IRAF NEWSLETTER

February 1989 Number 6

Central Computer Services National Optical Astronomy Observatories* P. O. Box 26732 Tucson, AZ 85726

Table of Contents

* Operated by the Association of Universities for Research in Astronomy, Inc. (AURA) under contract with the National Science Foundation

System News

System Overview

In early December, IRAF version 2.7 was frozen and released internally on all NOAO/Tucson computers as the local production version of the system. This version of IRAF contained all the latest enhancements to the system at that time including cursor readback capabilities for the Sun workstation and the IIS image displays, the lastest version of CCDRED installed in the IMRED package, the latest version of APPHOT now installed in the DIGIPHOT package, the new ECHELLE package utilizing a new echelle data format (a two dimensional image with each line consisting of an extracted echelle order), and the new tasks IRMATCH1D and IRMATCH2D in the PROTO package for matching intensity as well as position when generating IR mosaics.

The initial Sun386i/IRAF port was completed in early December and a beta release of IRAF version 2.7 for the 386i was made to all Sun386i sites. The performance of this system in its usual configuration with 8 Mb of memory and one or two local 300 Mb SCSI disks was somewhat disappointing due to the memory hungry nature of the 386i, and the relatively low i/o bandwidth of the SCSI disks (200-600 Kb/sec for SCSI on a 386i compared to 1000-2000 Kb/sec for SMD on a Sun-3 or Sun-4). This situation is improved somewhat with IRAF version 2.8 by the introduction of the IRAF shared library facility, but one will still notice the effect of the slow disks, and more than 8 Mb of memory is desirable to minimize paging. To date the 386i version of Sun/IRAF has been sent to 17 sites.

IRAF version 2.7 has also been released on a limited basis to other sites, in particular, sites upgrading to SunOS-4 (Sun/IRAF version 2.7 includes full support for SunOS-4), or sites requesting the new ECHELLE package. IRAF version 2.7 is available only for UNIX and VMS sites and Sun sites running either SunOS-3 or SunOS-4.

Preparations for the next major release of IRAF, version 2.8, are now in progress. This will be the first major release of IRAF since July 1987, when IRAF version 2.5 was released. IRAF version 2.8 will be available for all supported hosts including VAX/UNIX (BSD 4.3), VAX/VMS 4.7 (reported to run under VMS-5 as well), VAX/Ultrix 3.0, SunOS-3 and SunOS-4, AOSVS, Alliant, Convex, HP-UX (HP9000 series only), and the Masscomp. We are considering possible ports to the Apollo, Mac II, DECstation 3100, and SGI, and anyone interested in running IRAF on these or other host systems should contact us to discuss the status of IRAF on the system in question.

The release of IRAF version 2.8 is being coordinated with the first release of the STScI Science Data Analysis System (STSDAS) from the Space Telescope Science Institute. Sites requesting STSDAS through STScI will need to obtain and install IRAF version 2.8 before they can install STSDAS; older versions of IRAF do not provide all of the needed facilities.

System Software Update

A new system interface for photon or event-list data (QPOE) has been completed and is now in testing at the Center for Astrophysics, where it is being used to develop the PROS reduction package for Rosat. Other new system facilities include several interfaces used to implement image masks, and a file manager interface which will serve as the primary data storage facility for the new image structures, and which is currently being used in QPOE. These new interfaces have been integrated into IRAF version 2.8 and will be available with the next IRAF release.

A new shared library facility has been developed for Sun/IRAF (SunOS-4 only). The shared library facility minimizes disk space requirements and reduces runtime memory consumption and paging, speeding process startup and execution. The time required to link an applications program is also reduced, speeding software development. As an example of the savings in disk space, the executables for the Sun-3 version of IRAF before the implementation of the shared libraries, required 16Mb; with shared libraries the same set of binaries requires only 8Mb of disk space, or as little as 6 Mb if the executables are stripped. This implementation of shared libraries is a feature of Sun/IRAF and does not use the SunOS-4 shared library facilities, which

are currently only available for programs written in C.

The support provided by IRAF for external or "layered" packages has been much improved. Layered packages are packages which are external to the core IRAF system but which derive their functionality from it; examples are the NOAO packages (newly restructured as a layered package), STSDAS, PROS, and so on. Each site may now maintain their own version of the LOCAL package external to IRAF as a layered package, making it easier to maintain locally added tasks independently of the core system. Layered packages are completely self contained, with their own BIN and LIB directories, help database, etc., making installation trivial - only a single file has to be modified in the core system to install or deinstall a layered package. The facilities for software development of layered packages are now on a par with those for the core system.

