Subject: Re: Understanding IDLanROI
Posted by KRDean on Tue, 19 Oct 2010 15:02:43 GMT
View Forum Message <> Reply to Message

On Oct 8, 3:13 pm, kBob <krd...@gmail.com> wrote:
> I am attempting to use IDLanROI to determine if Shapefiles overlap
> another Shapefile.
>
> Sometimes it works, sometimes it doesn't. Especially, if the polygon
> lies within the interior.
>
> I put together a test program that retrieves the States and uses
> IDLanROI to determine if it lies within the US.
>
> Why does Colorado not considered to be in the United States, among
> some other interior States?
>
> Is IDLanROI not the IDL routine to use to perform this check?
>
> Kelly Dean
> Milliken, CO
>
> ;+
> ;
> ;
> ;
> ;---------- ----------
> PRO ITT_PolyOverlap
>
> states_path = FILEPATH('states.shp',SUBDIR=['resource', 'maps',
> 'shape'])
> country_path = FILEPATH('country.shp',SUBDIR=['resource', 'maps',
> 'shape'])
>
> PRINT, states_path
> PRINT, country_path
>
> oSHP = OBJ_NEW('IDLffShape', country_path)
> ent1 = oSHP -> GetEntity(34, /ATTRIBUTES) ; USA
> OBJ_DESTROY, oSHP
>
> PRINT, ' -- Working with country : ', (*ent1.attributes).attribute_4
>
> oSHP = OBJ_NEW('IDLffShape', states_path)
> oSHP -> GetProperty, N_ENTITIES = num_ent
> FOR staten = 0, num_ent DO BEGIN
>
> oSHP = OBJ_NEW('IDLffShape', states_path)

```
>    ent0 = oSHP -> GetEntity( staten, /ATTRIBUTES )
>
>    oROI = OBJ_NEW('IDLanROI', (*ent1.vertices)   )
>    pttest = oROI -> ContainsPoints( (*ent0.vertices) )
>    OBJ_DESTROY, oROI
>
>    overlap = WHERE( ptTest GT 0, count )
>
>    IF ( count GT 0 ) THEN BEGIN
>      PRINT, ' === ' , (*ent0.attributes).attribute_1, ' inside ',
>  (*ent1.attributes).attribute_4, ' === ', count
>    ENDIF ELSE BEGIN
>      PRINT, ' <<< ' , (*ent0.attributes).attribute_1, ' outside ',
>  (*ent1.attributes).attribute_4, ' >>> ', count
>    ENDELSE
>
>  ENDFOR
>
>  OBJ_DESTROY, oSHP
>
>  END
```

To answer my own question ...

Yes, IDLanROI is NOT the IDL routine to perform this check.

After pondering on this for a couple of weeks, I figure what I needed to do.

IDLgrROI is the prefer IDL routine to use.

It is a tessellation thing, especially when dealing with State and Country Shapefiles that may contain gaps. You may get away with very simple polygons using IDLanROI when dealing with a handful of vertices, but IDLgrROI and IDLgrROIGroup are the routines you should be using along with incorporating IDLgrTessellator in the code.

The code below produces the desire results I was after.

Kelly Dean
Milliken, CO

P.S. I thought IDL 8.0 removed memory leaks? I had to destroy some Objects after a code run.

```
;+
;
; @file_comments
```

```
; <P>Looking for (State) polygons that overlap (Country) polygons.
;
 ;------------------------------------------------------------ -----------
PRO ITT_TessOverlap

COMPILE_OPT DEFINT32, STRICTARR

states_path = FILEPATH('states.shp',SUBDIR=['resource', 'maps',
'shape' ])
country_path = FILEPATH('country.shp',SUBDIR=['resource', 'maps',
'shape' ])
oUSA = OBJ_NEW('IDLffShape', country_path )
oUSA -> GetProperty, N_ENTITIES = num_ent
;entUSA = oUSA -> GetEntity( 2, /ATTRIBUTES ) ; CAN
entUSA = oUSA -> GetEntity( 34, /ATTRIBUTES ) ; USA
;entUSA = oUSA -> GetEntity( 73, /ATTRIBUTES ) ; MEX
PRINT, ' -- Working with country : ', (*entUSA.attributes).attribute_4
oROIGroup = OBJ_NEW( 'IDLgrROIGroup' )
oROI = OBJARR( entUSA.n_parts )
FOR ii = 0L, entUSA.n_parts-1 do begin
  IF (ii EQ entUSA.n_parts-1)THEN BEGIN
    startpoint = (*entUSA.parts)[ii]
    endpoint   = entUSA.n_vertices-1
  ENDIF ELSE BEGIN
    startpoint = (*entUSA.parts)[ii]
    endpoint   = (*entUSA.parts)[ii+1]-1
  ENDELSE
  data = (*entUSA.vertices)[*, startpoint:endpoint]
  oTess = OBJ_NEW('IDLgrTessellator')
  oTess -> AddPolygon, data
  x = oTess -> Tessellate(verts, polys)
  oROI[ii] = OBJ_NEW('IDLgrROI', data = verts )
  oROIGroup -> ADD, oROI[ii]
ENDFOR
oStates = OBJ_NEW('IDLffShape', states_path )
oStates -> GetProperty, N_ENTITIES = nstates
FOR staten = 0, nstates-1 DO BEGIN
  overlap = 0
  entState = oStates -> GetEntity( staten, /ATTRIBUTES )
  PRINT,  ' -- Working with state : ',
(*entState.attributes).attribute_1
  FOR ii = 0L, entState.n_parts-1 do begin
    IF (ii EQ entState.n_parts-1)THEN BEGIN
      startpoint = (*entState.parts)[ii]
      endpoint   = entState.n_vertices-1
    ENDIF ELSE BEGIN
      startpoint = (*entState.parts)[ii]
      endpoint   = (*entState.parts)[ii+1]-1
```

```
    ENDELSE
    data = (*entState.vertices)[*, startpoint:endpoint]
    oTess = OBJ_NEW('IDLgrTessellator')
    oTess -> AddPolygon, data
    x = oTess -> Tessellate(verts, polys)
    pttest = oROIGroup -> ContainsPoints( verts )
    npts = N_ELEMENTS( ptTest )
    FOR sym = 1, 3 DO BEGIN
      indx = WHERE( ptTest EQ sym, symcnt )
      IF ( symcnt GT 0 ) THEN BEGIN
        overlap = ( overlap GE 0 ) ? 1 : 0
      ENDIF
    ENDFOR
  ENDFOR
  IF ( overlap GT 0 ) THEN BEGIN
    PRINT, ' === ' , (*entState.attributes).attribute_1, ' inside ',
(*entUSA.attributes).attribute_4, ' === '
  ENDIF ELSE BEGIN
    PRINT, ' <<< ' , (*entState.attributes).attribute_1, ' outside ',
(*entUSA.attributes).attribute_4, ' >>> '
  ENDELSE
ENDFOR
OBJ_DESTROY, oROIGroup
FOR staten = 0, nstates-1 DO OBJ_DESTROY, oROI[staten]
PRINT, " -- That's all Folks!"
END
```