [77, 1.000000], [78, 1.000000], [79, 1.000000], $
[80, 1.000000], [81, 1.000000], [82, 1.000000], [83, 1.000000], $
[84, 1.000000], [85, 1.000000], [86, 1.000000], [87, 1.000000], $
[88, 1.000000], [89, 1.000000], [90, 1.000000], [91, 1.000000], $
[92, 1.000000], [93, 1.000000], [94, 1.000000], [95, 1.000000], $
[96, 1.000000], [97, 1.000000], [98, 1.000000], [99, 1.000000], $
[100, 1.000000], [101, 1.000000], [102, 1.000000], [103, 1.000000], $
[104, 1.000000], [105, 1.000000], [106, 1.000000], [107, 1.000000], $
[108, 1.000000], [109, 1.000000], [110, 1.000000], [111, 1.000000], $
[112, 1.000000], [113, 1.000000], [114, 1.000000], [115, 1.000000], $
[116, 1.000000], [117, 1.000000], [118, 1.000000], [119, 1.000000], $
[120, 1.000000], [121, 1.000000], [122, 1.000000], [123, 1.000000], $
[124, 1.000000], [125, 1.000000], [126, 1.000000], [127, 1.000000], $
[128, 1.000000], [129, 1.000000], [130, 1.000000], [131, 1.000000], $
[132, 1.000000], [133, 1.000000], [134, 1.000000], [135, 1.000000], $
[136, 1.000000], [137, 1.000000], [138, 1.000000], [139, 1.000000], $
[140, 1.000000], [141, 1.000000], [142, 1.000000], [143, 1.000000], $
[144, 1.000000], [145, 1.000000], [146, 1.000000], [147, 1.000000], $
[148, 1.000000], [149, 1.000000], [150, 0.342597], [151, 0.353709], $
[152, 0.365820], [153, 0.378520], [154, 0.391920], [155, 0.405826], $
[156, 0.420085], [157, 0.434681], [158, 0.449451], [159, 0.464168], $
[160, 0.478558], [161, 0.492490], [162, 0.505688], [163, 0.518300], $
[164, 0.530263], [165, 0.541579], [166, 0.552645], [167, 0.563147], $
[168, 0.573724], [169, 0.584297], [170, 0.594649], [171, 0.605165], $
[172, 0.615041], [173, 0.624697], [174, 0.633930], [175, 0.642939], $
[176, 0.652494], [177, 0.662343], [178, 0.672957], [179, 0.683868], $
[180, 0.694709], [181, 0.705647], [182, 0.716449], [183, 0.727140], $
[184, 0.737658], [185, 0.747519], [186, 0.756526], [187, 0.765000], $
[188, 0.773006], [189, 0.780675], [190, 0.788400], [191, 0.795634], $
[192, 0.802358], [193, 0.808910], [194, 0.814820], [195, 0.820784], $
[196, 0.826807], [197, 0.832911], [198, 0.839726], [199, 0.846730], $
[200, 0.854282], [201, 0.861847], [202, 0.868996], [203, 0.875922], $
[204, 0.881870], [205, 0.887342], [206, 0.892068], [207, 0.895986], $
[208, 0.899772], [209, 0.903576], [210, 0.908062], [211, 0.913424], $
[212, 0.919184], [213, 0.924900], [214, 0.930176], [215, 0.935057], $
[216, 0.939872], [217, 0.944629], [218, 0.949373], [219, 0.953566], $
[220, 0.957125], [221, 0.960531], [222, 0.963905], [223, 0.967838], $
[224, 0.972160], [225, 0.975924], [226, 0.978881], [227, 0.980847], $
[228, 0.982331], [229, 0.984229], [230, 0.986860], [231, 0.990174], $
[232, 0.993251], [233, 0.995727], [234, 0.997265], [235, 0.997558], $
[236, 0.997668], [237, 0.997449], [238, 0.997269], [239, 0.997501], $
[240, 0.997218], [241, 0.996840], [242, 0.995999], [243, 0.994890], $
[244, 0.993796], [245, 0.992458], [246, 0.991099], [247, 0.989301], $
[248, 0.987542], [249, 0.985698], [250, 0.983797], [251, 0.982091], $
[252, 0.980000], [253, 0.977994], [254, 0.975844], [255, 0.973523], $
[256, 0.971188], [257, 0.968382], [258, 0.965341], [259, 0.961988], $
[260, 0.958652], [261, 0.955441], [262, 0.952416], [263, 0.949461], $
[264, 0.946219], [265, 0.942653], [266, 0.938350], [267, 0.933581], $
[268, 0.928679], [269, 0.924078], [270, 0.920200], [271, 0.916937], $
[272, 0.913857], [273, 0.910431], [274, 0.906783], [275, 0.902611], $
[276, 0.898346], [277, 0.894301], [278, 0.889596], [279, 0.884967], $
[280, 0.880359], [281, 0.876125], [282, 0.873628], [283, 0.872226], $
[284, 0.872060], [285, 0.872120], [286, 0.871468], [287, 0.870573], $
[288, 0.868930], [289, 0.867804], [290, 0.867314], [291, 0.867391], $
[292, 0.868236], [293, 0.868430], [294, 0.868724], [295, 0.867776], $
[296, 0.865953], [297, 0.864265], [298, 0.861422], [299, 0.859547], $
[300, 0.857871], [301, 0.856334], [302, 0.854927], [303, 0.852768], $
[304, 0.850305], [305, 0.846992], [306, 0.844138], [307, 0.841118], $
[308, 0.838077], [309, 0.835010], [310, 0.830819], [311, 0.826577], $
[312, 0.821828], [313, 0.817304], [314, 0.813853], [315, 0.810573], $
[316, 0.807687], [317, 0.804508], [318, 0.800475], [319, 0.795899], $
[320, 0.790792], [321, 0.785473], [322, 0.780083], [323, 0.774516], $
[324, 0.768978], [325, 0.763437], [326, 0.757880], [327, 0.752700], $
[328, 0.747696], [329, 0.742666], [330, 0.738123], [331, 0.733716], $
[332, 0.729646], [333, 0.725887], [334, 0.721517], [335, 0.716676], $
[336, 0.711145], [337, 0.705552], [338, 0.700529], [339, 0.696322], $
[340, 0.693311], [341, 0.691116], [342, 0.689086], [343, 0.686765], $
```

