
Subject: Making color images from 3 grayscale images

Posted by [roy](#) on Thu, 20 Jan 1994 18:37:43 GMT

[View Forum Message](#) <> [Reply to Message](#)

Does anybody have experience with constructing a true color image from three grayscale images (R+G+B) in IDL?

I have several astronomical images that have been made in appropriate wavebands, and would very much like to construct real RGB images for publishing.

If anybody out there have the IDL programs that does this kind of stuff, please let me know!

Roy OEstensen * E-mail: roy@mack.uit.no ** "Sometimes I think I
Auroral Observatory * ** understand everything
University of Tromsøe * PLEASE NOTE: The letters ** ---- Then I regain
Norway * oe and OE is \o and \O in TeX ** consciousness..."

Subject: Re: Making color images from 3 grayscale images

Posted by [saken](#) on Fri, 21 Jan 1994 16:41:06 GMT

[View Forum Message](#) <> [Reply to Message](#)

In article <1994Jan20.183743.1398@news.uit.no>, roy@mack.uit.no (Roy Oestensen) writes:

|> Does anybody have experience with constructing a true color image

|> from three grayscale images (R+G+B) in IDL?

|>

|> I have several astronomical images that have been made in appropriate

|> wavebands, and would very much like to construct real RGB images for

|> publishing.

|>

|> If anybody out there have the IDL programs that does this kind of stuff,

|> please let me know!

|>

|>

|> Roy OEstensen * E-mail: roy@mack.uit.no ** "Sometimes I think I

|> Auroral Observatory * ** understand everything

|> University of Tromsøe * PLEASE NOTE: The letters ** ---- Then I regain

|> Norway * oe and OE is \o and \O in TeX ** consciousness..."

|>

The IDL manual talks at great length about this in the Postscript section. I've followed their procedure with great success. However, I find it very useful to use the COLOR_QUAN procedure to "preview" the image on an 8-bit display. Slight changes in the scaling for your input color planes can have a large effect on the final color of the image.

Subject: Re: Making color images from 3 grayscale images
Posted by [knipp](#) on Mon, 24 Jan 1994 09:56:50 GMT
[View Forum Message](#) <> [Reply to Message](#)

I've found an old program we wrote some time ago to

- display 3 8-bit-images in pseudo-color on an 8-bit screen
- save as 8-bit SunRaster / or 24-bit PostScript.(opt. Encapsulated)

check_filepath: checks path in filename
 check_keyword: checks keyword if defined, sets it to 0 or 1
 exist_file(): checks file-existence via FINDFILE
 get_win_no(): returns index of NEXT window being opened
 is_idl(): returns 1, if IDL, or 0 if not (this routine
 has been 1st written for Wave on a Sun360)
 rm_file: deletes file via OPENR,/DELETE
 write_srf vs: almost the same as WRITE_SRF

// _ \\ // Karlheinz Knipp phone: +49 511 - 762 4922
 /// /_ /// University of Hannover fax: +49 511 - 762 2483
 // _ _ // Institute for Photogrammetry
 /// / / Nienburger Str.1
 / // / / / FRG 30167 Hannover 1 email: knipp@ipi.uni-hannover.de

```

;+
pro rgb_tv, imar, imag, imab, $
    output_file=output_file, bscale_wanted=bscale_wanted, $
    xpos=xpos, ypos=ypos, disp=disp, post_file=post_file, $
    ps=ps, xoffset=xoffset,yoffset=yoffset,xsize=xsize,ysize=ysize, $
    on screen=on screen, $

```

rct=rct, gct=gct, bct=bct, rcode = rcode

; display/creation 3 color composites in true color mode

; -----
; START OF DESCRIPTION

; routine IPI, Uni Hannover 12'92, modified 12'92
; (has been : RGB_DISP ('90) by vs)
; 08'93: modified to display large image
; (IDL only), skipped TIF
; add. keywords R(G,B)CT

; METHOD: display/creation 3 color composites in true color mode
; device has to be set to 256 colors
; display via scaling/bitshift

; opt. : create PostScript-file (either pure PS or EPSI)
; control via keyword PS (0 : no PS
; 1 : PS
; 2 : EPSI)

; INPUT PARAMETER: imar red image channel (byte-array)
; imag green image channel "
; imab blue image channel "

; OUTPUT PARAMETER: none

; INPUT KEYWORDS: output_file if set SRF-file to store result to [no store]
; bscale_wanted if true use bytscl instead of hist_equal [OFF]
; x(y)pos window - position (def.: mid-screen)
; disp if true, display messages [ON]
; on_screen if true, display image on screen [ON]
; ps 0 : no PS [DEF]
; 1 : pure PS
; 2 : EPSI
; post_file name of file for PS-output ['\$HOME/rgb_tv.ps']
; x(y)offset PS - page-margin [cm] (def = 4.)
; x(y)size PS - image-size on page [cm](def. = 6.5,10.5)

