

---

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,/NODATA
  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_COLORS/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 162 times

---