Subject: tiff_read
Posted by craig stevens on Thu, 10 Apr 1997 07:00:00 GMT
View Forum Message <> Reply to Message

hi

Does anyone know if there is a simple modification
to tiff_read that will read an image sequence
stored in tiff as opposed to a single image?

I have not come across such a mod although it
appears simple if one has the right tiff info.

any thoughts appreciated

Craig

--
Craig Stevens: Air-Sea Interaction
Natl. Inst. Water & Atmos. Research
po box 14-901 Kilbirnie WGTN New Zealand
fx 64 (0)4 386 2153 ph 64 (0)4 386 0476

Subject: Re: tiff_read
Posted by craig stevens on Tue, 15 Apr 1997 07:00:00 GMT
View Forum Message <> Reply to Message

FYI...to answer my own question which was....

> Does anyone know if there is a simple modification
> to tiff_read that will read an image sequence
> stored in tiff as opposed to a single image?

I have included the following butchered IDL file...
only for 8 bit images...

Incidently I couldn't really find any documentation
officially describing single-file tiff sequences...
the ones I am dealing with were written with software
from EPIX for the PIXCI framegrabber...don't know if
it'll work with any other implementation....

file is....

```
;-------------------------------------------
; NAME:
```

```
;
; tiff_seq  attempts to read in a whole bunch of images
; TIFF_seq a mod of TIFF_READ from IDL  - see original
; file for history etc
; CALLING SEQUENCE:
;   Result = TIFF_SEQ(Filename,num, ORDER = order)
;
; INPUTS:
;    Filename: A string containing the name of file to read.
;  The default extension is ".TIF".
;    num   is number of images to get...haven't worked
;          out how to automate this
; COMMON BLOCKS:
; TIFF_COM.  Only for internal use.
;
function tiff_long,a,i,len=len ;return longword(s) from array a(i)
common tiff_com, order, ifd, count

on_error,2              ;Return to caller if an error occurs

   if n_elements(len) le 0 then len = 1
   if len gt 1 then result = long(a,i,len) $
   else result = long(a,i)
   if order then byteorder, result, /lswap
   return, result
end


function tiff_rational,a,i, len = len   ; return rational from array a(i)
common tiff_com, order, ifd, count

on_error,2              ;Return to caller if an error occurs

if n_elements(len) le 0 then len = 1
tmp = tiff_long(a, i, len = 2 * len) ;1st, cvt to longwords
if len gt 1 then begin
 subs = lindgen(len)
 rslt = float(tmp(subs*2)) / tmp(subs*2+1)
endif else rslt = float(tmp(0)) / tmp(1)
return, rslt
end

function tiff_int,a,i, len=len ;return unsigned long int from TIFF short int
common tiff_com, order, ifd, count

on_error,2              ;Return to caller if an error occurs
if n_elements(len) le 0 then len = 1
if len gt 1 then begin ;Array?
```

```
 result = fix(a,i,len)
 if order then byteorder, result, /sswap
 result = long(result)
 if min(result) lt 0 then begin ;Convert to unsigned from signed 16bit
   negs = where(result lt 0)
   result(negs) = 65535L + result(negs)
   endif
endif else begin ;Scalar
 result = fix(a,i)
 if order then byteorder, result, /sswap
 if result lt 0 then result = 65535L + result
endelse
return, result
end

function tiff_byte, a,i,len=len ;return bytes from array a(i)
common tiff_com, order, ifd, count

on_error,2              ;Return to caller if an error occurs

   if n_elements(len) le 0 then len = 1
   if len gt 1 then result = a(i:i+len-1) $
   else result = a(i)
   return, result
end

function tiff_read_field, index, tag, lun  ;Return contents of field index
; On output, tag = tiff tag index.
;
common tiff_com, order, ifd, count


on_error,2                      ;Return to caller if an error occurs
TypeLen = [0, 1, 1, 2, 4, 8] ;lengths of tiff types, 0 is null type for indexin

ent = ifd(index * 12: index * 12 + 11)  ;Extract the ifd
tag = tiff_int(ent, 0)  ;Tiff tag index
typ = tiff_int(ent, 2)  ;Tiff data type
