
Subject: DIS and DISMENU image display programs: 4/4

Posted by [grunes](#) on Mon, 19 May 1997 07:00:00 GMT

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

DIS and DISMENU IDL/PV-WAVE image display programs: 4/4

```
-----CUT HERE----rdsdf.pro-----
; -----RdSDF-----
pro RdSDF,filename,a,name,number=number; Read array from HDF file in a
    ; scientific data format.
    ; -----INPUT-----
    ; filename=Name of file to read.
    ; name (optional) specifies the array
    ; name to read.
    ; Number (optional)=0-origin sequence
    ; number of variable in file.
    ; -----OUTPUT-----
    ; a=array to read from file
    ; name=name of variable.
    ; number=sequence number of variable.
;Written by Mitchell R Grunes.

    if n_elements(number) eq 0 then number=0
loop:
    if !prompt eq 'IDL> ' then begin
        sd_id=HDF_SD_Start(filename,/read)
        sds_id=hdf_sd_select(sd_id,number)

        HDF_SD_GetInfo,sds_id,name=name2
        print,'Sequence number=',number,' Variable Name=',name2
        if n_elements(name) gt 0 then begin
            if name ne name2 then begin
                number=number+1
                print,'Trying next sequence number.'
                HDF_SD_EndAccess,sds_id
                goto,loop
            endif
        endif
        HDF_SD_GetData,sds_id,a

        HDF_SD_EndAccess,sds_id
        HDF_SD_End,sd_id
    endif else begin
        stop,'My RdSDF stuff only works from IDL, not PV-WAVE.
    endelse
end
-----CUT HERE-----
```

```

-----CUT HERE----read_pgm.pro-----
;-----
; TABLE OF CONTENTS
; pro ReadFileInt,unit,n          ; Read ASCII # from PGM file.
; pro read_pgm,FilNam,a          ; Read raw PGM file.
;-----
; pro ReadFileInt,unit,n          ; Read ASCII # from PGM file.
;Written by Mitchell R Grunes.
start:
  n=0
  a=' '
  readu,unit,a
  if a eq '#' then begin          ; Skip comment
    readf,1,a
    goto,start
  endif
  if a le ' ' then goto,start      ; Skip leading white space
  while a ge '0' and a le '9' do begin
    n=n*10+long(a)
    readu,unit,a
  endwhile
end
;-----
; pro read_pgm,FilNam,a          ; Read raw PGM file.
;Written by Mitchell R Grunes.
openr,unit,FilNam,/get_lun
a=' '
readu,unit,a
if a ne 'P5' then print,'ERROR--Invalid PGM File'
ReadFileInt,unit,ncol
ReadFileInt,unit,nrow
ReadFileInt,unit,maxa
if maxa le 255 then a=bytarr(ncol,nrow)
if maxa gt 255 then a=intarr(ncol,nrow)
readu,1,a
if maxa gt 255 then begin
  print,'Enter 0 if this is a LSB-1st machine (PC).'
  print,'Enter 1 if this is a MSB-1st machine (Sun, SGI):'
  ReadVar,i
  if i eq 0 then byteorder,a,/sswap
endif
free_lun,unit
end
-----CUT HERE-----

-----CUT HERE----read_sgi_image.pro-----
;-----read_sgi_image-----
; pro read_sgi_image,filename,image ; Read an SGI format image file.

```

```

; -----INPUT-----
; filename=Name of file.
; -----OUTPUT-----
; image= 2 or 3 dimensional image.
; Monochrome (*.bw) images are 2D.
; RGB images (*.rgb) are 3D--3rd
; dimension has 3 planes, or 4
; (*.rgba) to include an "alpha"
; channel. If you want to
; understand alpha channels log onto
; an sgi and type
; man multisample
; By Mitchell R Grunes.
; We do NOT use run length encoding (RLE), or anything fancy.
; We also assume this is being run a MSB-first machine, such as a Sun Sparc
; or SGI workstation.
; That means it would fail on a PC-compatible or most HP or DEC machines.
openr,unit,filename,/get_lun
idummy=0
readu,unit,idummy ; Iris image file magic number
if idummy ne 474 then stop,'***read_sgi_image encountered bad magic number.'
idummy=byte(0)
readu,unit,idummy
if idummy ne 0 then stop,'***read_sgi_image can not handle RLE.'
bpp=byte(0) & nDim=0 & nCol=0 & nRow=0 & nChan=0
readu,unit,bpp ; Bytes/pixel
if bpp ne 1 and bpp ne 2 and bpp ne 4 then $
stop,'ERROR: Wrong bytes/pixel in read_sgi_image!'
readu,unit,nDim ; # of dimensions
if nDim lt 2 or nDim gt 3 then $
stop,'ERROR: Wrong # of dimensions in read_sgi_image!'
readu,unit,nCol ; # of columns
readu,unit,nRow ; # of rows
readu,unit,nChan ; # of channels
if nDim eq 2 then begin
if bpp eq 1 then begin
image=bytarr(nCol,nRow)
endif else if bpp eq 2 then begin
image=intarr(nCol,nRow)
endif else if bpp eq 4 then begin
image=lonarr(nCol,nRow)
endif
endif else if nDim eq 3 then begin
if bpp eq 1 then begin
image=bytarr(nCol,nRow,nChan)
endif else if bpp eq 2 then begin
image=intarr(nCol,nRow,nChan)
endif else if bpp eq 4 then begin

