Subject: Map_set limits
Posted by Erard on Tue, 25 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

Hello.

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.

So far, I have solved this problem by either increasing the map's dimensions or by reducing the limits in long/lat by 2% before calling Map_set, depending on the type of image I'm processing and the output.

I'm now in the process of writing of different version of Map_set adapted to planetary mapping, and I want to get rid of this kind of problems. Does anybody know why the position parameters are modified in map_set, and whether I can safely spare this modification?

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

Hi Folks,

I have to admit that I am confused about recent postings

having to fiddle with the Position parameters to get a
Map_Set command to fit the proper Limits around an image.

an image and part of his code, which he graciously did.
I've been fooling around with it a little in my "spare"
time. (Please don't tell my wife I am doing this. I am
in hot water already.) In truth, I can't reproduce the
problem. But, on the other hand, I am not absolutely
certain that I can recognize the problem either.

If I position an image in the window in such a way
that it takes up, say, 80% of the window space and then
try to position a map projection on it like this:

```
  Map_Set, /Sin, Limits=theLimits, Position=[0.1,0.1,0.9,0.9]
```

then in IDL 5 it seems to do this *exactly*. That is to
say, the Map_Set axes are coincident with the edges of
the image. If I now lay a grid on the image, then the
grid *appears* to go exactly where I want it.

What I am unsure about is if the grid is going *exactly*
where it is suppose to go. How would I know? You folks
presumable have some kind of ground point or something
that you know the exact latitude and longitude coordinate
of. If I do the same thing with a data set I am more
familiar with (say, the worldelv.dat data set in IDL), then
it still appears to do the right thing.

In any case, I am all ears. :-)

Cheers,

David

P.S. With respect to Martin's problems with images and
PostScript, I find I always use TVImage (available on
my web page) for positioning images in windows. You can
use the Position keyword with it like you do for a contour
or line plot. The advantage of this is that the routine is
Device independent. It works the same way in PostScript
as it does on the display.

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438
Coyote's Guide to IDL Programming: http://www.dfanning.com/

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--

Subject: Re: Map_set limits
Posted by Stephane Erard on Tue, 02 Dec 1997 08:00:00 GMT

David Fanning wrote:
> =

> I have to admit that I am confused about recent postings
> from St=E9phane Erard and Martin Schultz, who complain about
> having to fiddle with the Position parameters to get a
> Map_Set command to fit the proper Limits around an image.
>
> [...]
> =

> What I am unsure about is if the grid is going *exactly*
> where it is suppose to go. How would I know? You folks
> presumable have some kind of ground point or something
> that you know the exact latitude and longitude coordinate
> of. If I do the same thing with a data set I am more
> familiar with (say, the worldelv.dat data set in IDL), then
> it still appears to do the right thing.
> =

> In any case, I am all ears. :-)
> =

Ok, I'll try to make this problem a little clearer. I answer in the Newsgroup since this issue probably interests other people.

Included is a small piece of source code that plots a map. The map is a subset of the USGS shaded relief map of Mars, available for instance at http://www-pdsimage.jpl.nasa.gov/PDS/. The subset is saved as ptcarte.idl (it can be found at ftp://ftp.ias.fr/ias/erard), but any image of this size will do to figure out the problem. You can see a gif version of the (almost) final figure at http://www.ias.fr/cdp/ISM/specomp.html

If you run the following code in two versions, you'll see a difference:
- as is, makes a correction to the map dimension, and the output is just fine: the final product has the same number of pixel that the original map file and the image fills the coordinates grid.
- If now you comment out the 6 lines beginning with del=3D... (ie use map_set to plot the map without precautions) the output is different (at least in versions 4 and 3.6). The frame now has the dimension of the image (720 pixels in this case), so the image fits in the frame rather than in the lat/long grid, which is 2% smaller.

The (spurious) transform is made in Map_uv_bound, a function in the

map_set.pro file (look for "fudge" in the text). I understand that map_uv_bounds tries to plot a slightly larger frame so we can see the axes clearly in the frame. Practically, it enlarges the lat/long limits by 2% and plots these limits in the area reserved by the keyword position in map_set. By doing so, it establishes a correspondance between the area selected and lat/long limits that are 2% in excess. In the exemple code, it sets the frame's right border to long=3D 20+(20-(-70))*0.01.
If you don't set the map limits everything is fine - this explains why there is no problem with the whole Earth map.

