
Subject: Re: rebin and half pixel offset

Posted by [peter.julyan](#) on Mon, 07 Jun 2004 08:36:05 GMT

[View Forum Message](#) <> [Reply to Message](#)

Robert Barnett <retsil@zipworld.com.au> wrote in message
news:<40B68387.6020207@zipworld.com.au>...

```
> 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 preferable to
> see a bilinear interpolated image whilst at other times it is
> preferable 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 rebinned to a larger array of size (m * zoom)
> The results of using nearest 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 and 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

Robert,

This seems pretty much to make sense, this is spelt out explicitly in
 the manual for CONGRID where, since 5.5, there is the /CENTER option
 to "shift the interpolation so that points in the input and output
 arrays are assumed to lie at the midpoint of their coordinates rather
 than at their lower-left corner." I came across this a while back
 doing some ROI stuff and use CONGRID in this instance.

Hope this helps, Pete.

Peter Julyan, Ph.D - PET Physicist
 North Western Medical Physics
 Christie Hospital NHS Trust
 Withington, Manchester, M20 4BX
 Tel: 0161 446 3078 Fax: 0161 446 3543
 e-mail: p.julyan@physics.org
<http://www.ManPET.man.ac.uk>
