Subject: overlap of two polygons

Posted by suruchi on Sat, 18 May 2013 07:57:20 GMT
View Forum Message <> Reply to Message

Hello

What i have is set of rois defined over 1024x 1024 image. So for finding whether two different rois (ROI1 & ROI2) from two different image cover same region of CCD image plane I can use :

```
ROI1 -> GetProperty, data=a
result(k)=ROI2 -> ContainsPoints(a[0,k],a[1,k])
```

This way i can figure out whether two rois have overlapping region or not.

But, what if I wish to do the same exercise using latitude -longitude information of each image.
A pixel say (500,500) will have different lat-long values in different images.

Now if I convert the roi vertices x & y to lat-long can I create a Polygon?
And using this lat-lon vertices information of each roi can I find whether two set of rois overalp (so now testing overlap not in image plane but region covered by lat-lon range of the planetary object)?

So essential query is how to find the overlap of two regions defined by only vertices and second what if these vertices are in terms of lat-long?

Any suggestion will be appreciated.

Thanks
Suruchi

Subject: Re: overlap of two polygons

Posted by Chip Helms on Mon, 20 May 2013 18:52:41 GMT
View Forum Message <> Reply to Message

I think something along the lines of the following would tell you where two regions overlap (although there is probably a more efficient way of doing this)

```
nROIarr = bytarr(1024,1024)
 ; add 1 to each roi pixel
 ;(I'm assuming the indices of the ROIs are in ROIdex1 and ROIdex2)
nROIarr[ROIdex1]++
nROIarr[ROIdex2]++
 ; overlapping points will be those with values of 2
overlapdex = where(nROIarr eq 2, nOverlap)

 ; and just to get even fancier...
 ; here's how you can define polygons outlining these overlapping regions
 ; using contour's path_xy and path_info keywords
 ; (word of warning...if you're polygon ends up with a height
 ; or width less than 2 pixels this might not work...also I should
 ; credit Ken Knapp with introducing me to this nifty piece of IDL magic)

 ; calculate the contours (no contour plot is drawn)
contour, nROIarr, path_info=pinfo, path_xy=pxy, /path_data_coords, level=[2]
 ; locate all overlap polygons (we're looking for contours around higher values)
polyloc = where(pinfo.hihg_low eq 1, npoly)
 ; initialize list to hold polygon indices
xpoly = list()
ypoly = list()
 ; loop through each overlapping region and unpack the contour polygons
for ip=0, npoly-1 do begin
   ; determine index of first point in polygon
  i1 = pinfo[polyloc[ip]].offset
   ; determine index of last point in contour
  i2 = i1 + pinfo[polyloc[ip]].n - 1
   ; grab and store contour position
  xpoly.Add, pxy[0,i1:i2]
  ypoly.Add, pxy[1,i1:i2]
endfor
```

Each elements of the xpoly and ypoly lists should be a vector containing the x and y coordinates of each vertex of the enclosing polygon, respectively.

On re-reading your post I think I'm not sure I covered the question you were asking (my overeagerness to share nifty IDL tricks strikes again). If you have a list of vertices, you can use the method I outlined above if you first use polyfillv:

```
nROIarr[polyfillv(xindex, yindex, 1024, 1024)] = 2
```

If they are lat/lon pairs and you have 1D lat and lon vectors corresponding to the rows and columns of the 1024x1024 array then you could do something like:

```
 ; lat, lon are the list of all lats and lons
 ; vertlat and vertlon are the lat lons of the vertices
nROIarr[polyfillv(value_locate(lon,vertlon),$
     value_locate(lat,vertlat),1024,1024)] = 2
```

And with that I should probably stop procrastinating and get back to work.

Cheers,
Chip