```

```

    image=lonarr(nCol,nRow,nChan)
endif
endif
iDummy=0L
readu,unit,iDummy          ; Lo scaling value--ignored
readu,unit,iDummy          ; Hi scaling value--ignored
readu,unit,iDummy          ; Dummy value
dummy=bytarr(80)
readu,unit,dummy           ; Image name--not used
readu,unit,idummy          ; Colormap ID--not used
dummy=bytarr(404)
readu,unit,dummy           ; Dummy values.
readu,unit,image           ; The image
free_lun,unit
end
-----CUT HERE-----

-----CUT HERE----readimage.pro-----
;-----
pro ReadImage,FilNam,a,nCol,nRow,iCol1,iCol2,iRow1,iRow2, $
  nheader,nFrame,iType=iType      ; Read image--many formats.
                                   ; image=name of image file.
                                   ; a=Array to read image to.
                                   ; nCol=# of columns in image.
                                   ; Used only for raw formats.
                                   ; nRow=# of rows in image.
                                   ; Used only for raw formats.
                                   ; iCol1,iCol2=0-origin Col #'s to
                                   ; load.
                                   ; iRow1,iRow2=0-origin Row #'s to
                                   ; load.
                                   ; nHeader=# of header bytes or
                                   ; lines to skip. Used only for
                                   ; raw formats.
                                   ; nFrame=# of movie frames stored
                                   ; in file. Used only for raw
                                   ; formats.
                                   ; iType=Image file type--
                                   ; see printimtype.pro.
;Written by Mitchell R Grunes.
if n_elements(nCol) eq 0 then nCol=0
if n_elements(nRow) eq 0 then nRow=0
if n_elements(iCol1) eq 0 then iCol1=0
if n_elements(iCol2) eq 0 then iCol2=0
if n_elements(iRow1) eq 0 then iRow1=0
if n_elements(iRow2) eq 0 then iRow2=0
if n_elements(iType) eq 0 then iType=1
if n_elements(nHeader) eq 0 then nHeader=0

```

```

if n_elements(nFrame) eq 0 then nFrame=1
Start:
if iType lt 20 and iType ge 0 then begin
  iCol1=0 > iCol1
  if iCol2 le iCol1 or iCol2 gt nCol-1 then iCol2=nCol-1
  iRow1=0 > iRow1
  if iRow2 le iRow1 or iRow2 gt nRow-1 then iRow2=nRow-1
  print,'Reading image ',FilNam,' cols',iCol1,iCol2,' rows',iRow1,iRow2
endif else begin
  print,'Reading image ',FilNam
endelse
if iType eq -1 then begin
  a=byte(255)-byte(indgen(16,16))      ;multi-row Gray scale
  for i=1,15,2 do a(*,i)=a(15-indgen(16),i)
  a=rebin(a,32,32)
  for i=1,31,4 do a(0:29,i)=0
  for i=3,31,4 do a(2:31,i)=0
  a=a(*,0:30)
  nCol=256
  nRow=256
endif else if iType lt 20 then begin
  if iCol1 lt 0 then iCol1=0
  if iCol2 le iCol1 or iCol2 gt nCol-1 then iCol2=nCol-1
  if iRow1 lt 0 then iRow1=0
  if iRow2 le iRow1 or iRow2 gt nRow-1 then iRow2=nRow-1
  openr,unit,FilNam,/get_lun
  nBytePix=1
  if iType eq 2 or iType eq 3 or iType eq 9 or iType eq 10 then $
    nBytePix=2
  if iType eq 4 or iType eq 5 or iType eq 11 or iType eq 12 then $
    nBytePix=4
  if iType eq 6 or iType eq 7 or iType eq 13 or iType eq 14 then $
    nBytePix=8
  if iType eq 8 or iType eq 15 then nBytePix=16
  if iType eq 27 or iType eq 28 then nBytePix=2
  iskip=iRow1
  width=nCol
  keep=iRow2-iRow1+1
  if iType ge 16 and iType le 19 then begin
    a=' '
    if nheader gt 0 then for i=1,nheader do readf,unit,a
    if iType eq 16 then begin
      if nFrame eq 1 then a=intarr(width,keep)
      if nFrame gt 1 then a=intarr(width,keep,nFrame)
    endif else if iType eq 17 then begin
      if nFrame eq 1 then a=lonarr(width,keep)
      if nFrame gt 1 then a=lonarr(width,keep,nFrame)
    endif else if iType eq 18 then begin

