
Subject: Re: Targa (TGA) reader
Posted by [knipp](#) on Thu, 06 Jun 1996 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

In article 4A3@psicorp.com, Phil Mulhall <mulhall@psicorp.com> writes:
> IDL users,
>
> Has anyone seen an IDL procedure to read/convert TARGA (.TGA) files? I
> have a bunch of image files I would like to process and I do not want to
> have to convert them using HiJaak. I have checked with RSI, Ray
> Sterner, and a couple of other sites on the WEB without any luck.
>
> Thanks
> Phil Mulhall
> Physical Sciences Inc.

Hi Phil

I've written a procedure some years ago (same problem), and I can't remember how well it was working.

Anyway, here's the code ... work with it.

Karl

P.S. Sorry for the unusual documentation-header
Subroutines: chk_path adds IDL-path to filename
exist_f checks file-existence
rm_file removes file
wr_srf_h write SunRaster header

If you're in need of one I'll be happy to send it to you ...

----- snip -----

```
;+
pro tga_to_srf, input_file, output_file, trans=trans, rot_ima=rot_ima, $
    rtcode=rtcode

; converts images in TARGA format to SunRaster
; -----
; START OF DESCRIPTION
;
; subroutine      IPI, U Hannover 07'93, modified 07'93
;                  03'95: INT(LONG)_REVERSE > BYTEORDER
;
```

```

; METHOD:      converts images in TARGA format to SunRaster
;      assume header of 18 bytes,
;      with 9 entries of 2byte each:
;      # 7: number of columns
;      # 8: number of rows
;      # 9: number of bits - 8: grey, 24: color
;
;      color: read/write as bytarr (3,col, rows)
;      write with no color-tables
;
;      grey: read/write as bytarr (col, rows)
;      read no color-table, assume standard
;      write with standard color-tables
;
; INPUT PARAMETER: input_file  targa-image
;      output_file  SRF, type 1, 8/24 bit
;
; OUTPUT PARAMETER: none
;
; INPUT KEYWORDS : trans      grey-scaling  0: no scaling      [DEF]
;                  1: linear "
;                  2: histogram based "
;      rot_ima  rotate-control (ROTATE,i) [DEF: 0, no rot.]
; OUTPUT KEYWORDS : rtcode    Return-code : 0 for o.k., -1 for ERROR
;
; EXAMPLE:
;
; print, 'NO EXAMPLE'
;
; END  OF DESCRIPTION
; -----
;-

```

```

rtcode = -1 ; Return - code
message, ' INT(LONG)_REVERSE >> BYTEORDER: modification NOT CHECKED !!' + $
string(7b), /info

```

```

; -----
; test input

message, ' testing ...', /info

if n_params() ne 2 then begin
  print,"The number of parameters was wrong:",n_params()
  doc_library,"tga_to_srf"
  return
endif

```

```

if n_elements(input_file) eq 0 then begin
    print,string(7b)
    message, 'INPUTFILE UNDEFINED : ', /info
    return
endif
chk_path, input_file
found = findfile(input_file)
if found(0) eq "" then begin
    print,string(7b)
    message, 'FILE NOT FOUND : '+input_file, /info
    return
endif

if exist_f(output_file) then rm_file, output_file, /confirm
if exist_f(output_file) then begin
    print,string(7b)
    message, 'start again with new name for SRF-file ..', /info
    return
endif

; -----
; test keywords, pre - definitions

if n_elements(trans) eq 0 $
    then trans = 0 $
    else trans = long(trans(0))
case trans of
    0: s_trans = 'no grey-scaling at all'
    1: s_trans = 'linear grey-scaling'
    2: s_trans = 'histogram based grey-scaling'
    else: begin
        print,string(7b)
        message, 'un-supported code for TRANS: ' + string(trans),/info
        return
    end
endcase

if n_elements(rot_ima) eq 0 $
    then rot_ima = 0 $
    else rot_ima = long(rot_ima(0)) < 71

header = intarr(9) ; header
offset = 181 ; header-size

; -----

