Subject: A new puzzle for histogram
Posted by gknoke on Fri, 15 Sep 2006 22:07:02 GMT
View Forum Message <> Reply to Message

So, I've got this piece of code which is horribly horribly inefficient.
I know the solution lies in a clever application of the histogram function, but being Friday afternoon my brain isn't seeing it. Anyone else have any insight on how I might approach it?

This particular routine is mapping a piece of data from polar to cartesian coordinates. Currently the code generates sin/cos angle tables and calculates the x,y coordinates in meters for each range/angle, and then converts that to an x,y coordinate in terms of pixels from the center. I realize the calculation of the x,y coordinates can be replaced with a simple vector operation, but I can't see how to turn the separate resulting arrays for x and y into a single array I can use the histogram function to match to the mapped grid.

```
;Setup the output grid
map = fltarr(xysize, xysize)
count = intarr(xysize, xysize)

cos_table = cos(angles)
sin_table = sin(angles)

for j_theta = 0, n_elements(angles)-1 do begin
  for i_range = 0, n_range-1 do begin
    ;Calculate x and y in meters
    x = r_pts(i_range)*cos_table(j_theta)
    y = r_pts(i_range)*sin_table(j_theta)

    ;Find corresponding pixel on mapped grid
    ix = round((x-x0)/cellsize)+xysize/2
    jy = round((y-y0)/cellsize)+xysize/2

    ;If the pixel coord is inside the image put the data point there
    if(ix ge 0 and ix le xysize-1) then begin
      if(jy ge 0 and jy le xysize-1) then begin
        map(ix,jy)=map(ix,jy) + data(i_range,j_theta)
        count(ix,jy)=count(ix,jy)+1
      endif
    endif
  endfor
endfor ;End of nearest neighbor loops
```

Thanks,

--Greg