These are only a few of the many enhancements being made to IRAF as preparation of version 2.8 proceeds; we will report on more in the next newsletter.

Applications Software Update

A first version of the new DIGIPHOT package, which will combine APPHOT and the IRAF version of DAOPHOT, is nearing completion and will soon be undergoing user testing. Dennis Crabtree (DAO), and Lindsey Davis (NOAO/IRAF) are collaborating on the new package. The new software will not be included in IRAF version 2.8 but plans are to make it available as an add-on (layered package) later this year.

Work continues on a number of other large projects, most of which have been reported before. Lindsey Davis is working on a general image registration and mosaicing package, and Mike Fitzpatrick is working on the radial velocity analysis package (see accompanying article in this newsletter). Dyer Lytle is working on the DECOMP package, to be used for spectral line deconvolution, fourier filtering, etc., of high resolution and high signal to noise spectra. Steve Rooke is developing software for volume visualization (see accompanying article in this newsletter); the current version of this software is available upon request from Steve. Frank Valdes, currently on leave at CTIO, has contributed the new task SPECPLOT and some new ECHELLE and multispec reduction tasks, as noted elsewhere in this newsletter.

An Acknowledgment

As a final note, many of you stopped by the IRAF display at the AAS meeting in Boston in early January. We found our conversations with all of you very profitable, and welcomed our once a year chance to meet the IRAF user community. We would like express our special thanks to Susan Kleinmann of the University of Massachusetts for loaning us her Sun386i for this demo.

Doug Tody

IRAF at the USC Psychology Department

One of the long term goals of the research done in Richard F. Thompson’s laboratory, a core component of the USC Neurosciences program, is to understand the processes of learning and memory by analyzing how the brain codes, stores and retrieves information. Our methods include empirical neurobiological studies including electrophysiological recording from microelectrodes placed deep within the brain to monitor the activity of individual neurons and groups of neurons in the awake behaving animal; neuroanatomical tract tracing involving the use of chemicals to map the internal circuitry of the brain; receptor binding studies to identify changes in cell surface molecules that are believed to be involved in memory storage; and

carbon-14 2-deoxyglucose studies to monitor which regions of the brain are active during performance of any 30 to 45 minute task.

As part of this research images are acquired from serial sections of frozen brains cut on a sliding microtome at 40 micron intervals. These sections are usually stained with a dye that marks the location of the cells in each section. The primary purpose for examining these sections is to identify and document the location of microelectrode tips which were previously used to record signals from deep within the brain. Alternatively, these same sections are often used to identify the path followed by dyes injected into the intact brain a few days before the sections are cut. As it turns out, in the intact brain these dyes are transported either up or down the entire length of a neuron to provide evidence for which regions of the brain are connected to which other regions of the brain.

Images from these sections are currently captured with a CIRCON MV-9015-H CCD camera. The video images are digitized (512 x 384 pixels) with a DATACUBE IVG-128 video acquisition and display board on an IBM AT computer system. The images are transferred over ethernet to our SUN3/160C which is running IRAF. Background artifacts are then removed with the IMREPLACE task. The images are displayed on the Sun with the DISPLAY task and then stored as Sun rasterfiles using the IMTOOL/SAVE option in the IMTOOL/SETUP mode. The images are stored in a public domain rasterfile editing program (touchup) for manual removal of any remaining background artifacts. The rasterfiles are then converted back into IRAF image files using the task IRAFIL.

There are about 30 people involved in this research. About 6 of our people are actively involved in using image processing systems including the Drexel-Dumas Quantitative Autoradiography System, the US Army Ballistic Research Laboratory CAD Package, MOVIE.BYU, Utah Raster Toolkit, and the NOAO/IRAF system. We are continually looking for systems that will allow us to enhance our ability to perform increasingly sophisticated image processing tasks as our own knowledge of this area increases. We are particularly interested in working with larger images (1024x1024 to 4096x4096 pixels), 3D reconstructions, and developing FFT superresolution techniques.