```

```

; open file

message, 'reading header ...', /info

openr, uni, input_file, /get_lun, error=io_err

if io_err ne 0 then begin
  print, string(7b)
  message, 'ERROR opening file: ' + input_file, /info
  return
endif

; -----
; read header

readu, uni, header

; -----
; test on targa

if header(8) ne 8 and header(8) ne 24 then begin
  byteorder, header, /SSWAP & header = int_reverse(header)
  if header(8) ne 8 and header(8) ne 24 then begin
    print, string(7b)
    message, 'file not recognized as targa : ' + input_file, /info
    free_lun, uni
    return
  endif
  message, ' byte swapped ..', /info
endif

; -----
; dimensions, define image-array, read (?mem-size)

cols = header(6) & s_cols = strtrim(string(cols),2)
rows = header(7) & s_rows = strtrim(string(rows),2)

case header(8) of
8: begin
  s_grey = 'b&w - image'
  i      = assoc(uni,bytarr(cols,rows),offset)
end
24: begin
  s_grey = 'color - image'
  i      = assoc(uni,bytarr(3,cols,rows),offset)

```

```

end
else: begin
  print,string(7b)
  message, 'un-supported grey-code: ' + string(header(8)),/info
  message, 'expected 8 or 24 ..',/info
  free_lun, uni
  return
end
endcase

```

```

; -----
; ? scaling, close

```

```

message, s_trans, /info

```

```

case trans of
0: ima = i(0)
1: ima = bytscl(i(0), top=255)
2: ima = hist_equal(i(0), top=255)
end

```

```

free_lun, uni
i = 0

```

```

s_ima = size(ima)
length = s_ima ( n_elements(s_ima)-1 )

```

```

; -----
; open outfile

```

```

if rot_ima ne 0 $
  then message, 'storing/rotating ..', /info $
  else message, 'storing ..', /info

```

```

openw, uno, output_file, /get_lun, error = io_err

```

```

if io_err ne 0 then begin
  print,string(7b)
  message, 'ERROR opening file: ' + output_file, /info
  return
endif

```

```

; -----
; store

```

```

case header(8) of

```

```

8: begin
ct = bindgen(256)
wr_srf_h, output_file=output_file, width=cols, height=rows, $
  depth=header(8), length=length, type=1, maptype=1, $
  maplength=768, rr=ct, gg=ct, bb=ct, /update
o = assoc(uno, bytarr(cols,rows), 800)
if rot_ima ne 0 $
  then o(0) = rotate(ima,rot_ima) $
  else o(0) = ima
end

24: begin
wr_srf_h, output_file=output_file, width=cols, height=rows, $
  depth=header(8), length=length, type=1, maptype=0, $
  maplength=0, /update
o = assoc(uno, bytarr(3,cols,rows), 32)
if rot_ima ne 0 then begin
  tmp = reform(ima(0,*,*), cols, rows)
  ima(0,**) = rotate(tmp, rot_ima)
  tmp = reform(ima(1,*,*), cols, rows)
  ima(1,**) = rotate(tmp, rot_ima)
  tmp = reform(ima(2,*,*), cols, rows)
  ima(2,**) = rotate(tmp, rot_ima)
endif
  o(0) = ima
end

endcase

; -----
; close

free_lun, uno
o = 0

print
print, ' Conversion: from TGA(targa)- to SRF (SunRaster)- format: '
print, ' ====='
print
print, ' input_file:      ', input_file
print, ' output_file:      ', output_file
print, ' number of columns:   ', s_cols
print, ' number of rows:     ', s_rows
print, ' b&w/color:          ', s_grey
print

```

```
if rot_ima ne 0 $  
    then print, 'WARNING: image rotated with code: ', rot_ima  
print
```

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

```
rcode = 0
```

```
return  
end
```

```
----- snap -----
```
