Subject: Re: WANTED: 2 IDL routines
Posted by deutsch on Mon, 31 Aug 1992 17:42:39 GMT
View Forum Message <> Reply to Message

In article <28AUG199216204945@stars.gsfc.nasa.gov>,
fireman@stars.gsfc.nasa.gov (Gwyn Fireman) writes:

```
>   Hello, IDL users -
>
>   I am looking for two array analysis routines.  They'd both probably
>  be pretty straightforward to write, so maybe one of you has already done so!
>
>  1. Given a 2-d array, user-defined endpoints of a line in that array, and
>  a width in pixels around that line, compute the 1-d average of the array along
>  the line, within the given width. Essentially PROFILE, averaged along a width.
>
>  2. Given a 3-d array and an arbitrary line through that array, return the
>  column density.  Essentially PROFILE in three dimensions.
>
```

I've done some similar stuff with 2D images... This is how I did it... The function GetStrip could use some added work (like adding a Width= keyword with the following code.) I don't have time to do it now, but if you do, I'd appreciate your additions...

```
WIDTH=10 & LENGTH=50 & ANGLE=45. & XCEN=200 & YCEN=300

s1=GetStrip(img,XCEN,YCEN,ANGLE+90,WIDTH,x2,y2)
band=fltarr(LENGTH)
for i=0,n_elements(s1) do begin
  band=band+GetStrip(img,x2(i),y2(i),ANGLE,LENGTH)
  endfor
band=band/WIDTH
```

Here's the little function:

```
 ------------------------------------------------------------ ------------------

function GetStrip,img,xcent,ycent,angle,length,xvect,yvect
;+
; NAME:
;   GETSTRIP
; PURPOSE:
;   This procedure returns a profile through an image at any angle or position.
;   First the theoretical line is calculated given the center, angle and length
;   in pixels.  Then the nearest neighbor to each point of this line is put
;   in the appropriate strip element.  Subpixel addition and interpolation is
;   not performed.  This unfortunately makes some of the returned profiles
;   a bit jaggy and you might want to SMOOTH the result.
; CALLING SEQEUNCE:
;   tmp = GetStrip(img,xcent,ycent,angle,length,xvect,yvect)
; INPUT:
;   IMG       This is the 2D image array from which the profile is to be taken.
;   XCENT     This is the X coordinate of the center of the profile.  It does
;             not need to be an exact integer.
;   YCENT     This is the Y coordinate of the center of the profile.  It does
;             not need to be an exact integer.
;   ANGLE     This is the angle of the profile in degrees counterclockwise
;             of the X axis.
;   LENGTH    This is the length in elements (and pixels) of the returned
;             profile.  Therefore no matter what the angle is, the returned
;             vector will have LENGTH elements and is a profile with a
;             physical length of LENGTH image pixels.
; OPTIONAL OUTPUT:
;   XVECT     This variable returns the array of positions of each profile
;             pixel.  These returned values are floating.  Add .5 to round
;             properly before FIXing or subscripting to IMG.
;   YVECT     This variable returns the array of positions of each profile
;             pixel.  These returned values are floating.  Add .5 to round
;             properly before FIXing or subscripting to IMG.
; OUTPUT:
;   tmp       The returned profile array.
; EXAMPLE:    In a North Up and East Left image of size 512x512 and pixelsize
;             of 1.5", the following call:
;               IDL> tmp=GetStrip(img,256,256,45,51)
;             returns a 51 element vector where tmp(25)=img(256,256) and
;             elements less than 25 are to the Southeast and elements greater
;             than 25 increase toward the Northwest.  The pixelsize of the
;             strip tmp is 1.5" also.
;               IDL> tmp=GetStrip(img,256,256,0,51)
;             is equivalent to...
;               IDL> tmp=img(256-25:256+25,256)
; HISTORY:
;   27-JUL-92 Added header and spiffed up this procedure.   (E. Deutsch)
;-

  theta=angle/!radeg
   pt1=[xcent+.5*(length-1)*cos(theta),ycent+.5*(length-1)*sin( theta)]
   pt2=[xcent-.5*(length-1)*cos(theta),ycent-.5*(length-1)*sin( theta)]

  len=indgen(length)
  xvect=pt2(0)+len*cos(theta)
  yvect=pt2(1)+len*sin(theta)

  strip=img(xvect+.5,yvect+.5)
  return,strip

end
```