The NOAO/IRAF package has proven to be remarkably easy to learn and use. The large number of general purpose image processing routines, data analysis, plotting, and display subsystems provide the most comprehensive environment for image analysis, optical system transfer function modeling, and data management we have encountered to date. Indeed, we have used the NOAO/IRAF system as a standard in evaluating the sources of errors introduced into our data by each of the other image processing packages available to our laboratory. In addition, the valuable telephone hot line support provided by the NOAO/IRAF consulting team is far superior in both quality and timeliness of response to any of the other systems investigated.

This research is funded by NSF BNS-8106648, ONR N0001473K0238, and McKnight Foundation grants to Richard F. Thompson. Funds are also provided by the USC Neural, Informational & Behavioral Sciences (NIBS) program.

AJ Annala
USC, Psychology Department

Unix/IRAF Image File Protection (the `. .´ File)

The proper way to delete an IRAF image is with the IMDELETE task. Deleting IRAF images with host level facilities can result in only partial deletion of the images unless one fully understands how IRAF images are represented at the host level. This article discusses the pitfalls associated with host level deletion of IRAF images, and how to recover from such attempts.

An IRAF image (standard or OIF format) has two files associated with it, the image header file (extension .imh) and the pixel file (extension .pix, often in a different directory from the header file). To prevent users from inadvertently deleting only the image header file with the IRAF DELETE task, which would leave a headerless pixel file behind somewhere on disk, the image header file is a "protected" file. The IMDELETE task will unprotect the header, then delete both the header and pixel files.

On any IRAF system, deleting the header file will leave a headerless or "zombie" pixel file behind. Usually such files will be discovered eventually and deleted, by file expiration if nothing else. A potentially more insidious problem arises because of the file protection mechanism used to protect the image header file in Unix/IRAF.

The implementation of the IRAF file protection mechanism differs depending upon the host operating system. On Unix/IRAF systems, since there is no Unix level file delete protection, a hard link is used to flag the header file as protected. This link is just a separate directory entry for the header file and does not take up additional disk space, other than a few bytes in a directory file. The link has the same name as the image header file, with the two characters `. .´ prefixed. Since the filename begins with a `.´ it does not show up by default in Unix or IRAF directory listings; the `ls -a` command in Unix or `dir all+` in IRAF will show these *hidden* files.

If you use the Unix `rm` command to delete an IRAF image header file, not only will the pixel file remain, but the hard link to the image header will remain as well. When the header is removed from the directory, the file still exists with the hidden link as the sole remaining directory entry, and as long as any links remain, a Unix file is not deleted. For example, if m31.imh is removed with the `rm` command, the header file will still exist as ..m31.imh. Each IRAF image that is deleted with `rm` instead of the IMDELETE task leaves behind a `. .´ file, and although the header file appears to disappear in the default directory listing, the file is not actually deleted.

To clean up these zombie `. .´ files, a simple Unix `find` command can be issued, as shown below. This has been tested at NOAO on a VAX under BSD 4.3 and on Sun3 and Sun4 machines under SunOS 4.0, and should work on most Unix/IRAF hosts without modification.

To delete any zombie header files in the current directory and recursively in all subdirectories, with a prompt for each file, type:

```
% find . -name '..*.imh' -links 1 -exec /bin/rm -i {} \;
```

To delete the files silently, type:

```
% find . -name '..*.imh' -links 1 -exec /bin/rm {} \;
```

To make a list of the orphan zombies, without deleting them, type:

```
% find . -name '..*.imh' -links 1 -print
```

If one insists on using host level facilities to delete IRAF images (as is sometimes desirable for efficiency reasons when deleting very large numbers of images), this whole problem can be avoided by storing a related group of images in a dedicated directory on a scratch disk, using "reset imdir=HDR$pix/" to arrange for the pixel files to be created in a subdirectory of the header directory. This causes the image headers, header protect links, and pixel files to all be

placed in the same directory tree, allowing use of "rm -r" on the root directory to delete everything.

[Many users have had problems with the `. .´ files; we will try to do away with these hidden links in a future version of IRAF. -ed].

Rob Seaman
Suzanne Jacoby

Calling IMFORT Procedures from C Programs

The IMFORT interface is a library of subroutines used to create and access IRAF images, intended for use in user written host Fortran or C programs. Many IRAF sites have used the IMFORT library to write host level programs that create IRAF images from local format data or that access existing IRAF images. The IMFORT interface is documented in Volume 1A of the IRAF manual set.