```
          [344, 0.682986], [345, 0.677820], [346, 0.671943], [347, 0.665469], $
          [348, 0.659790], [349, 0.654591], [350, 0.649727], [351, 0.645708], $
          [352, 0.642190], [353, 0.639599], [354, 0.637885], [355, 0.636110], $
          [356, 0.633849], [357, 0.630800], [358, 0.626816], [359, 0.622625], $
          [360, 0.618719], [361, 0.615173], [362, 0.612178], [363, 0.609577], $
          [364, 0.607379], [365, 0.605230], [366, 0.603236], [367, 0.600965], $
          [368, 0.597741], [369, 0.594299], [370, 0.590232], [371, 0.586175], $
          [372, 0.582312], [373, 0.605346], [374, 0.655810], [375, 0.733639], $
          [376, 0.839545], [377, 0.919429], [378, 0.972983], [379, 1.000000], $
          [380, 1.000000], [381, 1.000000], [382, 1.000000], [383, 1.000000], $
          [384, 1.000000], [385, 1.000000], [386, 1.000000], [387, 1.000000], $
          [388, 1.000000], [389, 1.000000]]
    ;xc = 230
    ;yc = 536 - 268
    xc = xc0
    yc = yc0

    x = xy[0, *]
    y = xy[1, *]
    s = size(image)
    w = s[1]
    h = s[2]
    xc *= w / 514
    yc *= h / 536
    m = shift(dist(w * 2, h * 2), xc, yc)
    if 0 then begin
        dark = image[xc - 50:xc + 50, yc - 50:yc + 50]
        ndark = mean(dark)
        print, ndark
        tmp = image - ndark + 6122.89
        print, mean(tmp[xc - 50:xc + 50, yc - 50:yc + 50])
    end else begin
        tmp = image
    end
    m = m[0:w - 1, 0:h - 1]
    m = cspline(x, y, m)
    tmp /= m
    ;tmp[xc - 50:xc + 50, yc - 50:yc + 50] = 0
    return, tmp
end
```

8.14.test_leaks.pro

```
; run with:
;     idl -e test_leaks

function get_mem
    return, (memory())[1]-(memory())[2]
end

pro test_leaks
    ; FIXME: see also: ~/osservatorio/iras/memory_leaks.txt
    list_destroy, list()
    start = get_mem()
    list_destroy, list()
    for i = 0, 1 do begin
        tmp = list()
        list_destroy, tmp
        objs = list()
        n = objs->get_len()
        objs->append, i * 2.531e6
        objs->append, 0
        objs->append, [7, 8, 9]
        objs->set_item, 2, [4, 5, 6]

        print, get_mem() - start
        objs->append, [1, 2, 3]
        objs->del_item, 2
        ;objs->_debug_dump
        objs->append, list()
        (objs->get_item(-1))->append, list()
        (objs->get_item(-1))->append, ['abc', 'def', '123']
        list_destroy, (objs->get_item(-1))->get_item(0)
        list_destroy, objs->get_item(-1)
        objs->del_item, -1
        ;objs->_debug_dump
        print, get_mem() - start

        objs->append, 'abc' + 'def'
        list_destroy, objs
        print, get_mem() - start
    end
    ; the following list should print '<NullPointer>'
    print, ptr_valid()
end
```

8.15.test_list.pro