```

```

    if nFrame eq 1 then a=fltarr(width,keep)
    if nFrame gt 1 then a=fltarr(width,keep,nFrame)
endif else if iType eq 19 then begin
    if nFrame eq 1 then a=dblarr(width,keep)
    if nFrame gt 1 then a=dblarr(width,keep,nFrame)
endif
endif else begin
    print,'skipping ', nheader+iskip*long(width)*nBytePix
    point_lun, unit, nheader+iskip*long(width)*nBytePix
endelse
if iType eq 0 or iType eq 1 then begin
    if nFrame eq 1 then a=bytarr(width,keep)
    if nFrame gt 1 then a=bytarr(width,keep,nFrame)
    jtype=1          ; Will be stored in byte &&&
endif else if iType eq 2 or iType eq 3 or iType eq 9 or iType eq 10 $
or iType eq 16 then begin
    if nFrame eq 1 then a=intarr(width,keep)
    if nFrame gt 1 then a=intarr(width,keep,nFrame)
    jtype=2          ; Will be stored in int
    if iType eq 3 or iType eq 10 then jtype=3
                        ; unsigned will be converted to long
endif else if iType eq 4 or iType eq 11 or iType eq 17 then begin
    if nFrame eq 1 then a=lonarr(width,keep)
    if nFrame gt 1 then a=lonarr(width,keep,nFrame)
    jtype=3          ; Will be stored in long
endif else if iType eq 5 or iType eq 12 or iType eq 18 then begin
    if nFrame eq 1 then a=fltarr(width,keep)
    if nFrame gt 1 then a=fltarr(width,keep,nFrame)
    jtype=4          ; Will be stored in float
endif else if iType eq 6 or iType eq 13 or iType eq 19 then begin
    if nFrame eq 1 then a=dblarr(width,keep)
    if nFrame gt 1 then a=dblarr(width,keep,nFrame)
    jtype=5          ; Will be stored in double
endif else if iType eq 7 or iType eq 14 then begin
    if nFrame eq 1 then a=complexarr(width,keep)
    if nFrame gt 1 then a=complexarr(width,keep,nFrame)
    jtype=6          ; Will be stored in complex
endif
; &&& Type 8 and 15--not sure how to do
if iType ge 16 and iType le 19 then readf,unit,a
if iType ge 0 and iType lt 16 then readu,unit,a
print,"File read complete"
free_lun,unit
; if iGeom eq 0 or iGeom eq 2 or iGeom eq 5 or iGeom eq 7 then $
if nFrame eq 1 then a=a(iCol1:iCol2,0:keep-1)
if nFrame gt 1 then a=a(iCol1:iCol2,0:keep-1,*)
; if iGeom eq 1 or iGeom eq 3 or iGeom eq 4 or iGeom eq 6 then $
; a=a(iRow1:iRow2,0:keep-1)

```

```

if iType eq 9 or iType eq 10 then byteorder,a,/sswap
if iType eq 11 or iType eq 12 then byteorder,a,/lswap
; &&&Type 13 and 15--not sure how to do
        ; Unsigned--clip hi bits
if iType eq 2 or iType eq 3 or iType eq 10 then begin
    if double(min(a)) lt 0 then begin
        if iType eq 1 then a=a and 255
        if iType eq 3 or iType eq 10 then begin
            if !prompt eq 'IDL> ' then begin
                a=temporary(a) and 65535
            endif else begin
                a=a and 65535
            endif
        endif
    endif
endif
endif

; if jtype eq 4 then begin
;   if min(a eq long(a)) eq 1 then begin
;     a=long(a)
;     jtype=3
;   endif
; endif
if jtype eq 2 or jtype eq 3 then maxa=max(a,min=mina)
if jtype eq 3 then begin
    if mina ge -32768 and maxa le 32767 then begin
        a=fix(a)
        jtype=2
    endif
endif
if jtype eq 2 then begin
    if mina ge 0 and maxa le 255 then a=byte(a)
endif
endif else begin        ; Types containing size and shape...
    if iType eq 20 then begin
        read_gif,FilNam,a,r,g,b
    endif else if iType eq 21 then begin
        READ_PICT,FilNam,a,r,g,b
    endif else if iType eq 22 then begin
        READ_SRF,FilNam,a,R,G,B
    endif else if iType eq 23 then begin
        ; Not yet understood function description--probably won't work
        read_wave,FilNam,a,names,dimensions
    endif else if iType eq 24 then begin
        READ_X11_BITMAP,FilNam,a
    endif else if iType eq 25 then begin
        a=READ_XWD(FilNam,r,g,b)
    endif else if iType eq 26 then begin

