Subject: Re: Map_set limits
Posted by Martin Schultz on Mon, 01 Dec 1997 08:00:00 GMT
View Forum Message <> Reply to Message

This is a multi-part message in MIME format.

--------------15FB59E21CFB
Content-Type: text/plain; charset=iso-8859-1
Content-Transfer-Encoding: 8bit

>>
>>> I was trying to superimpose satellite images to a map in IDL 4, and had a
>>> problem:
>>>
>>> I have to set the map's dimensions to the image's to get a good match,
>>> using the keyword "position" in map_set.
>>> After several tries I found out that the map limits are enlarged by 2% in
>>> both directions (long/lat), while the dimensions are unchanged. The result
>>> is that I get a map of a larger area in the dimensions of my image.
>>
>> Unfortunately, IDL's map projection routines are not designed
>> to put a map projection on an image. (I am, however, sympathetic
>> to the argument that they *should* be.) Rather, they are
>> designed to put an image on a map projection.
>>
>
> I've tried this one first, yes. But it looks more like a drawing function
> than anything else to me. The problem is that, when you work with remote
> sensing or space images for instance, you simply need to superpose a
> geographic grid to perform automatic measurements. You don't want to
> degrade the image quality at all because the information you need is in
> there, and it was expensive to get it. In short you need to do something
> like Image_contour does for plots and images, and find out that map_set
> has these weird peculiarities.

Yes, that's what I found as well. You may want to take a look at my
image_map routine which I derived from image_contour (attached below).
It is not completely flexible because I set it up rather quickly to do
some analysis of data from geostationary satellites over the Pacific,
but it should give you a general idea what you can do. I marked my fudge
parameters; this is where you will have to spend some work if you want
to generalize the routine (in fact I would be VERY interested in the
result). I guess, the only key to success here is to use the satellite
projection and play around with its parameters (better of course, if you
know them). The major trouble I had, was to produce a postscript file
which would look similar to my screen image. THIS may really be a
typical David Fanning thing to answer.

Here is a sample call:

```
; ------------------------
read_jpeg,'~/download/gte/249_2100ful1.jpg',satim
satim = satim(93:1041,23:927)     ; cut off border

image_map,satim,/conti
; ------------------------
```

Regards,
Martin

>
> Yet, if somebody felt the need to change the maps limits in map_set there
> was probably a good reason. So the question is: why? And are there
> situations in which you really have to perform this strange
> transformation. I find it very surprising that apparently nobody ran into
> this problem before.
>

-- -------
Dr. Martin Schultz
Department for Earth&Planetary Sciences, Harvard University
186 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA

phone: (617)-496-8318
fax : (617)-495-4551

e-mail: mgs@io.harvard.edu
IDL-homepage: http://www-as.harvard.edu/people/staff/mgs/idl/
-- -------


```
--------------15FB59E21CFB
Content-Type: text/plain; charset=us-ascii; name="image_map.pro"
Content-Transfer-Encoding: 7bit
Content-Disposition: inline; filename="image_map.pro"

;+
; NAME:
; IMAGE_map
;
; PURPOSE:
; Overlay an image and a map (satellite projection)
;
```

```
; CATEGORY:
; General graphics.
;
; CALLING SEQUENCE:
; IMAGE_map, A
;
; INPUTS:
; A: The two-dimensional array to display.
;
; KEYWORD PARAMETERS:
; WINDOW_SCALE: Set this keyword to scale the window size to the image size.
;  Otherwise, the image size is scaled to the window size.
;  This keyword is ignored when outputting to devices with
;  scalable pixels (e.g., PostScript).
;         [original as in image_contour]
;
; ASPECT: Set this keyword to retain the image's aspect ratio.
;  Square pixels are assumed.  If WINDOW_SCALE is set, the
;  aspect ratio is automatically retained.
;         [original as in image_contour]
;
; INTERP: If this keyword is set, bilinear interpolation is used if
;  the image is resized.
;         [original as in image_contour]
;
;    CENTERX: longitudinal position of geostationary satellite
;         (default -135 = GEOS-9)
;
;    DIST: distance of satellite from Earth surface (in earth radii)
;         (default = 7)
;
;    CONTINENTS: superimpose map continents on the image
;
; OUTPUTS:
; No explicit outputs.
;
; COMMON BLOCKS:
; None.
;
; SIDE EFFECTS:
; The currently selected display is affected.
;
; RESTRICTIONS:
; None.
;
; NOTES:
;    Derived from IDL routine image_contour.
;    Not very flexible - quick hack to analyze PEM-T data
```

```
;
; PROCEDURE:
; If the device has scalable pixels, then the image is written over
; the plot window.
;
; MODIFICATION HISTORY:
; mgs, Oct 1997 : based on IMAGE_CONT by DMS, May, 1988.
;-

pro image_map, a, WINDOW_SCALE = window_scale, ASPECT = aspect, $
 INTERP = interp, DIST=dist, CENTERX=centerx, continents=continents


on_error,2                    ;Return to caller if an error occurs
sz = size(a)   ;Size of image
if sz(0) lt 2 then message, 'Parameter not 2D'

six = float(sz(1))  ;Image sizes
siy = float(sz(2))
aspi = six / siy  ;Image aspect ratio

dvx = !d.x_vsize
dvy = !d.y_vsize
aspd = float(dvx) / float(dvy)

; *** HERE ARE SOME FUDGE PARAMETERS AND DEBUG OUTPUT ***
!p.position=[(1.-aspi/aspd)/2.,0.05,(1.+aspi/aspd)/2.,0.95]
print,(1.-aspi/aspd)/2.,(1.+aspi/aspd)/2.,aspd,aspi

; *** Position of the satellite ***
if (not keyword_set(dist)) then dist=7.
if (not keyword_set(centerx)) then centerx=-135.

; *** set-up the map in satellite projection ***
map_set,0,centerx,/satellite,sat_p=[dist,0.,0.]

; *** DEBUG output ***
print,'!d.x_vsize,!d.y_vsize : ',!d.x_vsize,!d.y_vsize
print,'!x.window,!y.window : ',!x.window,!y.window
 ;set window used by contour

; *** old contour command #1 deactivated ***
; contour,[[0,0],[1,1]],/nodata, xstyle=4, ystyle = 4

px = !x.window * !d.x_vsize ;Get size of window in device units
py = !y.window * !d.y_vsize
swx = px(1)-px(0)  ;Size in x in device units
swy = py(1)-py(0)  ;Size in Y
```

```
aspw = swx / swy  ;Window aspect ratio
f = aspi / aspw   ;Ratio of aspect ratios

; *** DEBUG output ***
print,'aspw,aspi,f : ',aspw,aspi,f

if (!d.flags and 1) ne 0 then begin ;Scalable pixels?
  if keyword_set(aspect) then begin ;Retain aspect ratio?
    ;Adjust window size
 if f ge 1.0 then swy = swy / f else swx = swx * f
 endif

; *** Here are my attempts to match the image and map for postscript output
; (scalable pixels)
; tvscl,a,px(0)*1.04,py(0)*1.04,xsize = 0.98*swx, ysize = 0.98*swy, /device
  tvscl,a,px(0)*1.08,py(0)*1.20,xsize = 0.98*swx, ysize = 0.98*swy, /device
print,'px(0),px(1) : ',px(0),px(1)

endif else begin ;Not scalable pixels
   if keyword_set(window_scale) then begin ;Scale window to image?
 tvscl,a,px(0),py(0) ;Output image
 swx = six  ;Set window size from image
 swy = siy
    endif else begin  ;Scale window
 if keyword_set(aspect) then begin
  if f ge 1.0 then swy = swy / f else swx = swx * f
  endif  ;aspect

; *** and here for the screen (not scalable) ***
 tv,poly_2d(bytscl(a),$ ;Have to resample image
  [[0,0],[1.02*six/swx,0]], [[0,1.02*siy/swy],[0,0]],$
  keyword_set(interp),swx,swy), $
  px(0)+5,py(0)+5
 endelse   ;window_scale
  endelse   ;scalable pixels

mx = !d.n_colors-1  ;Brightest color
colors = [mx,mx,mx,0,0,0] ;color vectors
if !d.name eq 'PS' then colors = mx - colors ;invert line colors for pstscrp


; *** old contour command #2 deactivated ***
; contour,a,/noerase,/xst,/yst,$ ;Do the contour
;     pos = [px(0),py(0), px(0)+swx,py(0)+swy],/dev,$
;  c_color =  colors

; *** here is the map ! ***
map_grid,color=2,glinestyle=0,londel=15,latdel=15
```

```
if(keyword_set(continents)) then map_continents,color=7

return
end

--------------15FB59E21CFB--
```