Subject: rebin and half pixel offset
Posted by Robert Barnett on Fri, 28 May 2004 00:10:47 GMT
View Forum Message <> Reply to Message

I wondered if anyone can verify if I understand the behaviour of rebin correctly. Thanks in advance for looking at this problem.

I'm currently putting together a ROI drawing program which allows the user to draw regions on a zoomed image. Sometimes it is preferrable to see a bilinear interpolated image whilst at other times it is preferrable to see a nearest neighbour image.
After using rebin I noticed that there was a difference between the two methods. Because of the way rebin works, the bilinear method offsets the image and hence causes my ROI's (drawn using plots) to appear offset.

The offset is 0.5 pixels if you do the shifting before rebin or it may be zoom/2 pixels if the shifting is done after rebin

I've put together a little test program to demonstrate this.
The input array is [0,1 ... m-2,m-1]
This array is rebined to a larger array of size (m * zoom)
The results of using neareast neighbour and bilinear interpolation are printed. The difference is also printed

```
pro testRebinOffset, m, zoom
 m = floor(m > 1.0)
 zoom = floor(zoom > 1.0)
 n = zoom * m ; The size of the output array
 input = float(indgen(m))  ; The input array
    ; use float so that rebin an do       ; floating point arithmetic

 ; Rebin using bilinear interpolation and then apply the shift
 bi = round(shift(rebin(input,n),zoom/2))
 ; Fix up the 'wrapping' caused by the shift function
 bi[0:zoom/2] = input[0]
 print, "Bilinear Interpolation", bi
 ; Rebin using nearest neighbour method
 nn = round(rebin(input,n,/sample))
 print, "Nearest Neighbour", nn

 print, "Difference", nn - bi
end
```

; An example usage

```
IDL> testrebinoffset,3,4
Bilinear Interpolation         0         0         0         0
           1         1         1         1         2         2
```

```
          2          2
Nearest Neighbour          0          0          0          0
    1
          1          1          1          2          2          2
          2
Difference          0          0          0          0          0
          0          0          0          0          0          0
          0
```

This test fails when the input array is anything more complicated than
an indgen array, however, I am fairly certain that this is the best
approximation for making coordinates in both spaces equivalent

Regards, Robbie

Westmead Hospital,
Sydney
Australia