```

```

a=reverse(TIFF_READ(FilNam),2)
if RankAr(a) eq 3 and (size(a))(1) eq 3 then begin
    ; Can't really handle RGB images
    ; yet.
    s=size(a) ; For now, change dimension
    c=bytarr(s(2),s(3),3) ; order, make into a movie.
    for i=0,2 do c(*,*,i)=reform(a(i,*,*))
    a=c
    c=0 ;(to save memory)
endif
endif else if iType eq 27 or iType eq 28 then begin ;SOHO Compressed
    print,'The free trial copy of IDL cannot do this, and will give ', $
    'warnings.'
    if FilNam ne 'test.cim' then spawn,'cp '+FilNam+' test.cim'
    spawn,'rm test.rim test.siz'
    if iType eq 27 then spawn,'sohorecon test.cim test.rim'
    if iType eq 28 then spawn,'sohorecon test.cim test.rim skip'
    openr,unit,'test.siz',/get_lun
    readf,unit,nCol,nRow,iType
    FilNam='test.rim'
    iType=2
    goto,Start
endif else if iType eq 29 then begin ;PGM
    read_pgm,FilNam,a
endif else if iType eq 30 then begin;Tom Ainsworth's read_array
    read_array,FilNam,a ; format.
endif else if iType eq 31 or iType eq 32 then begin
    rd_image,FilNam,a,imginfo,nimgs ; Bob Jansen's routines to read
    ; Sealab and HDF files.
endif else if iType eq 33 or iType eq 34 then begin; Iris RGB Image
    read_sgi_image,FilNam,a
endif else if iType eq 36 then begin; DRM CEOS format
    print,'Extracting data from CEOS header
    close,1
    openr,1,FilNam
    temp=' ' ; 8 spaces
    point_lun,1,19498 ; location for # of rows
    readu,1,temp
    nCol=long(temp)+192/2 ; (Normal CEOS record header=192
    ; bytes.
    point_lun,1,19486 ; location for # of columns.
    readu,1,temp
    nRow=long(temp)
    jcol1=96
    jcol2=nCol-1
    jrow1=1
    jrow2=nRow-1
    nRow3=nRow<500 ; Test portion

```



```

a=intarr(nCol,nRow3)          ; It appears that image has
                                ; zeroed cols and and rows
ByteSkip=19250                ; at border, not documented in
point_lun,1,ByteSkip          ; header.
readu,1,a
while jcol1 lt jcol2-2 and total(a(jcol1:jcol1,30:nRow3-30) eq 0) $
  gt (nRow3-60)/2 do jcol1=jcol1+1
while jcol2 gt jcol1+2 and total(a(jcol2:jcol2,30:nRow3-30) eq 0) $
  gt (nRow3-60)/2 do jcol2=jcol2-1
while jrow1 lt nRow3-2 and total(a(jcol1+30:jcol2-30,jrow1) eq $
  0) gt (nCol-60)/2 do jrow1=jrow1+1
PixelType=2                    ;(ERS-1 PixelType=2)
point_lun,1,ByteSkip+(nRow-1-nRow3)*nCol*abs(PixelType)
readu,1,a
while jrow2 gt nrow-(nRow3-2) and total(a(jcol1+30:jcol2-30, $
  nRow3-(nrow-jrow2)) eq 0) gt (nCol-60)/2 do jrow2=jrow2-1
nCol2=jCol2-jCol1+1
nRow2=jRow2-jRow1+1
a=intarr(nCol,nRow2)
point_lun,1,ByteSkip+jRow1*long(nCol)*2
readu,1,a
a=a(jCol1:jCol2,*)
close,1
endif else if iType eq 37 then begin; DRM CEOS format
  RdSDF,FilNam,a,name          ; Read array from HDF file in a
                                ; scientific data format.

endif
s=size(a)
nCol=s(1)
nRow=s(2)
if iCol1 lt 0 then iCol1=0
if iCol2 le iCol1 or iCol2 gt nCol-1 then iCol2=nCol-1
if iRow1 lt 0 then iRow1=0
if iRow2 le iRow1 or iRow2 gt nRow-1 then iRow2=nRow-1
print,'Selecting cols',iCol1,iCol2,' rows',iRow1,iRow2
GetSize,a, nCol3,nRow3
if iCol1 ne 0 or iRow1 ne 0 or iCol2 ne nCol3-1 or iRow2 ne nRow3-1 $
  then a=a(iCol1:iCol2,iRow1:iRow2)
endelse
;if iGeom ne 0 then begin          ; Done elsewhere
; if !prompt eq 'IDL> ' then begin
;   a=rotate(temporary(a),iGeom)
; endif else begin
;   a=rotate(a,iGeom)
; endelse
;endif
end
-----CUT HERE-----