```
; this does a number of basic tests on list, it doesn't test for memory leaks
; run with:
;     idl -e test_list

pro assert_equal, a, b, s
    if a ne b then begin
        print, 'Error:', s
    end
end

pro test_list
    objs = list()
    ; use 10 items to be sure to trigger a list::_resize
    for i = 0, 9 do begin
        objs->append, i + 123
    end
    for i = 0, 9 do begin
        assert_equal, objs->get_item(i), i + 123, '1'
    end
    for i = 0, 9 do begin
        assert_equal, objs->get_item(i - 10), i + 123, '2'
    end
    for i = 1, 9 do begin
        assert_equal, objs->get_item(1), i + 123, '3'
        objs->del_item, 1
    end
    assert_equal, objs->get_item(0), 123, '4'
    objs->del_item, -1
    assert_equal, objs->get_len(), 0, '5'

    objs->append, 1
    objs->append, 2
    objs->append, 3
    objs->append, 4
    objs->del_item, 0
    objs->append, 5
    assert_equal, objs->get_item(-1), 5, '6'

    tmp = list()
    objs->append, tmp
    list_destroy, objs
    assert_equal, tmp->get_len(), 0, '6'
end
```

8.16.iras.sh

```
#!/bin/bash

exec idl -e '@iras_run'
```

8.17.build.py

```
#!/usr/bin/python

# FIXME: this shouldn't be hardcoded!
program_name = '''\
; IRAS - Image Reduction and Analysis Software
'''

program_version = '''\
; Version %s
'''

copyright_and_stuff = '''\
;
; Copyright (C) 2007, 2008, 2009  Osservatorio Astronomico di Torino
; Copyright (C) 2009  Lino Mastrodomenico
; Copyright (C) 2009  Maurizio Pancrazzi
; Authors: Silvio Giordano, Lino Mastrodomenico, Silvano Fineschi,
;     Maurizio Pancrazzi
;
; This program is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; This program is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with this program.  If not, see <http://www.gnu.org/licenses/>.

'''

default_blurb = program_name + copyright_and_stuff
tmp_prefix = "version = '"
for line in open('iras_gui.pro', 'rb'):
    line = line.strip()
    if line.startswith(tmp_prefix):  # FIXME: use a regex for all this stuff
        line = line[len(tmp_prefix):]
        version = line[:line.index("'")]
        break
print repr(version)

prefix = '.run '
f = open('iras.pro', 'wb')
f.write(program_name + program_version % version + copyright_and_stuff)

for line in open('iras_run.pro', 'rb'):
    if line.startswith(prefix):
        line = line[len(prefix):].strip()
        s = open(line + '.pro', 'rb').read()
        # FIXME: the removal of the blurb maybe should be more flexible?
        assert s.startswith(default_blurb), line
        s = s[len(default_blurb):]
        f.write('; file %s.pro\n\n' % line)
        f.write(s)
        f.write('\n')

f.write('''\
pro iras
    iras_gui
end
''')
f.close()
```

8.18.make_release.sh

```
#!/bin/bash

# WARNING: call ./build.py immediately before a new release!
# usage: ./make_release.sh VERSION_NUMBER

iras_dir='iras'  # FIXME: shouldn't be hardcoded if it's the current dir!

if [ -e 'iras.pro' -a ! -e "$iras_dir/iras.pro" ]; then
    cd ..
fi

if [ ! -e "$iras_dir/iras.pro" ]; then
    echo 'Error: cannot find IRAS directory'
    exit 1
fi

relname="iras-$1"
if [ -e "$relname" ]; then
    echo "Error: directory $relname exists"
    exit 1
fi

if [ -e "$relname.zip" ]; then
    echo "Error: file $relname.zip exists"
    exit 1
fi

cp -a "$iras_dir" "$relname"
cd "$relname"

# FIXME: remove this when iras_run.pro is fixed
sed -i 's/\(^!path = .\+$\)/;\1/' iras.pro

for i in *~ *.fits; do
    rm "$i"
done

if [ -e 'todo2.txt' ]; then
    rm 'todo2.txt'
fi

for i in *; do
    if [ -h "$i" ]; then
        rm "$i"  # delete symbolic links
    fi
done

cd ..
zip -9r "$relname.zip" "$relname"
echo "$relname.zip"
```

8.19.check_style.sh

```
#!/bin/bash

for i in *.pro
do
    ./fix_idl_indentation.py "$i" | diff -u "$i" -
    chk_txt.py "$i" | grep -v 'line longer than 79 characters'
done

for i in *.sh *.py
do
    chk_txt.py "$i"
done
```

8.20.fix_idl_indentation.py

```
#!/usr/bin/python

# fix_idl_indentation.py version 0.3
# this is a bit fragile but it's often good enough

import fileinput
import re
import sys
from string import ascii_letters, digits


# WARNING: it's very important to *ALWAYS* check for hostile code, unless you
#     receive the code from someone you completely and absolutely trust. This
#     danger is *NOT* purely theoretical!