Although the IMFORT interface is written in such a way that it can be called from programs written in any language, e.g., Fortran or C, only a Fortran *binding* (set of calling sequences) is provided. This article discusses how to call the IMFORT procedures (or any Fortran subroutines) from C programs. The methods for calling Fortran procedures from C are highly OS dependent; all we can do here is provide examples for some of the more popular host systems. This information is intended only as a stopgap measure, until we can provide C language bindings of the IMFORT interface for the most popular host IRAF systems. These bindings will hide the details of mixed language programming from the IMFORT C programmer, making IMFORT as easy to call from C as from Fortran.

C differs from Fortran in the way the external (procedure) names are formed, and in how string and scalar arguments appear in argument lists. C passes scalar arguments by *value*, whereas Fortran passes scalar arguments by *reference* (i.e., by the address of the referenced variable). Integer and floating array arguments are treated the same in both languages. The biggest difference is in how character strings are handled.

In Berkeley UNIX, SunOS, and probably on most systems derived from Berkeley UNIX, Fortran external names are generated by converting the Fortran name to lower case and adding leading and trailing underscores. C merely prepends an underscore, hence the C name for the Fortran procedure IMOPEN is "imopen_". The f77 compiler represents strings as NULL delimited character arrays, like C, but in argument lists an additional argument is added (after all the other arguments) to pass the allocated length of the string.

Consider the following Fortran code:

```
character*80 image
integer im, ier
call imopen (image, 1, im, ier)
```

The following would accomplish the same thing in C on a BSD derived system:

```
char    image[80];
int     im, ier, len_image=79, mode=1;
imopen_(image, &mode, &im, &ier, &len_image);
```

VMS C is similar except that the trailing underscore is omitted from the procedure name (in VMS, C and Fortran externals occupy the same name space), and strings are passed by a single argument, the pointer to a VMS *string descriptor*. The equivalent C code for VMS is as follows:

```
#include <descrip.h>
char     image[80];
struct   dsc$descriptor_s s;
int      im, ier, mode=1;

s.dsc$a_pointer = image;
s.dsc$w_length  = 79;
s.dsc$b_dtype   = DSC$K_DTYPE_T;
s.dsc$b_class   = DSC$K_CLASS_S;

imopen (&s, &mode, &im, &ier);
```

Fortunately, few character string arguments are used in IMFORT; most procedures require only simple integer and real scalar and vector arguments.

It is not difficult to see how to go about constructing a C language binding from these examples. Anyone needing to call IMFORT from C should first contact us to see if such a binding is already available for their system.

Suzanne Jacoby
Doug Tody

HEDIT Operations

Many of you are familiar with the task IMAGES.HEDIT for editing image headers. However, you probably only use it to explicitly add, delete, or change the value of a keyword in one image or possibly you are aware that a list of images can be edited in one fell swoop. HEDIT can do some more sophisticated things for you provided you learn the special syntax of the task and some tricks to overcome certain limitations. A number of cases of this type have come up and we will use them for examples. With these models you should be able to modify them for other similar problems.

There are many IRAF tasks which require the exposure time given in seconds; for example in CCDRED and ONEDSPEC. Some images from CFHT contain exposure times encoded as sexagesimal numbers (hours-minutes-seconds format). The keywords also contain hyphens which adds an extra complexity because hyphens may also be interpreted as subtraction. To add the exposure time in seconds under the keyword EXPTIME:

```
cl> hedit *.imh exptime 0. add+ ver- show-
cl> hedit *.imh exptime "(@'in-time')" ver- show-
cl> hedit *.imh exptime "(exptime*3600.)" ver- show-
```

The first line adds the new keyword with the real value 0. The second line converts the value of the string parameter IN-TIME to a real value and stores it in EXPTIME. The last line converts the exposure time from hours to seconds.

There are two points to note here. First, the @'in-time' is a special syntax used to reference the value of this keyword rather than the difference of the keywords IN and TIME. To see the second point one has to understand that the more concise command

```
cl> # The following DOES NOT WORK!
cl> hedit *.imh exptime "(@'in-time'*3600.)" add+
```

will not work currently. This is because of a datatype mismatch; i.e. HEDIT tells you that you

are attempting an arithmetic operation on a string parameter. There is no coercion operator to turn the numeric string IN-TIME into a number even though sexagesimal numbers are understood by IRAF. Thus, the first two steps are simply to create a real valued keyword and apply a trick which changes the string valued keyword to a numeric keyword.