```

```

-----CUT HERE----sq.pro-----
;
function sq,a          ; Format number, squeeze blanks.
                        ; (remove blanks).
                        ; -----INPUT-----
                        ; a=a numeric value.
                        ; -----OUTPUT-----
                        ; returns formatted string, with
                        ; no blanks.
;Written by Mitchell R Grunes.
for i=0,n_elements(a)-1 do begin
  b=a(i)
  if i eq 0 then s="" else s=s+' ' ; Blanks between numbers
  b=string(b)
                        ; Kill trailing 0's after ".".
  if strpos(b, '.') ge 0 and strpos(b, 'e') lt 0 then begin
    while strmid(b, strlen(b)-1, 1) eq '0' do begin
      b=strmid(b, 0, strlen(b)-1)
    endwhile
    if strmid(b, strlen(b)-1, 1) eq '.' then b=strmid(b, 0, strlen(b)-1)
  endif
  s=s+strcompress(b, /remove_all)
endfor
return, s
end

```

-----CUT HERE-----

```

-----CUT HERE----temporary.pro-----
;temporary.pro to use if using a version of IDL or PV-WAVE that does
; not include temporary
function temporary,a
return,a
end

```

-----CUT HERE-----

```

-----CUT HERE----usercol.pro-----
; -----USERCOL.PRO-----
pro usercol, no_window=no_window ; Create a user false color map
; by Mitchell R Grunes ATSC/NRL.
if n_elements(no_window) eq 0 then no_window=0
if no_window eq 0 then device, pseudo=8
if no_window eq 0 then window, 0, colors=256, retain=2
;device, set_mode=5

print, 'Enter 0 to modify old usercol.dat
print, 'Enter 1 to create new usercol.dat
read, choice

```

```

if choice eq 0 then begin
  openr,unit,'usercol.dat',/get_lun
  ncolorr=0 & ncolorg=0 & ncolorb=0
  readf,unit,ncolorr,ncolorg,ncolorb ; # of mark points

  xrf=intarr(ncolorr) ;independent coord of transfer function: R,G,B (0-255)
  xgf=intarr(ncolorg)
  xbf=intarr(ncolorb)

  rf=xrf & gf=xgf & bf=xbf ;dependent coord for R,G,B
  readf,unit,xrf,xgf,xbf,rf,gf,bf
  free_lun,unit
endif else begin
  ncolorr=7 & ncolorg=7 & ncolorb=7
  ; independent variable.
  xrf=[0,42.5,110,127.5,145,212.5,255] & xgf=xrf & xbf=xrf
  rf=[0, 0, 0, 0,255,255,255] ; dependent variable:
  gf=[0, 0,255,255,255, 0,255] ; black,blue,cyan,green,yellow,red,
  bf=[0,255,255, 0, 0, 0,255] ; white
endelse
loop:
  print,'red pixel:',xrf
  print,' becomes:',rf
  print,'grn pixel:',xgf
  print,' becomes:',gf
  print,'blu pixel:',xbf
  print,' becomes:',bf

  r=interpol(float(rf),xrf,float(indgen(256)))
  g=interpol(float(gf),xgf,float(indgen(256)))
  b=interpol(float(bf),xbf,float(indgen(256)))

  scale=tan(1) ; Try to compensate for saturation,round.
  r=fix( (scale+tan((r-127.5)/127.5)) * (255/scale/2) +.5)
  g=fix( (scale+tan((g-127.5)/127.5)) * (255/scale/2) +.5)
  b=fix( (scale+tan((b-127.5)/127.5)) * (255/scale/2) +.5)

  i=fix(indgen(251)*255./250+.5) ; We shall temporarily reserve
  ; 0 for black, 252-255 for
  ; white,red,green,blue

  rr=[0,r(i),255,255,0,0]
  gg=[0,g(i),255,0,255,0]
  bb=[0,b(i),255,0,0,255]
  COMMON colors,ro,go,bo,rc,gc,bc
  rc=rr & gc=gg & bc=bb
  ro=rr & go=gg & bo=bb
  tvlct,rr,gg,bb

  ; Draw transfer functions.

