Subject: adjustimg brightness of an image
Posted by akk on Sat, 17 Oct 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Hi,
 I am writing a program which scales images(i.e. images of galaxies) ,
adjusts their brightness and apparent distances, and then adds each image
to a final image, which in the end just comprises all of the images which
were added to it.

Many of the images on the final image appear very faint, and I was told
that
one can use the tvscl command to adjust brightness by using the following
syntax:

```
  tvscl<Number1>Number2
```

To my understanding this will make all pixels with values LESS
than Number1, equal to Number1 and make all pixels w/ values GREATER than
Number2
equal to Number2, while adjusting all other pixels with values in between
Number1 and Number 2 accordingly.

I've tried using the tvscl keyword (with varying values of Number1 and
Number2) but after the image is redisplayed, only a black screen appears
(with no apparent image on it). I've tried adjusting the brightness
of other already existing images, and also have had no luck.

In addition I tried making Number1 = MIN(finalimage), and Number2 =
MAX(finalimage), thinking that the final image would be redisplayed
as if i had just entered in "tvscl, finalimage". However a only blank
image is redisplayed to the screen.

I've looked through the IDL userguide books, ONLINE Help, and various web
pages, but haven't seen any information on using tvscl in this different
format (tvscl<...>...).
Does someone know another way of adjusting the brightness of an image, a
web site where this format of tvscl is explained, or
could someone tell me if I am misunderstanding the use of tvscl<..>..?

In addition when viewing my final image with "tvscl, finalimage"
the foloowing error was outputted:

```
    Program caused arithmetic error: Floating illegal operand
```
How can I correct this error?

Thanks, in advance...

Anil

p.s.: Please respond to my email address

Subject: Re: adjustimg brightness of an image
Posted by David Foster on Wed, 21 Oct 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Anil Kochhar wrote:
>
> Hi,
> I am writing a program which scales images(i.e. images of galaxies) ,
> adjusts their brightness and apparent distances, and then adds each image
> to a final image, which in the end just comprises all of the images which
> were added to it.
<snip>

I forgot to mention that if you would like to scale images such that
values are "reserved" at the top AND the bottom of the range
0 - !d.table_size-1 then you can use my BYTE_SCALE.PRO routine
that has a BOTTOM keyword as well as TOP:

```
;============================================================ ==========
; BYTE_SCALE.PRO    2-22-96  DSFoster
;
; Routine to scale the values in an array into the range 0 -
!D.TABLE_SIZE-1,
; producing the same results as IDL's TVSCL procedure for displaying
images.
;
; This procedure is useful when you need to get a BYTE version
; of an image, change/assign some of its values, and then use
; TV to display the actual image.
;
; If keyword MIN is set then only values above or equal to this in ARRAY
; will be considered. The resulting array will be scaled with MIN as its
; minimum; all elements less than MIN in ARRAY will be 0 in the result.
; Keyword MAX works the same way.
;
; If keyword TOP is set to a value then the resulting array will be
scaled
; with TOP as its maximum value. If keyword BOTTOM is set then the array
; is scaled with BOTTOM as its minimum value.
;
; 7-13-94 DSF Allow returning the scaled value of a scalar within an
;            array (if this array were scaled, the value for this
;            scalar would be scaled to...).
;  8-31-94 DSF Correct scaling to !D.TABLE_SIZE-1, not !D.TABLE_SIZE .
;  7-31-95 DSF Remove ROUND() function from final calculation to improve
speed.
;  2-23-96 DSF Improve coding.


FUNCTION byte_scale, array, example, MIN=min, MAX=max, BOTTOM=bottom,
TOP=top

result = 0

if (keyword_set(BOTTOM)) then begin
   botval = 0 > bottom
endif else begin
   botval = 0
endelse

if (keyword_set(TOP)) then begin
   topval = top < (!D.TABLE_SIZE-1)
endif else begin
   topval = !D.TABLE_SIZE - 1
endelse

if (keyword_set(MIN)) then begin
   minval = min
endif else begin
   minval = min(array, MAX=array_max)
endelse
if (keyword_set(MAX)) then begin
   maxval = max
endif else begin
   if (n_elements(array_max)) then begin
      maxval = array_max
   endif else begin
      maxval = max(array)
   endelse
endelse

if (n_elements(array) lt 2) then begin
   message, 'First argument must be an array', /continue
   result = -1
endif
if (topval le botval) then begin
   message, 'Keyword BOTTOM must be less than keyword TOP', /continue
   result = -1
endif
if (maxval le minval) then begin
   message, 'Keyword MIN must be less than keyword MAX', /continue
   result = -1
endif

if (result eq 0) then begin
   if (keyword_set(MIN) or keyword_set(MAX)) then $
      ; Limit to MIN, MAX
      array = ((minval-1) > array) < (maxval+1)

   constant = FLOAT(topval - botval) / FLOAT(maxval - minval)

   if (n_elements(example)) then begin
      result = BYTE( constant * (example - minval) + botval )
   endif else begin
      result = BYTE( constant * (array - minval) + botval )
   endelse
endif

return, result
END
```

========== BYTE_SCALE.DOC ==

BYTE_SCALE

Use this routine to scale the values in an array
into the range 0 - !D.TABLE_SIZE-1, giving the same
results as IDL's TVSCL procedure.

You can use the keyword TOP to specify a different
maximum value for the resulting array, and BOT to
specify a minimum value other than zero. Use MIN
and MAX to specify the mininum and maximum values
in the original array to consider when scaling. Use
these to produce uniformly scaled images by
specifying the same values for MIN and MAX.
Otherwise each image will be scaled according to
its own minimum and maximum.

If a second argument is included, then BYTE_SCALE
will return the value it would have if scaled
according to specified parameters. Use this to
find what a specific value would be scaled to.

Calling Sequence

```
  Results = BYTE_SCALE(Array [,Element])
```

Inputs

Array

The array whose values are to be scaled and returned.

Element

If this optional argument is included then BYTE_SCALE returns the value it would be scaled to, if Array were to be scaled (so Example should be a value contained in the original unscaled Array).

Outputs

Returns an array of the same dimensions as Array, with the values scaled appropriately, unless argument Element is supplied, in which case it returns the scaled value of Element only (scalar).

Keywords

BOTTOM

Set this to specify an alternate minimum value for the scaled array, to scale the values from BOT to !D.TABLE_SIZE-1 (or TOP). This defaults to zero.

MIN, MAX

Use these to specify the minimum/maximum value in the array to consider when scaling. Use the same values with multiple images to produce uniformly scaled arrays. If MIN/MAX is not set then the minimum/maximum of the array will be used.

TOP

Set this to specify an alternate maximum value for the scaled array, to scale the values from 0 (or BOTTOM) - TOP. If not set then !D.TABLE_SIZE-1 is used (the number of available colors...this routine is intended primarily for images).

Example

To scale an array into a range half the size of
the number of available colors (10 - !D.TABLE_SIZE/2),
with a minimum value of 10 (say you want to reserve
colors 0-9 for the interface):

```
  byte_image = BYTE_SCALE(image, BOTTOM=10, $
        TOP=!D.TABLE_SIZE/2)
```

== =========
--

~~~~~~ ~~~~~~~
David S. Foster   Univ. of California, San Diego
Programmer/Analyst   Brain Image Analysis Laboratory
foster@bial1.ucsd.edu Department of Psychiatry
(619) 622-5892   8950 Via La Jolla Drive, Suite 2240
                       La Jolla, CA 92037
~~~~~~ ~~~~~~~