; OUTPUT KEYWORDS: r(g,b)ct red (green,blue) color-table entries
; rcode 0 : o.k.
; 1 : o.k., but no output to screen
; -1 : FAILURE

; EXAMPLE:

; print, 'no example - blame k.knipp'

```

;
; END OF DESCRIPTION
; -----
;-

rtcode = -1 ; Return - code

ON = 1 ; Flags for check_keyword
OFF = 0 ; (or for use as TRUE/FALSE)

check_keyword, disp, ON

; -----
; test input

if disp then print, ' testing ...'

if n_params() ne 3 then begin
    print,"The number of parameters was wrong:",n_params(),'/007'
    doc_library,"rgb_tv"
    return
endif

; -----
; get current color tables, definitions

common colors, r_orig, g_orig, b_orig, r_curr, g_curr, b_curr

dummy_char = ''
muell_window = 0
maxr = 2^0 + 2^1 + 2^2 ; ?????????????????????????????????????????
maxg = 2^0 + 2^1 + 2^2 ; ?????????????????????????????????????????
maxb = 2^0 + 2^1 ; ?????????????????????????????????????????

; -----
; test keywords

check_keyword, on_screen, ON
check_keyword, bscale_wanted, OFF

if n_elements(output_file) eq 0 $
    then out_wanted = 0 $
    else begin

```

```

out_wanted = 1
output_file = strtrim(string(output_file(0)),2)
check_filepath, output_file
endelse

```

```

if n_elements(ps) eq 0 then ps = 0 ; check for PS !!!!!!!!!!!!!!!!!!!!!!!

```

```

case ps of
1: epsi = 0
2: epsi = 1
else: ps = 0
end

```

```

if ps ne 0 then begin

```

```

if n_elements(xoffset) eq 0 then xoffset=4.d
if n_elements(yoffset) eq 0 then yoffset=4.d
if n_elements(xsize) eq 0 then xsize=6.5d
if n_elements(ysize) eq 0 then ysize=10.5d

```

```

if n_elements(post_file) eq 0 then begin
post_file = '$HOME/rgb_tv.ps'
message, 'no name for output defined ...', /info
message, 'setting to default : '+ post_file, /info
endif

```

```

check_filepath, post_file

```

```

if exist_file(post_file) then begin
beep
print, ' File already exists : ',post_file
print
print, ' Enter d to delete file'
print, ' or q to quit'
dummy = strlowcase(get_kbrd(1))
print
if dummy eq 'q' then begin
print, ' EXIT on user-request '
return
endif
rm_file, post_file, confirm=0
endif

```

```

endif

```

```

; -----

```

```

; test images

s_ima_r = size(imar)
s_ima_g = size(imag)
s_ima_b = size(imab)

if s_ima_r(0) ne 2 or $
    s_ima_g(0) ne 2 or $
    s_ima_b(0) ne 2 then begin
    beep
    message, 'at least one channel is NOT 2.dim. ...', /info
    return
endif

if s_ima_r(3) ne 1 or $
    s_ima_g(3) ne 1 or $
    s_ima_b(3) ne 1 then begin
    beep
    message, 'at least one channel is NOT 8-bit deep (BYTE) ...', /info
    return
endif

cols = s_ima_r(1)
rows = s_ima_r(2)

if s_ima_g(1) ne cols or s_ima_g(2) ne rows then begin
    beep
    message, '2nd channel does NOT fit dimensions of 1st one ...', /info
    return
endif

if n_elements(xpos) eq 0 then xpos = (!xscreen_size - cols) / 2 $
    else xpos = long(xpos(0)) > 0

if n_elements(ypos) eq 0 then ypos = (!yscreen_size - rows) / 2 $
    else ypos = long(ypos(0)) > 32

; -----
; POSTSCRIPT

if ps ne 0 then begin

; -----
; store device-name, set to PostScript

if disp then message, 'PS : storing device-name ...', /info
save_device = !d.name