```

```

plot,[0,255],[0,300],color=252,/nodata,xstyle=1,ystyle=1
oplot,xrf,rf,color=253           ; Draw transfer functions
oplot,xgf,gf,color=254
oplot,xbf,bf,color=255
plots,xrf,rf,color=253,psym=4
plots,xgf,gf,color=254,psym=5
plots,xbf,bf,color=255,psym=6

for color=0,255 do begin           ; fill in false color scale
  for t=-.5,.51,.25 do $
    plots,[color,color]+t,[265,260],color=fix((color*250./255+1.5) )
endfor

print,'---Menu---'
print,'1 Adjust a red mark'
print,'2 Adjust a green mark'
print,'3 Adjust a blue mark'
print,'4 Add a red mark'
print,'5 Add a green mark'
print,'6 Add a blue mark'
print,'7 Delete a red mark'
print,'8 Delete a green mark'
print,'9 Delete a blue mark'
print,'0 to exit and save to file'
read,choice
if choice eq 0 then begin
  openw,unit,'usercol.dat',/get_lun
  printf,unit,ncolorr,ncolorg,ncolorb
  printf,unit,"
  printf,unit,xrf
  printf,unit,xgf
  printf,unit,xbf
  printf,unit,"
  printf,unit,rf
  printf,unit,gf
  printf,unit,bf
  free_lun,unit
  return
endif

tvcrs,1
if (choice ge 1 and choice le 3) or (choice ge 7 and choice le 9) then begin
  if (choice eq 7 and ncolorr le 1) or (choice eq 8 and ncolorg le 1) or $
    (choice eq 9 and ncolorb le 1) then goto,loop
  if choice le 3 then print,'Click mouse on mark you wish to adjust'
  if choice gt 3 then print,'Click mouse on mark you wish to delete'
  cursor,x,y
  !err=1 & while !err ne 0 do cursor,xx,yy,/device,/nowait

```

```

x=0 > x < 255
y=0 > y < 255
index=0
if choice eq 1 or choice eq 7 then begin
  for j=1,ncolorr-1 do begin
    if (xrf(j)-x)^2+(rf(j)-y)^2 lt (xrf(index)-x)^2+(rf(index)-y)^2 then $
      index=j
  endfor
  if choice eq 1 then print,'replacing ',xrf(index),rf(index)
  if choice eq 7 then begin
    if index eq 0 or index eq ncolorr-1 then goto,loop
    i=where(indgen(ncolorr) ne index)
    xrf=xrf(i)
    rf=rf(i)
    ncolorr=ncolorr-1
    goto,loop
  endif
endif else if choice eq 2 or choice eq 8 then begin
  for j=1,ncolorg-1 do begin
    if (xgf(j)-x)^2+(gf(j)-y)^2 lt (xgf(index)-x)^2+(gf(index)-y)^2 then $
      index=j
  endfor
  if choice eq 2 then print,'replacing ',xgf(index),gf(index)
  if choice eq 8 then begin
    if index eq 0 or index eq ncolorg-1 then goto,loop
    i=where(indgen(ncolorg) ne index)
    xgf=xgf(i)
    gf=gf(i)
    ncolorg=ncolorg-1
    goto,loop
  endif
endif else if choice eq 3 or choice eq 9 then begin
  for j=1,ncolorb-1 do begin
    if (xbf(j)-x)^2+(bf(j)-y)^2 lt (xbf(index)-x)^2+(bf(index)-y)^2 then $
      index=j
  endfor
  if choice eq 3 then print,'replacing ',xbf(index),bf(index)
  if choice eq 9 then begin
    if index eq 0 or index eq ncolorb-1 then goto,loop
    i=where(indgen(ncolorb) ne index)
    xbf=xbf(i)
    bf=bf(i)
    ncolorb=ncolorb-1
    goto,loop
  endif
endif
print,'Click mouse where you wish to move it to.

