Astronomical Data Analysis Software and Systems VIII
ASP Conference Series, Vol. 172, 1999
D. M. Mehringer, R. L. Plante, and D. A. Roberts, eds.

Image Display Paradigm #3

Dyer Lytle

Lunar and Planetary Laboratory, University of Arizona, Tucson, AZ 85721

Elizabeth Stobie, Anthony Ferro, Irene Barg

Steward Observatory, University of Arizona, Tucson, AZ 85721

Abstract. Image Display Paradigm #3 (IDP3) is an IDL (Interactive Data Language) package developed by the NICMOS Software Group at the University of Arizona. Its original purpose was to provide a versatile graphics tool for PSF subtraction with NICMOS data. However, with a clever structured design IDP3 became a unique and very powerful image analysis tool. In this presentation we explore IDP3, its development and its features.

1. Introduction

IDP3 is a tool for manipulating data images that employs a powerful but easy-to-use graphical user interface with support for FITS and HDF input formats. It allows the user to work with a collection of images and display one or more in a graphics window simultaneously. Images may be individually moved, scaled, and rotated to bring features into registration. Each image may be subtracted from, added to, or combined with the composite display using one of several other functions (divide, multiply, XOR, etc). IDP3 provides a region-of-interest pop-up tool for intensive examination of sub-regions of the main display. There are cross-section plots, masks, surface plots, statistics, spreadsheet displays, radial profiles, and other tools at the user's disposal. Combination or individually registered images may be saved to FITS files.

2. Features of IDP3

- The main image display [work area] is resizable, scalable, zoomable, and shiftable displaying multiple images simultaneously.
- IDP3 employs Lytle's general principles of interactivity (see Appendix A).
- Data alignment [shift and rotation] by visual, statistical, and centroidal criteria.
- User definable and adjustable data masks and anti-masks.
- Region of interest display.
- Cross sections and radial profiles.
- Surface plots and surface views from multiple perspectives.

- Zooming and interpolation with conservation of flux.
- Arithmetic operations on selected images including addition, subtraction, division and XOR.
- Image blinking.
- Multiple mouse buttons perform separate functions in multiple windows.
- Documentation, all modifications to data recorded in header of saved image.
- Support for FITS and HDF formats on input, FITS format on output.
- User preferences allow customization.

3. Design and philosophy

The evolution of IDP3 was driven by a need for an interactive tool for subtracting point spread functions (PSFs) from data taken with the space telescope instrument, NICMOS. This evolutionary process had the following steps.

- A prototype tool (psfsubtract) was written and tested in IDL[1].
- A few users tried the tool and made suggestions for improvement.
- At a general meeting at which a large fraction of the potential users were present the prototype tool was presented and demonstrated. Attendees were encouraged to comment on and criticize the prototype and to suggest features they would like to see.
- The notes taken from this meeting and were used to outline a requirements document for the final program. The design for the new program was begun and the data structures that would be needed were specified.
- It was at this time that it was seen that there was a possibility that the software being written could be generalized to handle a broader spectrum of input data and to allow the user more general functionality than the original specification dictated. Thus, it was decided to make a program that would satisfy the PSF group (the needs of the few) but that would also be useful (hopefully) for the general user community (the needs of the many).
- The program was then written, tested, and released to users.
- In the months following the initial release, the program evolved based on user requests for added features. These requests were carefully evaluated and an attempt has been made, whenever possible, to interpret user requests in such a way that when the feature was added to the software, it was added in a sufficiently general way so as to be useful in many situations, not just for the particular problem that particular user was trying to solve at that particular time.

4. Internal structure

Structures vs. COMMON Statements In IDP3, an attempt was made to write the program without the use of common statements. This requirement was satisfied although it proved unnecessary. Common statements

[1]IDL is the trademark of Research Systems, Inc.

cause problems when multiple copies of the same program are run from inside a given instance of IDL. However, because named structures were used, multiple copies of IDP3 should not be run together. To make IDP3 truly self-independent, the named structures should be replaced with anonymous structures, that is, when structures are allocated, they should be inline in the program rather than being referenced from an external file, i.e., idp3_structs. This would make the code look a bit sloppier (in our opinion) but would allow the user to run multiple copies of IDP3 simultaneously from a single IDL session without unwanted interactions between copies.

Structure and Pointer usage The first parts of the program written were the data structures. Pointers were used to allow easier memory management and more elegant structures and information passing.

Root structure, Updates, Coordinating windows With a number of interactive windows on the screen, the event handlers for these windows must coordinate in such a way as to keep the internal data up to date and valid at all times.

Active windows, Automatic updates Some of the windows on the screen display current internal data values. When any of this internal data changes, the displayed values and/or displayed 1D and 2D plots must be automatically updated to reflect these changes.

Flags and Signals The various data structures in the program contain a variety of variables used to pass current values of flags around inside the program. Some of these flags can be thought of as signals to other parts of the program.

Event handling Various event handlers handle events from various windows. Sometimes, events from a particular widget must be turned off to allow another subroutine to turn on and interpret events for that widget. This is a way of overloading events for a particular widget.

5. Conclusions and Future Plans

The time invested in the software engineering of IDP3 was time very well spent. Advancements in IDL Version 5, particularly the support for data pointers, made it possible to develop a very solid but flexible framework for IDP3 that easily facilitates future enhancement.

IDP3 has been a huge success with members of the NICMOS Project and continues to be enhanced according to user requests. It may also be evolved as part of a suite of analysis routines for other NASA missions including the Imaging Science Subsystem of the Cassini Mission to Saturn.

IDP3 is available from the NICMOS website[2] at the University of Arizona.

---

[2]`http://nicmos.as.arizona.edu/software`

6. Appendix A

Lytle's General Principles of Interactivity

I User Help

- Online help will be available.
- Make active button and field help (cursor tracking events) available and selectable if possible.

II General computer display layout considerations

- Multiple windows for flexibility
- Resizable graphics where appropriate
- Limit number of windows so they don't overfill the display

III Active window updates

- If the window layout must change with the update, kill and redraw the window
- If the window layout remains the same with the update, just update the fields in the window and/or replot the 1-D or 2-D data.

IV 'Modes' considered harmful

- When possible, use separate mouse buttons to perform multiple input functions in a window.
- Avoid using 'modes' where mouse buttons perform different functions depending on which 'mode' button has been pressed.

V Pull-down menu buttons vs display screen real estate

- If an application involves image display and lots of function buttons, it is often best to put the function buttons in pull-down menus, including walking menus.

VI Grey out buttons when their use would be inappropriate.

VII Use 'bulletin board' bases for differentiating grouped functions.

VIII Implement 'user preference' files when appropriate.

IX Do not build a GUI that requires changes to the user's .Xdefaults file.

X Catch errors and report them rather than allowing the program to bomb.