Now this problem is important, because:
- if you work with geographical or celestial images you know the lat/long limits precisely, but you get a shrinked lat/long grid; this precludes any precise position measurement of features of interest;
- if you try and superimpose another imaging data set on the first image, IDL will use the coordinate system set by plot in map_uv_bound, therefore corresponding to an area smaller than the first image. This precludes precise superposition of geographical data sets, namely construction of data cubes through the graphic interface. This is a _major_ problem for a plotting/computing application (of course, this will happen only if you plot the second data set with oplot or polyfill).

My original question was about the reason why somebody took the pain to make this change (modication made in march 1993, that should be version 3.6). Perhaps David is right in suggesting I'm optimistic in supposing there is a reason, but still... I'm planning to adapt map_set to planetary mapping, so I need to correct also problems like this one. If it was a bug correction, I wouldn't like to resurect an old ghost. Anybody has an idea?

```
Pro bid, d, PS=3DPS

;  =3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D
=3D=3D=3D=3D=
 =3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D
=3D=3D=3D=3D=3D=
=3D=3D=3D

; example version from Mapirs5.pro. Only the map is plotted, no
additional data
;  =3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D
=3D=3D=3D=3D=
 =3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D=3D
```

```
=3D=3D=3D=3D=3D=
=3D=3D=3D

common colors,r_o,B_O,G_O,R,G,B

x=3Dfltarr(4)
y=3Dfltarr(4)
tab=3Dbytarr(1440,723)
lat=3Dfltarr(4)
lon=3Dfltarr(4)
!P.font=3D0
if keyword_set(ps) then begin
 set_plot,'ps'
 !P.font=3D0
 coefPS=3Dmin([!D.X_vsize/850.,!D.Y_vsize/420.]) ;image on the whole shee=
t

 device,filename=3D'Mapirs5.ps',/color,bits=3D8,/landscape,/B old,scale_fac=
tor=3D1.
endif else begin
 window, /free, xsize=3D850, ysize=3D420
 coefPS=3D1.
endelse

loadct, 0
l=3Dreplicate(0.7,16) ; define the rest of the color map for other data
s=3Dreplicate(1.,16)
h=3Dfindgen(16)*290./16.
h=3Dreverse(h)
h(0)=3D0.
L(0)=3D0.5
S(0)=3D1.
tvlct, H,L,S, 212,/HLS
tvlct, ro, gr,bl, /get


     ; following lines select the area of interest from the complete
map.
;e=3Da(440:799,280:439)  ; lat: -20 to 20, long: -70 to 20 on the equator=

;a=3D1             ; (terrestrial convention for longitude). =

;d=3Drebin(e,720,320)
;save, filename=3D'ptcarte.idl', d,/XDR

 ; Reads the PDS map, enlarged, of the selected area
 ; (complete map is 1440 x 720). Sinusoidal projection
restore, 'ptcarte.idl' =
nbcol=3D211. ; first 201 values for the map
d=3Dbytscl(d,top=3Dnbcol)

sz=3Dsize(d)
px0 =3D20.  ;shift a little
py0 =3D20.
print, sz, coefPS, !D.x_vsize,!D.y_vsize
qx=3D[px0,px0+sz(1)+0.5]/!D.x_vsize*coefPS ; Back to normal coord for
Map_set
qy=3D[py0,py0+sz(2)+0.5]/!D.y_vsize*coefPS
print, qx, qy

    ; these lines must be commented out to see what map_set does by
itself
del=3D(qx(1)-qx(0))*0.01  ;Compensate limits transform in
qx(0)=3Dqx(0)-del        ; MAP_SET by a similar enlargement
qx(1)=3Dqx(1)+del        ; of the plot.
del=3D(qy(1)-qy(0))*0.01
qy(0)=3Dqy(0)-del
qy(1)=3Dqy(1)+del
print, qx, qy, del

la=3D[-20,20]
lo=3D[-70, 20]     ;At the equator
lax=3D(la(0)+la(1))/2.
lo1=3Dlo/cos(lax*!dtor)     ;limits in long on top (latmax)
print, lo1, la
c0=3D[lax,lo1(0)]  ; coord on the border centre =

c1=3D[la(1),(lo(0)+lo(1))/2.]
c2=3D[lax,lo1(1)]
c3=3D[la(0),(lo(0)+lo(1))/2.]
;print, lo1
tv, d, px0*coefPS,py0*coefPS,xsize=3Dsz(1)*coefPS, ysize=3Dsz(2)*coefPS
map_set, /sinu,/grid, limit=3D[c0,c1,c2,c3],/noerase, /label,title=3D$
 'IRS-ISM overlaping observations of Mars!C',latlab=3D-72,lonlab=3D-22,$
  londel=3D10.,latdel=3D10.,position=3D[qx(0),qy(0),qx(1),qy(1 )],$
 glinethick=3D2.,glinestyle=3D0


maxi=3D0.5
mini=3D0.


;superpose another data set, no matter.
```

```
xyouts, 0,0, '!4'
colors=3Dreverse(findgen(15)+213)       ; Couleurs albedo
citems=3Dstrarr(15)
citems(0)=3Dstring(format=3D'(F5.2)',maxi/32537./2.)
citems(7)=3Dstring(format=3D'(F5.2)',(maxi+mini)/32537./4.)
citems(14)=3Dstring(format=3D'(F5.2)',mini/32537./2.)
     ; use a routine from ASTRON
legend,  citems,/fill,psym=3D8+intarr(15),colors=3Dcolors,char=3D1.,$
 pos=3D[qx(1)+0.01,qy(1)],/normal
xyouts, qx(1)-0.015,qy(1)+0.02,/normal, 'ISM reflectance',charsize=3D0.8

!P.font=3D-1
if keyword_set(ps) then begin
 device,/close
 set_plot, 'x'
endif =



end
```