```

```

set_plot,'PS'

; -----
; follow Sasses-approach in RGB_DISP

if disp then message,'PS : defining parameters ...', /info

if is_idl()      $
  then device, /color, filename=post_file,  $
  xoffset=xoffset,yoffset=yoffset,xsize=xsize,ysize=ysize, $
  preview=epsi, encapsulated=epsi, inches=0  $
  else device, /color, filename=post_file,  $
  xoffset=xoffset,yoffset=yoffset,xsize=xsize,ysize=ysize, $
  epsi=epsi, encapsulated=epsi, inches=0

print
help, xoffset, yoffset, xsize, ysize, epsi,post_file

loadct, 0
ima = replicate(0b,cols,rows,3)
if disp then message, 'copy RED into buffer ...', /info
ima(*,*,0) = imar
if disp then message, 'copy GREEN into buffer ...', /info
ima(*,*,1) = imag
if disp then message, 'copy BLUE into buffer ...', /info
ima(*,*,2) = imab
if disp then message, 'print buffer to PostScript-file ...', /info
tv, ima, true=3

; -----
; reset device

if disp then message, 're-store device to ' + save_device, /info
device, /close_file
set_plot, save_device

endif

; -----
; TIFF

; -----
; if not on screen, return

if not on_screen then begin

```

```

if disp then message, 'no display on screen ..', /info
rtcode = 1
return
endif

```

```

; -----
; check device

```

```

if !d.name ne 'X' and $
    !d.name ne 'SUN' then begin

```

```

    beep
    message, 'unsupported device : ' + !d.name, /info

```

```

    return

```

```

endif

```

```

; -----
; for SUN device with only 8 bit, following table show bits to be set for color
; bit:  7    6    5    4    3    2    1    0
;      <  red  >   < green >   < blue >

```

```

if !d.window eq -1 then begin
    window,/free,xsize=4,ysize=4,colors=256
    muell_window = 1
    m_win        = !d.window
endif

```

```

if !d.n_colors ne 256 then begin
    print,'256 colors are necessary, only ',!d.n_colors, $
    string(7b),'          are available !!'
    if muell_window then wdelete, m_win
    return
endif

```

```

if disp then begin
    t0 = systime(1)
    print, format=('$," scaling ...",a14)', dummy_char
endif

```

```

if bscale_wanted then begin

```

```

    imarh = bytscl(imar, top=maxr)      ; reduce no. of greys

```

```

    imagh = bytscl(imag, top=maxg)
    imabh = bytscl(imab, top=maxb)

endif else begin

    imarh = hist_equal(imar, top=maxr)      ; reduce no. of greys
    imagh = hist_equal(imag, top=maxg)
    imabh = hist_equal(imab, top=maxb)

endelse

if disp then begin
    t1 = systime(1)
    print, format = '(f7.3,a6)', t1-t0," [sec]"
    t0 = t1
    print, format=('$," shifting bits ...",a8)', dummy_char
endif

imarh = ishft(imarh,5)                      ; bit shift to left
imagh = ishft(imagh,2)

ima = imarh + imagh + imabh                  ; new image

;-----
; create lookup table and load it and display

if disp then begin
    t1 = systime(1)
    print, format = '(f7.3,a6)', t1-t0," [sec]"
    t0 = t1
    print, format=('$," creating lookup-tables ...")
endif

nc = 256
rgb = intarr(nc,3)
r = intarr(nc)
g = intarr(nc)
b = intarr(nc)

rfaktor = float(nc-1) / (1.d * maxr)
gfaktor = float(nc-1) / (1.d * maxg)
bfaktor = float(nc-1) / (1.d * maxb)

i = 0
for rr=0,maxr do begin
    for gg=0,maxg do begin

```

```

    for bb=0,maxb do begin
        r(i) = fix(rr * rfaktor)
        g(i) = fix(gg * gfaktor)
        b(i) = fix(bb * bfaktor)
        i = i + 1
    endfor
endfor
endfor

tv!ct,r,g,b
rct = r
gct = g
bct = b
r_curr = r
g_curr = g
b_curr = b

if disp then begin
    t1 = systime(1)
    print, format = '(f6.3,a6)', t1-t0," [sec]"
    t0 = t1
    print, format='($," displaying ...",a11)', dummy_char
endif

if cols gt (!xscreen_size + 256) and $
    rows gt (!yscreen_size + 256) and $
    is_idl() then begin
    slide_image, ima, xvis=512, yvis=512
endif else begin
    win = get_win_no()
    window, /free, xsize=cols,$
        ysize=rows, $
        xpos=xpos, $
        ypos=ypos, $
        colors=nc, title = string(win) + ' : RGB_TV'

    tvcrs,cols/2,rows/2,/dev
    tv, ima
endelse

if disp then print, format = '(f7.3,a6)', t1-t0," [sec]"

if muell_window then wdelete, m_win

;-----
; if created image should be kept

```

```
if out_wanted then begin
  if disp then print, ' storing (pure SRF) to ', output_file
  write_srf_vs, output_file, rotate(ima,7), r,g,b
endif
```

```
;-----
; return & end
```

```
rtcode = 0
```

```
return
end
```
