Subject: A distracting puzzle
Posted by John-David T. Smith on Mon, 17 Sep 2001 20:58:37 GMT
View Forum Message <> Reply to Message

Given a polygon defined by the vertex coordinate vectors x & y, we've seen that we can compute the indices of pixels roughly within that polygon using polyfillv(). You can run the code attached to set-up a framework for visualizing this. It shows a 10x10 pixel grid with an overlain polygon by default, with pixels returned from polyfillv() shaded.

You'll notice that polyfillv() considers only integer pixels, basically truncating any fractional part of the input polygon vertices (you can see this by plotting fix([x,x[0]]), etc.). For polygons on a fractional grid, this error can be significant.

The problem posed consists of the following:

Expand on the idea of the polyfillv algorithm to calculate and return those pixels for which *any* part of the pixel is contained within the polygon, along with the fraction so enclosed.

For instance, the default polygon shown (invoked simply as "poly_bounds"), would have a fraction about .5 for pixel 34, 1 for pixels 33 & 43, and other values on the interval [0,1] for the others. Return only those pixels with non-zero fractions, and retain polygon vertices in fractional pixels (i.e. don't truncate like polyfillv() does).

JD

```
pro poly_bounds,x,y,N=n
  if n_elements(n) eq 0 then n=10
  if n_elements(x) eq 0 then begin
     x=[1.2,3,5.3,3.2] & y=[1.3,6.4,4.3,2.2]
  endif
  window,XSIZE=500,YSIZE=500
  ;; Set up the plot region, etc.
  plot,[0],[0],XRANGE=[0.,n],YRANGE=[0.,n], XMINOR=-1,YMINOR=-1, $
        XTICKS=n,YTICKS=n,POSITION=[.05,.05,.95,.95],TICKLEN=0,/NODA TA
  p=polyfillv(x,y,n,n)
  for i=0,n_elements(p)-1 do begin
     xp=p[i] mod n
     yp=p[i]/n
      polyfill,[xp,xp,xp+1,xp+1],[yp,yp+1,yp+1,yp],COLOR=!D.N_COLO RS/2
  endfor
  oplot,[x,x[0]],[y,y[0]]
  for i=0,n-1 do begin
     plots,i,!Y.CRANGE
```

```
    plots,!X.CRANGE,i
    for j=0,n-1 do begin
      plots,i+.5,j+.5,PSYM=3
      xyouts,i+.1,j+.1,strtrim(i+j*n,2)
    endfor
  endfor
end
```

File Attachments

1) poly_bounds.pro, downloaded 118 times

Subject: Re: A distracting puzzle
Posted by John-David T. Smith on Tue, 25 Sep 2001 16:54:40 GMT
View Forum Message <> Reply to Message

Stein Vidar Hagfors Haugan wrote:
>
> If what's being sought here is only to distinguish which pixels have *some*
> area inside the polygon and which do not, wouldn't it be sufficient to check
> the corners? I.e., in a continuum of pixel coordinates, given corners with
> coordinates [0,0], [1,0], [1,1], [0,1], it can be checked whether each of
> those are inside versus outside any defined polygon. If one or more of the
> corners is inside, then some area is also inside..
>
> I have included some simple-minded routines I wrote some years ago to check
> whether a point is inside or outside a polygon...

Thanks Stein Vidar. Your method would seem to provide the answer for the boolean question; however, my intent was to provide a list of pixels which are at least partly inside the polygon, *along with* a list of their fractional areas included. I came up with a solution I call polyfillaa, which is a direct replacement for polyfillv.

inds=polyfillaa(x,y,sx,sy,AREAS=a)

returns the pixel indices, along with the clipping areas if desired. It performs a straightforward form of polygon clipping. The "aa" is for anti-aliasing, which is basically what it does. It works quite well, but is very slow, thanks to a surplus of looping. In general it returns more pixels than polyfillv, which neglects pixels with small areas inside, and (erroneously, I feel) truncates polygon points to integer pixels.

I may document it and put it up somewhere soon, but I'm embarrassed by all the for loops. We'll see.

JD

Subject: Re: A distracting puzzle
Posted by Martin Downing on Wed, 26 Sep 2001 08:53:25 GMT
View Forum Message <> Reply to Message

Ah go on JD, show your code - then the rest of us can decide whether we could do better without reinventing the wheel!

Martin

"JD Smith" <jdsmith@astro.cornell.edu> wrote in message news:3BB0B6D0.43C7859F@astro.cornell.edu...
> Stein Vidar Hagfors Haugan wrote:
>>
>> If what's being sought here is only to distinguish which pixels have *some*
>> area inside the polygon and which do not, wouldn't it be sufficient to check
>> the corners? I.e., in a continuum of pixel coordinates, given corners with
>> coordinates [0,0], [1,0], [1,1], [0,1], it can be checked whether each of
>> those are inside versus outside any defined polygon. If one or more of the
>> corners is inside, then some area is also inside..
>>
>> I have included some simple-minded routines I wrote some years ago to check
>> whether a point is inside or outside a polygon...
>
> Thanks Stein Vidar. Your method would seem to provide the answer for
> the boolean question; however, my intent was to provide a list of pixels
> which are at least partly inside the polygon, *along with* a list of
> their fractional areas included. I came up with a solution I call
> polyfillaa, which is a direct replacement for polyfillv.
>
> inds=polyfillaa(x,y,sx,sy,AREAS=a)
>
> returns the pixel indices, along with the clipping areas if desired. It
> performs a straightforward form of polygon clipping. The "aa" is for
> anti-aliasing, which is basically what it does. It works quite well,
> but is very slow, thanks to a surplus of looping. In general it returns
> more pixels than polyfillv, which neglects pixels with small areas
> inside, and (erroneously, I feel) truncates polygon points to integer
> pixels.
>
> I may document it and put it up somewhere soon, but I'm embarrassed by
> all the for loops. We'll see.
>
> JD