cnt = tiff_long(ent, 4)  ;# of elements
nbytes = cnt * TypeLen(typ) ;Size of tag field
IF (nbytes GT 4) THEN BEGIN  ;value size > 4 bytes ?
        offset = tiff_long(ent, 8) ;field has offset to value location
        Point_Lun, lun, offset
        val = BytArr(nbytes)  ;buffer will hold value(s)
        Readu, lun, val
        CASE typ OF  ;Ignore bytes, as there is nothing to do
    1: i = 0  ;Dummy
          2: val = String(val)  ;tiff ascii type
```

```
          3: val = tiff_int(val,0, len = cnt)
    4: val = tiff_long(val,0, len = cnt)
          5: val = tiff_rational(val,0, len = cnt)
 ENDCASE
ENDIF ELSE BEGIN   ;Scalar...
        CASE typ OF
    1: val = ent(8)
      2: val = string(ent(8:8+(cnt>1)-1))
    3: val = tiff_int(ent,8)
    4: val = tiff_long(ent,8)
        ENDCASE
     ENDELSE
return, val
end


 ;############################################################## #
;################
;################   HERE WE ARE
;################
 ;############################################################## #

function tiff_seq, file, n_img, order = ord
common tiff_com, order, ifd, count

pc_hdr=286  ; header skip for subsequent images
   ; don't know if this is general or EPIX specific
on_error,2                      ;Return to caller if an error occurs

openr,lun,file, error = i, /GET_LUN, /BLOCK
if i lt 0 then begin ;OK?
 if keyword_set(lun) then free_lun,lun
 lun = -1
 message, 'Unable to open file: ' + file
 endif

hdr = bytarr(8)   ;Read the header
readu, lun, hdr

typ = string(hdr(0:1))  ;Either MM or II
if (typ ne 'MM') and (typ ne 'II') then begin
 message,'TIFF_READ: File is not a Tiff file: ' + string(file)
 return,0
 endif
order = typ eq 'MM'    ;1 if Motorola 0 if Intel (LSB first or vax)
endian = byte(1,0,2)  ;What endian is this?
endian = endian(0) eq 0  ;1 for big endian, 0 for little
order = order xor endian ;1 to swap...
```

```
; print,'Tiff File: byte order=',typ, ',  Version = ', tiff_int(hdr,2)

offs = tiff_long(hdr, 4) ;Offset to IFD

point_lun, lun, offs  ;Read it

a = bytarr(2)   ;Entry count array
readu, lun, a
;print,'entry count',a
count = tiff_int(a,0)  ;count of entries
; print,count, ' directory entries'
ifd = bytarr(count * 12) ;Array for IFD's
readu, lun, ifd   ;read it

; Insert default values:
compression = 1
bits_sample = 1
ord = 1
samples_pixel = 1L
pc = 1
photo = 1
rows_strip = 'fffffff'xl ;Essentially infinity

for i=0,count-1 do begin ;Print each directory entry
 value = tiff_read_field(i, tag, lun)  ;Get each parameter
 print,i,value
 case tag of  ;Decode the tag fields, other tags could be
added
256: width = value
257: length = value
258: bits_sample = value
259: compression = value
262: Photo = value
273: StripOff = value
274: Ord = value
277: samples_pixel = long(value)
278: Rows_strip = value
279: Strip_bytes = value
284: PC = value
320: ColorMap = value
else:   value = 0  ;Throw it away
 endcase
endfor


; Do a cursory amount of checking:
 if bits_sample(0) ne 8 then $
```

```
  message,'TIFF_READ: bits_sample must be 8'
 if compression ne 1 then $
  message,'TIFF_READ: Images must be un-compressed'
; if photo ge 4 then $
;  message,'TIFF_READ: Photometric Interpretation must be 0 to 3.'
 if (pc eq 2) and (samples_pixel ne 3) then $
  message,'TIFF_READ: RGB data must have 3 SamplesPerPlane'

nbytes = width * long(length)
strips_image = (length + rows_strip -1) / rows_strip


 if samples_pixel eq 1 then image = bytarr(width, length,n_img, /nozero)
 if samples_pixel eq 1 then img = bytarr(width, length ,/nozero)

  point_lun, lun, stripoff(0)   ;1st image data

hdr=bytarr(pc_hdr)

 for i=0,n_img-1 do begin
  readu, lun, img  ;Yes....
  image(*,*,i)=img
  readu, lun, hdr  ;header....

 endfor


free_lun, lun
return, image
end
```

--
Craig Stevens: Air-Sea Interaction
Natl. Inst. Water & Atmos. Research
po box 14-901 Kilbirnie WGTN   New Zealand
fx 64 (0)4 386 2153 ph 64 (0)4 386 0476