```

```

cursor,x2,y2
!err=1 & while !err ne 0 do cursor,xx,yy,/device,/nowait
x2=0 > x2 < 255
y2=0 > y2 < 255
if index eq 0 then x2=0
if index eq ([ncolorr,ncolorg,ncolorb])(choice-1)-1 then x2=255
if choice eq 1 then begin
  xrf(index)=x2
  rf(index)=y2
endif else if choice eq 2 then begin
  xgf(index)=x2
  gf(index)=y2
endif else if choice eq 3 then begin
  xbf(index)=x2
  bf(index)=y2
endif
endif else if choice ge 4 and choice le 6 then begin
  print,'Click mouse near new point.
  cursor,x,y
  !err=1 & while !err ne 0 do cursor,xx,yy,/device,/nowait
  x=1 > x < 254
  y=1 > y < 254
  if choice eq 4 then begin
    ncolorr=ncolorr+1
    xrf=[xrf,x]
    rf=[rf,y]
  endif else if choice eq 5 then begin
    ncolorg=ncolorg+1
    xgf=[xgf,x]
    gf=[gf,y]
  endif else if choice eq 6 then begin
    ncolorb=ncolorb+1
    xbf=[xbf,x]
    bf=[bf,y]
  endif
endif
goto,loop
end
;-----
usercol
end
-----CUT HERE-----

-----CUT HERE-----vartyp.pro-----
;-----
function VarTyp,a          ; Variable Type of a:
                          ; 0=undefined
                          ; 1=byte

```

```

; 2=integer*2
; 3=integer*4
; 4=real*4
; 5=real*8
; 6=complex
; 7=string
;Written by Mitchell R Grunes.
s=size(a)
return,s(n_elements(s)-2)
end
-----CUT HERE-----

-----CUT HERE----wrfil.pro-----
; -----WrFil-----
pro WrFil,FilNam,a ; Write a raw to file FilNam.
; By Mitchell R Grunes.
print,'-----'
print,'Writing file ',FilNam
print,'Shape given by:'
if !prompt eq 'IDL> ' then help,a
if !prompt ne 'IDL> ' then info,a ; Lately, Wave replaced
openw,unit,FilNam,/get_lun
writeu,unit,a
free_lun,unit

; 'help' with 'info'.
print,'-----'
end
-----CUT HERE-----

-----CUT HERE----write_pgm.pro-----
; -----
pro write_pgm,FilNam,a ; Write raw PGM file--not tested yet.
;Written by Mitchell R Grunes.
GetSize,a, nCol,nRow
openw,unit,FilNam,/get_lun
writeu,unit,'P5'
writeu,unit,sq(ncol)+' '
writeu,unit,sq(nrow)+' '
vt=VarTyp(a)
if vt eq 1 then maxa=255
if vt eq 2 then maxa=65535
if vt ne 1 and vt ne 2 then stop,'Wrong variable type for write_pgm'
writeu,unit,sq(maxa)+' '
if maxa gt 255 then begin
print,'Enter 0 if this is a LSB-1st machine (PC).'
print,'Enter 1 if this is a MSB-1st machine (Sun, SGI):'
ReadVar,i
if i eq 0 then byteorder,a,/sswap

```

```

endif
writeu,1,a
free_lun,unit
end
-----CUT HERE-----

-----CUT HERE----write_sgi_image.pro-----
;-----write_sgi_image-----
pro write_sgi_image,filename,image,pseudo=pseudo,scale=scale
    ; Write an SGI format image file.
    ; -----INPUT-----
    ; filename=Name of file.
    ; It is suggested that
    ;   monochrome file names end in .bw
    ;   RGB      file names end in .rgb
    ;   RGB+alpha file names end in .rgba
    ; image= 2 or 3 dimensional image.
    ; Monochrome images are 2D.
    ; RGB images are 3D--3rd dimension
    ; has 3 planes, or 4 to include an
    ; "alpha" channel. If you want to
    ; understand alpha channels log onto
    ; an sgi and type
    ;   man multisample
    ; Must be of type byte or int--but is
    ; handled as unsigned.
    ; /pseudo: Indicates that monochrome
    ; images (i.e., 2 dimensional) should
    ; be mapped to an RGB mage using the
    ; pseudo-colors of the current window
    ; /scale: Indicates that the minimum
    ; and maximum should be used by be
    ; used by SGI utilities to scale the
    ; image. If not present, 0 and 255
    ; will be used.
    ; -----OUTPUT-----
    ; image will be modified if /pseudo
    ; is selected.

; By Mitchell R Grunes.
; We do NOT use run length encoding (RLE).
; We also assume this is being run a MSB-first machine, such as a Sun Sparc
; or SGI workstation.
; That means it would fail on a PC-compatible or most HP or DEC machines.
if n_elements(scale) eq 0 then scale=0
if n_elements(pseudo) eq 0 then pseudo=0
openw,unit,filename,/get_lun
writeu,unit,474          ; Iris image file magic number
writeu,unit,byte(0)      ; No RLE