Another similar example is encountered with ESO data. Again to use CCDRED one needs the exposure time as well as a header translation file. The image header contains the starting and ending times of the exposure; in this case they are in decimal hours. Again we have to deal with hyphenated keywords.

```
cl> hedit *.imh exptime \
>>> "(mod (@'tm-end'-@'tm-start'+24., 24.)*3600.)" add+ \
>>> ver- show-
```

The modulus operator covers the transition around midnight.

The final example occurs with using the UT time (again as a sexagesimal string) for arc reference spectrum interpolation using REFSPECTRA (IRAF V2.7 and later). The arcs are at the beginning and end of long object exposures. The interpolation weights come out wrong because the UT in the header is at the beginning of the exposure and so one arc appears much closer in time than the other. One solution is to compute the midtime of the exposure from UT and EXPTIME.

```
cl> hedit *.imh midut 0. add+ ver- show-
cl> hedit *.imh midut "(ut)" ver- show-
cl> hedit *.imh midut "(mod (midut+exptime/7200.,24.))" \
>>> ver- show-
```

The idea is basically the same as the first example with the modulus operator from the second example to cover the transition at midnight.

Frank Valdes

APSCATTER: A New Task for Subtracting Scattered Light

A new task, APSCATTER, has been added to the APEXTRACT and ECHELLE packages. It determines and subtracts scattered light from two dimensional aperture or echelle spectra. The scattered light outside the apertures defining the two dimensional spectra is found, smoothed, and subtracted from each input image. The approach is to first select the pixels outside the defined apertures and outside a buffer distance from the edge of any aperture at each point along the dispersion independently. A one dimensional function is fit using the ICFIT package. This fitting uses an iterative algorithm to further reject high values and thus fits the minima between the spectra. (This even works reasonably well if no apertures are defined).

Because each fit is done independently the scattered light thus determined will not be smooth along the dispersion. If desired each line along the dispersion in the scattered light surface may then be smoothed by again fitting a one dimensional function using the ICFIT package. The final scattered light surface is then subtracted from the input image to form the output image. The scattered light surface may also be output if desired.

The reason for using two one dimensional fits as opposed to a surface fit is that the actual shape of the scattered light is often not easily modeled by a simple two dimensional function. Also the one dimensional function fitting offers more flexibility in defining functions and options as provided by the ICFIT package. Experiments with surface fitting did not produce as satisfactory

results as the method adopted. There is no attempt to correct for the wings of the spectra which fall outside the apertures and which may blend together. This is a much more complicated problem.

Frank Valdes

Support for MULTISPEC Format

Many new spectroscopic systems produce multiple spectra on a 2D detector. Last year a new format was introduced for echelle spectra in which all the extracted orders are stored in one two dimensional image rather then as many separate one dimensional images. The APEXTRACT package was modified to produce this format and many of the ONEDSPEC package tasks were modified to allow access to this new format more or less transparently. Also, new tasks were added specifically for echelle spectra; in particular ECIDENTIFY and ECDISPCOR to handle the wavelength calibration.

The same basic approach applies to multiple spectra in which the individual spectra are of different objects rather than different orders of the same object. When the echelle output was added to APEXTRACT a multispec format was also included. The physical format is the same and they differ in having a different APFORMAT identifier which other tasks can use to determine the type of format. This was not advertised since other tasks were not yet prepared to use this format.

Recently the ONEDSPEC tasks were modified to support this new multispec format, and a new task MSDISPCOR was written to provide dispersion correction for multispec data. Dispersion correction is a cross between echelle and individual one dimensional spectra. IDENTIFY, REIDENTIFY, and MSREIDENTIFY (a new task described below) are used to define the dispersion functions for each spectrum. MSDISPCOR applies these solutions, matching them up appropriately, and provides several output formats. Typically one would produce a linearized multispec format where each spectrum could have the same or different wavelength coordinates.

In addition the new tasks MSSELECT and ECSELECT have been added to allow selection and extraction of a subset of the apertures or orders in the multispec and echelle formats. These tasks allow selection by the user defined aperture numbers and properly propagate the aperture wavelength information which would be lost when using image sections.

Except for ECSELECT the new tasks are included in a new IMRED package called MSRED. This package contains most of the tools needed to reduce multifiber and multiaperture spectra in a more efficient manner than previously available.

As a schematic example of how the new package and format is used consider the ARGUS instrument at CTIO. It is a fiber positioner type instrument in which pairs of fibers are set to observe an object and nearby sky. The spectra are fed through the spectrograph to produce a number of alternating parallel spectra of sky and object, all with about the same wavelength coverage.

