Subject: Re: Histogram Hot-shots Required
Posted by davidf on Fri, 16 Jul 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Liam Gumley (Liam.Gumley@ssec.wisc.edu) writes:

> Assuming I know the minimum and maximum values (the range) used in
> creating the histogram, the histogram binsize, and the number of bins,
> the zero-based bin index is given by
>
> bin_index = long(float(pixel_value - histogram_min_value) /
> float(binsize))
>
> and then to protect against pixel values LT histogram minimum value, or
> GE histogram maximum value
>
> bin_index = (bin_index > 0L) < (number_of_bins - 1L)
>
> At least, that's how it looks to me on paper....

Well, I think this is the correct answer, surely. But
I still have rather large differences on my plot between
this calculated value and the "perceived" value on
the graph. This must be a function of the distribution of
the data in certain bins and the way the graph is being
drawn.

To solve this I may have to go to a bar plot, which
I have *really* been trying to avoid. :-(

In any case, I appreciate the help very much.

Cheers,

David

P.S. I am going to pursue Ronn Kling's suggestions about
Reverse_Indices some more, even if I don't think it is the
answer in this case. I just love the irony of having a joke
throw-away line become the real answer. :-)

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: Histogram Hot-shots Required
Posted by Liam Gumley on Fri, 16 Jul 1999 07:00:00 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> What I can't figure out tonight is how to find out
> what bin that pixel is in, given that I know the pixel
> value. (Even as I write this sentence I have the sense
> that this is a trivial exercise, but I'm afraid it is
> not yielding the shear number of hours I have spent
> on it. At least not for me.)

Assuming I know the minimum and maximum values (the range) used in creating the histogram, the histogram binsize, and the number of bins, the zero-based bin index is given by

bin_index = long(float(pixel_value - histogram_min_value) /
float(binsize))

and then to protect against pixel values LT histogram minimum value, or GE histogram maximum value

bin_index = (bin_index > 0L) < (number_of_bins - 1L)

At least, that's how it looks to me on paper....

Cheers,
Liam.

--
Liam E. Gumley
Space Science and Engineering Center, UW-Madison
http://cimss.ssec.wisc.edu/~gumley

Subject: Re: Histogram Hot-shots Required
Posted by ronn on Fri, 16 Jul 1999 07:00:00 GMT
View Forum Message <> Reply to Message

In article <MPG.11f876f9f15942a98982b@news.frii.com>,
davidf@dfanning.com (David Fanning) wrote:
> Ok you histogram cowboys. Let's see what ya got!
>
OK, I grew up on a ranch and I work with histograms so I qualify.

> I don't know if it's early on-set Alzheimers, or
> I'm just pressing with a long soccer weekend coming
> up, or I just haven't had enough beers yet, but

> this one is stumping me. I thought I'd give you
> folks a chance to see if you can explain something
> simply enough that even I can understand it. :-)
>

I also love challenges! (Must be the Klingon in me)

<lots cut ot>

> In any case, I'm fresh out of ideas as well as beer. So
> I thought I'd turn it over to you. Any ideas will be
> *gratefully* accepted. I'm sure it has something to do
> with that Reverse_Indices keyword, but whatever it is
> escapes me. :-)
>

You got it! Reverse indices will help you. Turn to my book on pages 2-53 and 7-163 for the write up on Reverse Indices. But here is how I would do it.

Given an array created by dist(250) find all the pixels that equal 100.

```
window,0,xsize=250,ysize=250
device,decomposed=0
loadct,0
z = dist(250)
tek_color
tv,z
h = histogram(z,reverse=r)
top = n_elements(r)
;r is a run length encoded array where the beginning of the array is an
;index into the rest of the array.
;Have to worry about the case where a pixel value  might equal the
;index.
;So create a two arrays from r. The first of the indexes, the second of
;the values.
index = where(r eq top,count)
if count eq 0 then print, 'something wrong'
indices = r[0:index[0]] ;indices of the values
values = r[index[0]+1:*] ;the values only
binIndex = where(values eq 100,count) ;find the pixel value=100
if count eq 0 then print, 'something wrong'
realIndex = binIndex + index[0] ;get the "real" index of it.
;now we have to find where this value falls into the indices.  The r
;array only has begin and end points into the array.
bottom = where( realIndex[0] gt indices,count)
if count eq 0 then print, 'something wrong'
;count is the bottom index of the values. To get all the pixels that
;equal 100 do
```

```
binPixels =  r[r[count]:r[count+1]]
print, n_elements(binPixels)
;And just to make sure
z[binPixels]=3
tv,z
```

Hope this helps!

-Ronn

--
Ronn Kling
Ronn Kling Consulting
www.rlkling.com

Sent via Deja.com http://www.deja.com/
Share what you know. Learn what you don't.