Replacing the Convexes — New Color Algorithms in AIPS

Eric W. Greisen

September 24, 1992

Abstract

This memo summarizes the progress we have made in replacing the previously unique capabilities of NRAO's Convex C-1 computers. The situation with disks, tape devices, and basic television displays is quite good. Two new tasks have now appeared in $\mathcal{AIPS}$ to emulate the IIS Model 70 displays' capabilities to display two or three independent images at the same time. One task displays hue and intensity images together, while the other displays independent red, green and blue images at the same time and both work on workstation $\mathcal{AIPS}$ displays as well as on special-purpose devices. These tasks are not as fast or as capable of subtle colors as the IIS, but they go a long ways to replacing the last of the Convexs' unique properties.

1 Introduction

NRAO's Convex C-1 computers have served us well for a number of years. For some time, they have provided unique capabilities in fast vector computing, large quantities of striped disk space, numerous (four each) half-inch high-speed tape drives, and special-purpose television display devices. Our early work stations could not compete in any of these areas — they were not particularly fast, they had only modest local disk, they had no tape drives, and they depended on $\mathcal{AIPS}$' initial television server programs.

But times have changed.

Our current workstations are rather faster than the C-1s, especially since the C-1s are shared among multiple users while the workstations are normally restricted to single users. In particular, our new IBM RS 6000/560 workstations run the large DDT test 2.5 times faster than the C-1. Even the Sun IPX's and IPC's run this test only 1.5 and 3 times slower, respectively, than the C-1. Algorithmic adjustments to this test should improve the result according to Owen (private communication). Prices for disk (on workstations, in particular) have plummeted and we have been able to equip most workstations with at least 1 Gbyte of local space. In addition, due to the labors of Mark Calabretta and Pat Murphy, recent releases of $\mathcal{AIPS}$ (beginning with `15APR92`) have been able to share disk resources between workstations across the local area network.

Exabyte and DAT technologies have finally settled down (to single agreed formats and to low prices) allowing very high capacity tape devices to be attached to many of the workstations in the local network. Again beginning with the `15APR92` release, $\mathcal{AIPS}$ has supported "remote tapes" (Greisen, [1]), allowing users to read and write to tape devices attached to any cooperating computers on the network. The performance of tapes over the network depends on the loading of the network, of course, but it can approach that for directly-connected tape devices. These new types of tape are slower than modern half-inch drives (Greisen, [2]), but their capacity is very much greater and their small media size makes them very much more convenient. Gradually these new media will replace half-inch tapes to a large extent. Until they do, however, we will need to provide half-inch drives. NRAO Charlottesville now has two half-inch drives on the IBM RS 6000/530 called lemur which should support our requirements with little problem. The situation at the AOC in Socorro and the VLA site is only slightly more difficult. To reduce the demand for half-inch tape drives (and for reliability), the entire VLA data archive is being copied to Exabyte tapes. The operators at the VLA site have been provided with tape drives on their workstation to copy the basic half-inch data tapes to Exabyte for export from the site. The AOC now has two half-inch drives on Suns as part of the copying project and a third is anticipated. With all new data coming on Exabyte tapes, these should be enough to meet the demands to read older half-inch reels.

The special-purpose television display devices supported by the C-1s are the IIS Model 70 and IVAS displays. At the AOC, the Model 70s are equipped with NRAO-designed and built image storage devices. Both the Model 70 and the IVAS (in 8-bit mode) are capable of showing three independent images, one in each color. Our Model 70s can show up to four independent images in every color with the sum (through lookup tables) being displayed. The *AIPS* model of a television display device supports this extreme capability, but very few algorithms have required it. Perhaps more use would have been made of such powers if other special-purpose displays (*e.g.*, DeAnza, Comtal, *et al.*) had also been as capable. The earliest versions of the screen servers for workstations had very limited dynamic range (0–63 grey values) and only one graphics overlay plane. These limitations were relaxed in the `15APR92` release of the generic X-Windows server `XAS`. The screen size was made adaptable to the workstation, the number of grey memories was made site-selectable (and usually 2), the number of graphics overlay planes was raised to four (displayed as a four-bit value), and the grey range was raised to a maximum level of 199. These increases require that the graphics overlay (when selected) be imposed on the image every time the image is moved from the memory of `XAS` to the memory used by the screen display. At this stage, the special-purpose displays still outperformed `XAS` for most operations especially blinks, movies, and anything using a graphics overlay.

Greisen ([2]) discusses the next step in the process. In the `15OCT92` release of *AIPS*, `XAS` was improved to use shared memory (optionally), to support deferred display of changes, and to do its "blit" operations in a more efficient manner. These changes have made a dramatic difference in the performance of `XAS` and the numerous verbs and tasks that now take advantage of the deferred or asynchronous display. Most operations with `XAS` take more or less the same time as with the IIS displays (using the Convex). However, Greisen ([2]) stated "Of course, `XAS` cannot display true-color images, nor can it do our hue-intensity algorithm." The present memorandum describes the manner in which that statement has been rendered "inoperative."