One begins by defining apertures using the APEXTRACT package. This need be done only once for a reference image such as a flat field. Spectra are then extracted for all the observations using the reference apertures with possible recentering into the multispec format. There will be one output image for each input image.

For an arc IDENTIFY is used to define the dispersion function for the central spectrum. Since the spectra have nearly the same dispersion REIDENTIFY will quickly determine the dispersion functions for the rest of the apertures. A new task MSREIDENTIFY reidentifies lines and computes new dispersion functions from one multispec arc to others, doing this aperture by aperture

rather than with tracing as is done with REIDENTIFY. REFSPEC is used to associate multispec arcs with multispec object observations. MSDISPCOR is then used to linearize the object spectra (usually specifying that the final spectra will all have the same linear wavelength coordinates).

MSSELECT extracts the alternate sky and object spectra yielding two images per original image, each with half the original number of spectra. Because of the alternating format sky subtraction is then a simple matter of image subtraction with IMARITH. Similarly flat fielding is accomplished by dividing with IMARITH. Display is aided by the new SPECPLOT task which also understands onedspec, multispec, echelle, and longslit spectra.

This example is schematic in that one must correct for the fiber throughput and possibly combine skys and exclude sky fibers with accidental objects in them. A procedure script is being written that will do all this with one command and small amount of user interaction. However, the example shows how straightforward it is to reduce ARGUS data. Since ARGUS will eventually produce 48 spectra at a time it is a great benefit to have the multispec format.

Frank Valdes

Radial Velocity Analysis Package Under Development

Work is currently in progress on a new package for IRAF to do radial velocity analysis of one and two dimensional spectra. Two methods are available for one-dimensional velocity computation: a Fourier correlation method following the work of Tonry & Davis (1979, Astron. J, **84**, 1511) in which the user may filter the data prior to correlation, and a squared difference method in which the minimum squared difference of the spectra at trial shifts is computed. The resulting correlation function from either task is fit by a user selected function and a heliocentric velocity computed. Dispersion information may be obtained by previously running the RVDISP task to create a database record containing coefficients for a curve describing the relation between correlation function width and dispersion velocity. Multiple template spectra are supported and a statistical summary of the results for each object spectrum computed. Input spectra may be either true one-dimensional images, echelle format images (where each order is correlated with the corresponding order in the template image), or the new multispec format images.

Redshift analysis of galaxy spectra is also available using either the now standard Fourier Quotient method or a revised Fourier Difference method in which the difference of the FFT's is minimized. This latter method has the benefit of more robust error calculation. A task is available to create equal flux bins from two-dimensional longslit spectra. Each bin in the galaxy spectrum is fit against the template spectrum, producing redshift and dispersion information at various points along the slit and hence the galaxy. A hardcopy of the galaxy spectrum may later be made with absorption lines marked according to their known wavelength plus the computed redshift to check the results. For a less detailed analysis of galaxy spectra, a script task is also included to call the cross correlation tasks to compute a velocity for each bin in the galaxy spectrum.

Other tasks in the package include tasks for sorting output (similar to the APSELECT task in the APPHOT package), image header keyword translation, filter and process parameter editing, and a proposed task for fitting line profiles to trace velocity changes is being considered. The ASTUTIL.RVCORRECT task and IMRED.OBSERVATORY database will be included for completeness.

Suggestions for new tasks or improvements to existing tasks are welcomed. For more information on the Radial Velocity Analysis Package or questions regarding its structure or availability for testing, please contact the author (fitz@noao.edu or 5355::fitz, 602/325-9387).

Mike Fitzpatrick

New Spectrum Plotting Task

A new IRAF task, SPECPLOT, has been written for plotting multiple spectra with provisions for scaling them, separating them vertically, shifting them horizontally, and labeling them. This task is useful for the compression of many spectra to a page for review, intercomparison of spectra, classifications against standards, and final display. It is solely a plotting task and complements the task SPLOT which is primarily an analysis tool that does not plot multiple spectra easily or provide labeling and format options.

Some features of this task are:

- an automatic layout algorithm to make a nice stacked display
- control of the order and separation step between stacked spectra
- ability to move spectra in wavelength or intensity using the cursor or by explicit values
- placement of labels of different types associated with the spectra
- specifyable title and axis labels
- both marker and line plotting

This task resides in the ONEDSPEC package and will be included in the next release of IRAF, IRAF version 2.8.