# the following commands can be used to obfuscate hostile code:
bad_commands = ['call_function', 'call_method', 'call_procedure', 'execute']

# the following regex catches obfuscated code like:
#     string("142b) + string("165b) + string("147b)
bad_regex = re.compile(r'string\s*\(\s*"')

# you should never use the following commands, unless you know exactly what you
# are doing:
dangerous_commands = ['spawn', 'call_external']

indent_start = ['function', 'pro', 'case']
indent_end = ['begin']
dedent_start = ['end', 'endif', 'endcase', 'endfor', 'endelse', 'endwhile',
                'endrep', 'endswitch']  # IDL sucks!
common_start = 'common'
valid_name_chars = '_' + ascii_letters + digits
indent_size = 4


def warning(s):
    sys.stderr.write('WARNING: ' + s + '\n')


NORMAL_MODE, OPEN_BRACKET_MODE, IN_COMMAND_MODE = range(3)

indent = extra_indent = next_extra_indent = 0
mode = NORMAL_MODE
for line in fileinput.input():
    assert '\t' not in line  # do an expand first!
    orig_line = line.strip()
    line = orig_line.lower()

    for bad_command in bad_commands:
        if bad_command in line:
            warning('this line may contain malicious code: %r' % orig_line)
    if bad_regex.search(line) is not None:
        warning('this line may contain obfuscated code: %r' % orig_line)
    for dangerous_command in dangerous_commands:
        if dangerous_command in line:
            warning('this line may contain dangerous code: %r' % orig_line)

    if ';' in line:
        line = orig_line[:orig_line.index(';')].strip()
    current_indent = indent
    if mode == NORMAL_MODE:
        if line.count('(') > line.count(')'):
            mode = OPEN_BRACKET_MODE
            next_extra_indent = line.index('(') + 1
        elif line.count('[') > line.count(']'):
            mode = OPEN_BRACKET_MODE
            next_extra_indent = line.index('[') + 1
        elif line.count('{') > line.count('}'):
            mode = OPEN_BRACKET_MODE
            next_extra_indent = line.index('{') + 1
        elif (',' in line and line.endswith('$') and
              (all(c in valid_name_chars for c in line[:line.index(',')]) or
               line.startswith(common_start + ' '))):
            mode = IN_COMMAND_MODE
            next_extra_indent = indent_size
        else:
            for s in dedent_start:
                if line == s or line.startswith(s + ' '):
                    indent -= indent_size
                    current_indent = indent
            for s in indent_start:
                if line.startswith(s + ' '):
                    indent += indent_size
            for s in indent_end:
                if line.endswith(' ' + s):
                    indent += indent_size
    elif mode == OPEN_BRACKET_MODE:
        if (line.count('(') < line.count(')') or
            line.count('[') < line.count(']') or
            line.count('{') < line.count('}')):
            mode = NORMAL_MODE
            next_extra_indent = 0
    elif mode == IN_COMMAND_MODE:
        if not line.endswith('$'):
            mode = NORMAL_MODE
            next_extra_indent = 0
    if not orig_line:
        print
    else:
        print ' ' * (current_indent + extra_indent) + orig_line
    extra_indent = next_extra_indent
```

8.21.img2idl.py

```
#!/usr/bin/python

import sys

import Image


for filename in sys.argv[1:]:  # FIXME do proper arg parsing
    assert '.' in filename
    prefix = filename[:filename.rfind('.')]  # FIXME: do something smart
    out = 'get_' + prefix + '.pro'
    assert out != filename
    f = open(out, 'wb')
    f.write('function get_' + prefix + '\n')
    im = Image.open(filename)
    f.write('    return, byte([')
    w, h = im.size
    for y in range(h - 1, -1, -1):
        if y < h - 1:
            f.write('                 ')  # there should be one more space here
        f.write('[')
        for x in range(w):
            a = str(im.getpixel((x, y)))
            b = ',' if x < w - 1 else ''
            f.write(a.replace(' ', '').replace('(', '[').replace(')', ']') + b)
        # IDL 6.3 has problems with very long lines, so split every row
        f.write('],$\n' if y > 0 else ']')
    f.write('])\n')
    f.write('end\n')
    f.close()
```

8.22.img2idl.sh

```
#!/bin/bash

cd data
for filename in close_small.png t[0-4].png warning.png
do
    echo "$filename"
    ../img2idl.py "$filename"
    mv "get_$(basename "$filename" '.png').pro" ..
done
```