```



```

s=size(image)
nDim=s(0)                ; # of dimensions.
                        ; Monochrome images are two
                        ; dimensional.
                        ; RGB images have a third dimension.
if pseudo ne 0 and nDim eq 2 then begin
  tvlct,/get,r,g,b
  if total(r ne g)+total(r ne b) ne 0 then $
    image=[[[r(image)]],[[g(image)]],[[b(image)]]]
  s=size(image)
  nDim=s(0)                ; # of dimensions.
endif
if nDim lt 2 or nDim gt 3 then $
  stop,'ERROR: Wrong # of dimensions in write_sgi_image!'
nCol=s(1)                  ; # of columns of pixels
nRow=s(2)                  ; # of rows of pixels
if nDim eq 2 then nChan=1 else nChan=s(3) ; # of channels (i.e., colors)
if nDim eq 2 then VarTyp=s(4);1=byte,2=integer*2,
                        ; 3=integer*4
if VarTyp eq 1 then writeu,unit,byte(1) ; Bytes/pixel
if VarTyp eq 2 then writeu,unit,byte(2)
if VarTyp lt 1 or VarTyp gt 2 then $
  stop,'ERROR: Wrong variable type in write_sgi_image'
writeu,unit,fix(nDim)
writeu,unit,fix(nCol)
writeu,unit,fix(nRow)
writeu,unit,fix(nChan)
if Scale eq 0 then begin
  writeu,unit,long(0)        ; Minimum pixel value=0.
  writeu,unit,long(255)     ; Maximum pixel value=255.
endif else begin
  writeu,unit,long(min(image))
  writeu,unit,long(max(image))
endelse
writeu,unit,long(0)          ; Dummy value
writeu,unit,bytarr(80)       ; Image name--not used
writeu,unit,long(0)          ; Colormap ID
writeu,unit,bytarr(404)      ; Dummy values
writeu,unit,image            ; The image
free_lun,unit
end
-----CUT HERE-----

-----CUT HERE-----wrsdf.pro-----
; -----WrSDF-----
pro WrSDF,filename,a,varname,frames=frames,palette=palette
                        ; Write array to HDF file in a
                        ; scientific data format.

```

```

; (Can be read back with rdsdf.pro.)
; -----INPUT-----
; filename=Name of file to create.
; a=array to write to file.
; varname=name of variable (optional).
; if /frames is set, and the array
; is 3 dimensional (last dimension
; = # of frames), each frame will
; be written as a seperate HDF
; entry.
; (That is the usual way HDF movies
; are written.)
; if /palette is set, the current
; pallete will be added.

;Written by Mitchell R Grunes.
if n_elements(varname) eq 0 then varname='a'
if n_elements(frames) eq 0 then frames=0
if n_elements(palette) eq 0 then palette=0
if !prompt eq 'IDL> ' then begin
  sd_id=HDF_SD_Start(filename,/create)

  s=size(a)
  vt=s(n_elements(s)-2)      ; variable type
  if vt lt 1 or vt gt 5 then stop,'Bad Data type in WrHDF'
  ndim=s(0)                  ; # of dimensions
  dims=s(1:ndim)              ; dimensions
  nFrames=1                   ; # of frames
  if ndim eq 3 and frames then begin
    nframes=s(ndim)
    dims=dims(0:1)
  endif
  print,'Adding variable ',a,' to HDF file ',filename
  help,a

  if ndim eq 3 and frames then begin
    print,'Writing array as ',nframes,' frames.'
    for i=0,nframes-1 do begin
      sds_id=HDF_SD_Create(sd_id,varname,dims,byte=vt eq 1,int=vt eq 2, $
        long=vt eq 3,float=vt eq 4,double=vt eq 5)
      HDF_SD_AddData,sds_id,reform(a(*,*,i))
      HDF_SD_EndAccess,sds_id
    endfor
  endif else begin
    sds_id=HDF_SD_Create(sd_id,varname,dims,byte=vt eq 1,int=vt eq 2, $
      long=vt eq 3,float=vt eq 4,double=vt eq 5)
    HDF_SD_AddData,sds_id,a
    HDF_SD_EndAccess,sds_id
  endelse

```

```
HDF_SD_End,sd_id
if palette then begin
  tvlct,/get,R,G,B
  p=transpose([[R],[G],[B]])
  hdf_dfp_addpal,filename,p
endif
endif else begin
  stop,'My WrSDF stuff only works from IDL, not PV-WAVE.
endelse
end
```

-----CUT HERE-----

----- -
Mitchell R Grunes, grunes@imsy1.nrl.navy.mil. Opinions are mine alone.