Frank Valdes

Volume Rendering Update, Video Recording

The prototype PVOL task, mentioned in Newsletter No. 5, has been improved. It now supports 6 different types of transmission/absorption projections, and runs faster. The execution time of about 11 minutes for rendering 36 projections of a 50*50*50 datacube on a diskless Sun-3/60 has been reduced to under 4 minutes. Following are PVOL cpu times for different projection types on a diskless Sun-3/60 and a Sun-4/280, with 36 rotations per run for a 50*50*50 type short integer datacube:

	diskless Sun3/60	Sun4/280
ptype	minutes cpu	cpu
1 opacity only	3:20	:46
2 average intensity	4:32	1:12
3 sum of intensities	3:50	:58
4 inverse distance power	10:51	2:29
5 mod(voxval,param) [thins out cube]	13:39	2:56
6 last intensity-clipped voxel only	3:41	:59

Volume projections can be rather sensitive to the set of projection (transmission, absorption) parameters chosen. One would typically make just a single projection, change a parameter, try another value, and so on. If the datacube is large, one would first cycle through this parameter hunting process on a subsampled image, then run the full volume rotation in background. Like storage requirements for datacubes, PVOL runtimes increase approximately as the volume of the datacube, given enough processor memory. A 450*450*450 datacube took 17:20 hours of cpu on a Sun-4 to do 36 rotation steps with depth cueing (ptype=4); the same run would take about 5:20 hours cpu with an opacity projection (ptype=1).

Various tasks are in testing for manipulating volume images, like 3d transpose, binary conversion to an iraf image, stack, slice, etc. An IRAF script and local host-level task have been set up for recording video frames automatically on a Panasonic videodisc recorder at NOAO. Readers who have a single frame video recording capability that can accept RS-232 commands should contact Steve Rooke if they are interested in doing this themselves. Scripts can be set up to record frames from a datacube (raw or PVOL output) or from a list of ordinary 2d images, or to place images from different sources into different color channels prior to recording.

A 20 minute video demonstrating cross correlation image registration and volume rendering was produced for the IRAF demo at the AAS meeting in Boston (January 1989) using this software. A filmloop capability will eventually be available on most IRAF image displays. In the meantime, the I2SUN task can be used to produce Sun rasterfiles for playback with the host-level MOVIE task found on many Suns.

Steve Rooke

FITS Standards Efforts and IRAF

The IRAF project is committed to the use of FITS for the archival storage and transport of astronomical data processed by IRAF. In the past most of our needs could be met using FITS to store simple image matrices, but our data structures are rapidly getting more complex and there is much discussion within astronomy currently on how to extend FITS to support new types of data, e.g., floating point data, large sets of spectra, event counting data, tabular data, and the complex auxiliary data structures often associated with such datasets. Anyone with unusual requirements involving future use of FITS within IRAF, or anyone interested in or concerned about current FITS standardization activities, is encouraged to contact us with your comments.

Doug Tody

First FOCAS Users Meeting

A first-ever meeting for users of FOCAS (Faint Object Classification and Analysis System) is being planned for Wednesday, June 21, 1989 in Berkeley, CA. This date coincides with the first day of the Berkeley ASP meeting (the Centennial meeting of the ASP, June 21-23). The FOCAS meeting will be held during the afternoon at the ASP conference center.

It is intended that this first meeting will take on a workshop atmosphere, with the purpose of discussing the current status of FOCAS, and charting possible future directions for the package. With this in mind, it is hoped that various attendees will informally present their own scientific

applications of FOCAS, including testing and comparison with other photometry packages, and site-specific enhancements to the software. Frank discussion of shortcomings and desired improvements to FOCAS will be encouraged.

Possible issues to be addressed could include:

- New photometric algorithms
- Image classification techniques
- Multiple objects, image splitting, crowded fields, confusion-limited data
- Detection improvements
- Sun/IRAF compatibility
- Catalog transportability, standardization
- Matched catalogue issues
- Documentation
- Agreement on a standard version of FOCAS

It is hoped that there will be plenty of time for ample and lively discussion concerning broader issues in the field of faint object photometry. To obtain future updates on this meeting, or to make comments and suggestions concerning particulars of the meeting, contact Steven Majewski, Yerkes Observatory (srm@delilah.yerkes.uchicago.edu). Since a special room needs to be reserved at the conference center, notification of intention to attend is appreciated.