2 Hue-Intensity Images

One of the more complicated display problems faced by *AIPS* is to use one image to control the intensity of the display and a second to control the color. Typical examples from spectral-line data would be neutral hydrogen images of zero (total HI) and first (predominant velocity) moment for colors that range from blue to red and, from continuum data, polarization images of total polarization and polarization angle for colors that range from blue through red and back to blue. The `AIPS` verb `TVHUEINT`, originally devised by Jim Torson and Arnold Rots, provides this algorithm for IIS Model-70 display devices. Using the lookup tables of the IIS, it takes the logarithm of the intensity image and the logarithm of the "color" of the hue image (using hue as the index into lookup tables of the logarithm of color filters). Then the IIS sums the two logarithmic images and the OFM lookup table takes the anti-logarithm to produce a displayed image of the product. Enhancement takes place at frame rates due to the use solely of quickly-changed lookup tables. The result has been a scientifically valuable and powerful display of paired complex images.

This lovely algorithm is not available on less powerful display devices particularly workstations with `XAS`. The `15OCT92` release of *AIPS* contains a very new task called `TVHUI` that attempts to get around this limitation. It divides the N available colors of a single TV memory (N is typically 199 for `XAS`) into n intensities and m hues, where $mn \leq N$. The eye is usually more sensitive to subtle changes in intensity than to subtle changes in hue, so we take $n \approx \beta m$ where β is a user-controlled parameter defaulted to 1.5. The grey levels (pixel values) loaded to the TV memory are then $g = i(n-1) + h(m-1)$, where i and h are intensities and hues scaled and clipped to the range 0–1, *e.g.*, $h(x,y) = \max(0, \min(1, (H - H_0)/(H_X - H_0)))$. The TV lookup table is loaded with appropriate red, green, and blue values using one of two hue-intensity to RGB translation algorithms.

Any change in the scaling of the intensity or hue images requires recomputing and reloading the pixel values. The large size of modern displays prevents us from keeping the full images in core and disk reading is too slow to be truly interactive. Therefore, to provide interactive enhancement of the hue and intensity images decimated copies of the two images are kept in core and displayed at the lower left of the TV window. The grey levels for these much smaller, in-core images can be recomputed and reloaded rapidly enough to feel interactive to the user. Whenever he/she desires, the user may press a cursor "button" and the full image will be recomputed and reloaded from disk to the display. Two-dimensional step wedges may be displayed along the top and/or right sides of the image. **TVHUI** also offers the option of writing out an RGB cube image based on the interactive scalings developed while using the task in its interactive phases. Of necessity, the hue image is scaled and clipped between 0 and 1 even in this phase. However, the intensity image is neither scaled nor clipped.

Two algorithms are available to do the conversion between hue and intensity and RGB. The default is one due to Gonzales and Woods ([3]). It is a geometric algorithm using color triangles (or pyramids actually). The hue is converted to an angle in the range 0–360 degrees and separate solutions apply in each 120-degree range. In the first 120 degrees (the "red to green sector"),

$$
\begin{aligned}
b &= \tfrac{1}{3}(1 - S) \\
r &= \tfrac{1}{3}\left[1 + \frac{S\cos H}{\cos(60^\circ - H)}\right] \\
g &= 1 - (r + b),
\end{aligned}
$$

with similar equations applying in the other thirds. Under user control, H is restricted to 0–240 degrees for velocity like displays and to 0–360 degrees for displays in which angles control hue. **TVHUI** displays fully saturated images, forcing $S = 1$.

The other algorithm (chosen by `OPTYPE = 'LUT'`) is an emulation of the Torson-Rots algorithm even to the use of lookup tables. These eliminate the need to evaluate the color filters at every pixel at the cost of using hues accurate only to about one part in a thousand. The output is given by,

$$
\begin{aligned}
R(x,y) &= I(x,y)red(h(x,y)) \\
G(x,y) &= I(x,y)green(h(x,y)) \\
B(x,y) &= I(x,y)blue(h(x,y)),
\end{aligned}
$$

where *red*, *green*, and *blue* are piecewise-linear filters evaluated via lookup tables, $I(x,y)$ is the intensity image with no scaling and clipping, and $h(x,y)$ is the scaled and clipped hue image. The parameters of the color filters are moderately arbitrary and, in fact, those that have been used in the verb **TVHUEINT** did not look good in this application. The currently chosen parameters appear appropriate, but are likely to be adjusted over time.

TVHUI has a second mode of operation for TV display devices that are capable of displaying separate images, one in each color. It is very similar to the mode described above, but offers wider dynamic range and more colors. The interactive enhancement of intensity is very fast since it can be done with lookup tables. Hue enhancement requires reloading the TV memory, however. Even with the use of a decimated image, it is at least as slow as on a workstation.

