Subject: Re: IDL 8.2, read pixel value along a surface
Posted by topratingblogs on Sun, 21 Jul 2013 21:42:13 GMT
View Forum Message <> Reply to Message

> I have some untested/uncommented code that performs the first method and the code for the second method is hidden in some old hard-drive.
>
> The procedure below can actually calculate a given line made of more than two points. The input parameters should be:
>
> Img: n x m image
>
> Pts: 2 x p points in the form [[Pt0x,Pt0y],[Pt1x,Pt1x],[Pt2x,Pt2x],...]
>
> InWidth: Pixel width
>
> The averaged values will be returned in xRes, yRes
>
> You can test the procedure like this:
>
>
>
> ;Test code
>
> a=dist(400)
>
> pts = [[0.0,0.0],[300.0,300.0]]
>
> w = 10
>
> GetCS_Image, a, pts, w, xres=x, yres=y
>
> plot, x,y
>
> ;*********
>
>
>
> Here is the code:
>
>
>
>
>
>
>
> PRO GetCS_Image, Img, Pts, InWidth, Distances = Distances, $
>
> xRes = xRes, yRes=yRes, PosRef=PosRef ;Output variables
>

```
>
>
>  IF N_ELEMENTS(InWidth) EQ 0 THEN Width=1.0 ELSE Width= ((ROUND(InWidth) MOD 2)
NE 1 ) ? (FLOAT(InWidth+1.0)>1.0) : (FLOAT(InWidth)>1.0)
>
>  IF N_ELEMENTS(Img)   EQ 0 THEN Message, 'GetCS_Image: Please supply an image',
/NoName
>
>  IF N_ELEMENTS(Pts)   LT 4 THEN Message, 'GetCS_Image: Please supply two points for the
cross-section', /NoName
>
>
>
>  FltHalfWidth    = (Width-1.0) / 2.0
>
>  nPoints         = (SIZE(Pts, /DIMENSIONS))[1]
>
>  nLines          = nPoints-1
>
>  FloatPts        = FLOAT(Pts)
>
>  ShiftedFloatPts = SHIFT(FloatPts,0,-1)
>
>  Distances       = SQRT(TOTAL(((ShiftedFloatPts-FloatPts)[*,0:(nLines-1)])^2, 1))
>
>  CumulDistances  = TOTAL(Distances,/CUMULATIVE)
>
>  WidthArr        = FINDGEN(Width)
>
>  OnesArr         = WidthArr*0.0+1.0
>
>  Angle           =  ATAN((shiftedfloatpts-FloatPts)[1,0:-2],(shiftedfloatpts-Flo atPts)[0,0:-2])
>
>  NormAngle       = Angle+!PI/2.
>
>  CosNormAngle    = COS(NormAngle)
>
>  SinNormAngle    = SIN(NormAngle)
>
>  CosAngle        = COS(Angle)
>
>  SinAngle        = SIN(Angle)
>
>  CeiledDist      = CEIL(Distances)
>
>  FOR I=0,nLines-1 DO BEGIN
>
>    xNorm   = (WidthArr-FltHalfWidth)*CosNormAngle[I]+FloatPts[0,I]
```

```
>
>    yNorm   = (WidthArr-FltHalfWidth)*SinNormAngle[I]+FloatPts[1,I]
>
>    xNorm   = xNorm # (FLTARR(CeiledDist[I])+1.0)
>
>    yNorm   = yNorm # (FLTARR(CeiledDist[I])+1.0)
>
>    xlocArr = OnesArr # FINDGEN(CeiledDist[I])*CosAngle[I]
>
>    ylocArr = OnesArr # FINDGEN(CeiledDist[I])*SinAngle[I]
>
>    profile = TOTAL(INTERPOLATE(Img, xlocArr+xNorm, ylocArr+yNorm, CUBIC=-0.5,
MISSING=0.0),1)/Width
>
>    IF I EQ 0 THEN BEGIN
>
>       nPos   = N_ELEMENTS(Profile)
>
>       xRes   = LIST(FINDGEN(nPos), /EXTRACT)
>
>       yRes   = LIST(Profile, /EXTRACT)
>
>       PosRef = LIST(nPos)
>
>    ENDIF ELSE BEGIN
>
>       nPos      = N_ELEMENTS(Profile)
>
>       xRes.Add,   FINDGEN(nPos)+CumulDistances[I-1], /EXTRACT
>
>       yRes.Add,   Profile, /EXTRACT
>
>       PosRef.Add, nPos+PosRef[I-1]
>
>    ENDELSE
>
>  ENDFOR
>
>  xRes   = xRes.ToArray()
>
>  yRes   = yRes.ToArray()
>
>  PosRef = PosRef.ToArray()
>
>  END
>
```

I am very new to IDL coding. I want to use the code above for my Impervious Surface Image. IS it okay if someone could highlight and comment which lines of the code should I change so it will work for my own image data? For instance, how will I read my image?

Help will be very much appreciated. Thank you.

JP