Steven Majewski
Yerkes Observatory

IRAF Copyright and Licensing Agreement

With increasingly frequent inquiries from the profit-making sector of the technological world concerning use and possible commercial exploitation of IRAF, we have begun to understand why the other national laboratories have been obliged to implement copyrights and license agreements for their major software packages. With some regret, we have concluded that we must do likewise in order to avoid possible legal complications in the future. Paradoxically, this increased protection is required to guarantee continued wide availability of IRAF.

Therefore with IRAF version 2.8 we plan to initiate the copyright process. Subject to AURA and NSF approval, a licensing agreement will probably accompany a later release. The licensing agreement will be mainly a restatement of matters agreed upon informally in the past.

We specifically want to assure the IRAF community that we will continue to distribute source code and offer consultation and support as necessary to allow each site to make effective use of the software, and to modify it as needed.

Steve Ridgway
Manager, CCS

IRAF Version 2.8 in Preparation for Release

We are preparing IRAF version 2.8 for general release to the user community. Although an exact date has not been set for this release, preparation of V2.8 is currently in full tilt and our plans are to get the system out as quickly as possible. This is the first general release on all

supported host systems since July, 1987, when IRAF version 2.5 was distributed. Our current plans are to release IRAF version 2.8 for the following host systems: Berkeley UNIX 4.3, Ultrix 3.0, VMS 4.7 (although indications are that the same executables may also run under 5.0), Sun-3 (OS3 and OS4), Sun-4 (OS4), Sun386i, AOSVS, Alliant, Convex, HP-UX (HP 9000 only), and possibly the Masscomp (68020 only).

An IRAF order form is attached to the back of this newsletter. Sites wanting to obtain IRAF version 2.8 should complete this order form and mail it to the address on the back of the order form. Note that IRAF version 2.8 will *not* be sent automatically to those sites already running an earlier version of the system; an IRAF order form is necessary to receive this new version.

Please be sure to fill the order form out with complete information about the host computers and operating systems you wish to run IRAF on. This information is important so that we may send you the proper IRAF system for your configuration. Please pay particular attention to the type of operating system and its version number, i.e. SunOS 3.5, SunOS 4.0, Ultrix 3.0, VMS 4.7, BSD/UNIX 4.3. Sun sites with multiple but different model Sun's should indicate that their Sun systems are standalone or that they will be operating off a file server - the SUN/IRAF system that is sent to you will depend on this information.

Our preferred medium for distribution is a 9-track tape written at 6250bpi. This is the quickest and least expensive way for us to distribute IRAF. If you have a choice please indicate this option on the order form. Even if you do not have a 9-track tape drive on your IRAF host computer you may have access to one over a network on a computer running the same type of operating system. Please investigate this alternative as well. If there is any doubt, indicate this on the order form and we will be back in touch with you before we cut your distribution tape.

We are also asking that when you send in the order form that you include any old IRAF tapes, Sun microcassettes, or TK-50's, that have been sent to you in the past, if at all possible. Be sure to pack them properly to ensure their safe arrival. Your cooperation in this matter will help us keep our costs at a minimum for this major distribution of IRAF.

This release of IRAF is being coordinated with the first release of STSDAS. All order forms that were sent to ST for STSDAS have been forwarded to us for the distribution of IRAF version 2.8. If any of the information given on the STSDAS order forms concerning the host computers, operating systems, or distribution medium is no longer current, please be sure to send us a revised order form with the current information.

Any order forms received will be held until the release of 2.8. If an earlier release is needed right away, please indicate this on the order form.

Jeannette Barnes

Current IRAF E-mail Addresses

The current electronic mail addresses for sending mail to IRAF are listed below.

UUCP: {arizona,decvax,ncar}!noao!iraf uunet!noao.edu!iraf

Internet: iraf@noao.edu

SPAN/HEPNET: 5355::IRAF or NOAO::IRAF

BITnet: iraf@noao.edu (through a gateway)

If you have any problems communicating with us electronically, our local in-house consultant is Steve Grandi, (602)-325-9228.

For those of you who like "voice" contact, the IRAF HOTLINE is still going strong - (602)-323-4160.

Jeannette Barnes

_____

The IRAF NEWSLETTER is published three times a year (February, June, and October) by the Central Computer Services, National Optical Astronomy Observatories, P. O. Box 26732, Tucson, AZ 85726. Editors: Jeannette Barnes, Doug Tody