3 Three-Color Images

Three-color images are those having three separate two-dimensional images intended to be viewed superimposed, with one in red, one in green, and one in blue. They arise quite naturally from color photographs digitized through filters designed for redisplay on television devices. The long-familiar picture of the mandrill baboon is in fact a 512 x 512, 8-bit data cube with separate planes for red, green, and blue. Landsat imagery represents another overwhelming area in which separate color images need to be viewed simultaneously. Less use of three-color images has been made in radio astronomy. Primarily they arise from the attempt to represent one function with intensity and another with color. The older *AIPS* task **RGBMP** creates RGB cubes by weighted sums of the planes of spectral-line (or other) cubes while the new task **TVHUI** described above can create similar cubes. Other uses of separate colors for separate images might include images at separate wavelengths (*e.g.*, radio vs optical images), images of different polarizations, or even the same image but with very different scaling and clipping in the different colors. The last is quite effective in producing a display with a wider dynamic range than is usually available on a television screen.

Until now, an *AIPS* user wishing to view a three-color image would have required a special-purpose display device capable of displaying separate images one in each color. The very new task **TVRGB** attempts to remove this restriction. One could play the simple game, used in **TVHUI**, of dividing the available N intensities between i blues, j greens, and k reds. This yields 5 levels of blue (0–4), and 6 levels of red and green (0–5) for typical levels in **XAS**. The resulting pictures, while useful, are not very pleasing. The color encoding is simply too coarse. Therefore, **TVRGB** attempts to optimize its colors to "best" represent the actual colors in the image. Since this is slow, **TVRGB** displays the image in the coarse coloring while it computes the optimization and then redisplays the image with the "optimal" coloration. Rapid interactive enhancement is available through manipulation of the OFM lookup table with button pushes causing a re-optimization at the new scaling.

The color optimization scheme currently used by **TVRGB** is the "median-cut algorithm." The coding of the algorithm is based on **ppmquant**, a UNIX utility written by Jef Poskanzer, and his assistance, via this utility program, is gratefully acknowledged. The semi-infinite number of colors in the RGB picture is quantized into n levels per base color or n^3 total colors and a histogram of the colors is computed. The algorithm begins by treating the histogram as a single "box" and finds which color varies the most within that box. It then sorts all the colors in the box in ascending order of the chosen color and splits the box into two boxes at the median (in pixels *not* colors). The algorithm then finds the box containing the most pixels and repeats the splitting process. The algorithm terminates when N boxes are found, one for each level available in the display device. Each box is assigned a color equal to the pixel-weighted mean of the colors in the box and the display lookup tables are assigned these colors. The image is re-read, converted from colors to box numbers, and loaded to the TV. A (colorless) OFM may be used to enhance the display with no loss of information. Note that the order of colors versus box number is nearly random. (Try a **TVWEDGE** after running **TVRGB** to see the distribution. An **OFFTRAN** is also amusing, but renders the display relatively meaningless.)

4 Futures

It seems that most, if not all, of the Convex's unique capabilities have been rendered obsolete, or at least provided in some acceptable form elsewhere. The (under-utilized) image storage units at the VLA are an exception with no obvious cure. Perhaps, the sliders, knobs, and other widgets of AVS, and its memory and movie capabilities, will provide a partial replacement on the visualization computers now installed in Charlottesville and the AOC. Considerable effort will need to be expended to make AVS easier to use for radio astronomers and to provide easy connections into the *AIPS* data bases for users of the visualization systems. Equivalents of **TVHUI** and **TVRGB** will be needed in AVS to combine *AIPS* images into the correct color-vector form used inside AVS. What other additions to the AVS environment will be required is not clear at this time, at least to me.

Both of the new tasks are already being revised to enhance their flexibility and interactivity. Additionally, **TVHUI** will be modified to allow a third image, if present, to control saturation. Since the eye is not particularly sensitive to saturation, only a few levels would suffice in the interactive part of the task. With some cleverness (*i.e.*, hue is meaningless when saturation is zero), 3 (4) levels of saturation would still allow 11 (9) levels of intensity and 8 (7) of hue. These are probably enough to enhance the image interactively without the optimization algorithm. Then an RGB image can be written with essentially continuous levels of hue and saturation and optimized for display by **RGBMP**.

References

[1]Greisen, E. W., 1992, "Remote Tapes in AIPS," AIPS Memo No. 80, NRAO, Charlottesville, Virginia, June 30.

[2]Greisen, E. W., 1992, "Tape and TV Performance in AIPS," AIPS Memo No. 81, NRAO, Charlottesville, Virginia, August 26.

[3]Gonzales, R. C. and Woods, R. E., 1992, *Digital Image Processing*, Addison-Wesley Publishing Company.