Subject: IDLanROI->computeMask and "new" MASK_RULE for center points
Posted by Fabzi on Mon, 26 Mar 2012 15:16:37 GMT
View Forum Message <> Reply to Message

Hi IDLers,

To transform vector info into raster images I am using the very usefull
IDLanROI object, with the ->ComputeMask() function.

Unfortunately, ComputeMask thinks "pixel", I think "grind point". Here
is a very simple program that illustrates what I mean. I programmed a
"MASK_RULE=3" if you want, that computes a mask following the rule
"center point of the pixel is included".

Here the code:

```
pro roimaskrules, npix

  mask = BYTARR(npix, npix)

  ; Polygon vertices
  x = [0.6,2.6,2.6,0.6,0.6]
  y = [0.6,0.6,2.6,2.6,0.6]

  roi = OBJ_NEW('IDLanROI', x, y)
  t0 = SYSTIME(/SECONDS)
  mask_0 = roi->ComputeMask(MASK_IN=mask*0B, MASK_RULE=0)
  print, 'Time 0: ', SYSTIME(/SECONDS)-t0
  t0 = SYSTIME(/SECONDS)
  mask_1 = roi->ComputeMask(MASK_IN=mask*0B, MASK_RULE=1)
  print, 'Time 1: ', SYSTIME(/SECONDS)-t0
  t0 = SYSTIME(/SECONDS)
  mask_2 = roi->ComputeMask(MASK_IN=mask*0B, MASK_RULE=2)
  print, 'Time 2: ', SYSTIME(/SECONDS)-t0

  ; The "center point method"
  t0 = SYSTIME(/SECONDS)
  i = INDGEN(npix) # (LONARR(npix) + 1)
  j = (LONARR(npix) + 1) # INDGEN(npix)
  mask_3 = mask
  cont = roi->ContainsPoints(i, j)
  p_in = where(cont ge 1, cnt_in)
  if cnt_in ne 0 then mask_3[p_in] = 255B
  print, 'Time 3: ', SYSTIME(/SECONDS)-t0

  if npix le 4 then begin ;print results to show what we want
    print, 'mask_0: Boundary only.'
    print, mask_0
```

```
      print, 'mask_1: Interior only'
      print, mask_1
      print, 'mask_2: Boundary + Interior'
      print, mask_2
      print, 'mask_3: Contains centerpoint'
      print, mask_3
    endif

    OBJ_DESTROY, roi

end
```

In this program, Mask_3 (and only mask_3) does what I want. For example with a 4x4 array:

```
IDL> roimaskrules, 4
Time 0:    4.0054321e-05
Time 1:    3.0994415e-05
Time 2:    2.1934509e-05
Time 3:    6.3180923e-05
mask_0: Boundary only.
   0   0   0   0
   0 255 255 255
   0 255   0 255
   0 255 255 255
mask_1: Interior only
   0   0   0   0
   0   0   0   0
   0   0 255   0
   0   0   0   0
mask_2: Boundary + Interior
   0   0   0   0
   0 255 255 255
   0 255 255 255
   0 255 255 255
mask_3: Contains centerpoint
   0   0   0   0
   0 255 255   0
   0 255 255   0
   0   0   0   0
```

So, that works. But the problem is that if I want to do it for large array this becomes VERY slow. Don't even try with large number of polygons ;-):

```
IDL> roimaskrules, 4000
Time 0:      0.027777910
Time 1:      0.027390003
Time 2:      0.018553019
Time 3:        6.7377200
```

I come to the point: how can I make it faster? I know that testing each
point is the problem here, but I don't understand the real difference
between "my" MASK_RULE and IDL's MASK_RULE (why would IDL not implement it?)

Thanks for your help,

Fabz