St=E9phane Erard

Institut d'Astrophysique Spatiale
Universit=E9 Paris-Sud, bat. 121 F-91405 Orsay, France
tel : 33 1 69 85 86 41   fax : 33 1 69 85 86 75
e-mail : Erard@ias.fr

Subject: Re: Map_set limits
Posted by nick on Mon, 08 Dec 1997 08:00:00 GMT

View Forum Message <> Reply to Message

In article <3483EC71.7861@ias.fr>, =?iso-8859-1?Q?St=E9phane?= Erard
<Erard@ias.fr> writes:
|>
|> My original question was about the reason why somebody took the pain
to
|> make this change (modication made in march 1993, that should be
version
|> 3.6). Perhaps David is right in suggesting I'm optimistic in
supposing
|> there is a reason, but still... I'm planning to adapt map_set to
|> planetary mapping, so I need to correct also problems like this one.
If

|> it was a bug correction, I wouldn't like to resurect an old ghost.
|> Anybody has an idea?
|>
|>
i had a discussion about this with idl support a couple of years ago.
i often plot a rectangular array of global data, and overlay the
continents.
when version 3.5.1 came out, i began to see that the overlay didn't
quite
reach the edges of the window, so that the outline no longer matched
the
lat/lon of the data. supposedly setting xmargin/ymargin to zero was
supposed
help with this (at least as i read the documentation) but didn't.

after some convincing, i received the following e-mail form rsi tech
support,
with some explanation of why the change was made to map_set.

> The following is a suggested workaround from development here at RSI
> regarding the problem you were having with map_set.pro. I hope this
> helps.
>
> Sincerely,
> Mark
> RSI
>
> ---
> suggest that for now you tell the customer to copy map_set.pro
> and make the following change.
>
>   About line 300, is the curious line,
>
> fudge=0.01
>
> Which the user should replace with something like,
>
> if not(border) and total(xmargin) eq 0 and total(ymargin) eq 0 then $
>   fudge=0.0 else fudge=0.01
>
> The 'fudge' is a protector against near zero values of the
transformation
> coordinates (U,V). On some projections, this is necessary, while on
others
> it may not be. I agree that there should be some way to turn this off,
so
> I propose this solution.

--

Nick DiGirolamo
nick@boingo.gsfc.nasa.gov