
Subject: interpolate over bad pixels

Posted by [Em](#) on Mon, 27 Jun 2005 22:22:59 GMT

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

I was hoping you could help me out here. I have a set of images where a few pixels are bad, or set to "NaN". Can you tell me how I can interpolate over these bad pixels, using preferably a spline or cubic interpolation? The region with bad pixels is usually around 3-8 pixels in diameter on a 1024x1024 image. Is there a routine that takes the image and a list of the indices of bad pixels, and outputs an interpolated image?

I have looked at the convol routine, and tri_surf. If these do what I need them to do, I have not figured out how to use them...

Thanks,
Em

Subject: Re: interpolate over bad pixels

Posted by [Jess](#) on Wed, 29 Jun 2005 02:56:59 GMT

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

Hi Em,

Sorry I can't help but this seems similar to what I want to do except yours is 2D whereas I want to expand a vector with congrid, but have it only use certain pixels (in my case non-zero pixels) and have it stretch and interpolate correctly. I ended up writing an iterative script which was sufficient in the vector case, but would much prefer a smarter way and being able to do it over 2D images also.

Look forward to the solutions you get,
Jess

Subject: Re: interpolate over bad pixels

Posted by [Jess](#) on Wed, 29 Jun 2005 10:06:57 GMT

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

Hi again Em,

This is what I'm now doing for the 1D arrays which works really well. I'm sure its not the fastest, but it works fine. Below I've given the syntax for a 2D example (sorry if its not quite right, as I haven't tested this 2D one).

Create 3 arrays, one that contains just the indices of the good pixels, second that contains just the values of the good pixels, and third that contains the indices of all the pixels, eg.

```
; where not eq NAN
wh_good = WHERE(FINITE(orig_image))
; image with just good pixels
image_good = orig_image(wh_good)

; size of image
dims = size(orig_image,/dimensions)
xnum = dims[0]
ynum = dims[1]
; indices of all pixels
wh_all = indgen(xnum,ynum)

; Then interpolate, to get corrected image
; eg. linear interpolation
image_corr = interpol(image_good,wh_good,wh_all)
; or for cubic spline
image_corr = interpol(image_good,wh_good,wh_all,/spline)
```

Have a look at the help on where, finite and interpol.

Cheers,
Jess
