Subject: Re: A new puzzle for histogram
Posted by gknoke on Fri, 15 Sep 2006 22:58:49 GMT
View Forum Message <> Reply to Message

Hm... not quite what I had in mind. I've solved the half of the problem dealing with x and y, what I have now is:

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

;Calculate x and y in meters
x = r_pts#cos_table
y = r_pts#sin_table

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

count = hist_2d(ix, jy, max1=(xysize-1),max2=(xysize-1),min1=0,min2=0)
```

Which does away with the need for the for loops. The only line I can't figure out how to replace is:

```
map(ix,jy)=map(ix,jy) + data(i_range,j_theta)
```

Is there a hist_2d equivalent to reverse_indices? If I could simply figure out the indices of all the points I've pulled out of ix and jy I suddenly have all of the indices of the data points I need. From there it's a simple problem of summing the points that are in the same histogram bins.

Thanks,

--Greg

Jean H. wrote:

```
>  Hi,
>
>  If I corectly understand your problem, you might want to look at the
>  rebin function:
>
>  a = indgen(4)   ;could be your angle
>  b = transpose(indgen(4)) ;could be your range
>
>  print, rebin(a,4,4) * rebin(b,4,4)
>
>>>      0      0      0      0
>        0      1      2      3
```

```
>        0     2     4     6
>        0     3     6     9
>
>  hope that helps...
>
>  Jean
>
>  gknoke wrote:
>>  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
>>