Subject: Re: efficient kernel or masking algorithm ? UPDATE
Posted by Martin Downing on Mon, 26 Feb 2001 15:32:14 GMT
View Forum Message <> Reply to Message

"John-David Smith" <jdsmith@astro.cornell.edu> wrote in message news:3A99C6B4.10549265@astro.cornell.edu...
>
> P.S. I think I originally got the idea from sigma_filter.pro, a NASA library
> routine, dating back to 1991. It's chock-full of other good tidbits too.
> Thanks Frank and Wayne!

Hi John,
Just checked the file SIGMA_FILTER.pro at
http://idlastro.gsfc.nasa.gov/ftp/pro/image/?N=D
I really must spend more time browsing these great sites.
The code is similar, however it does not calculate the true variance under the mask
they calculate for a box width of n, (ignoring centre pixel removal):

```
-------------------------------------
 mean_im=(smooth(image, n) )
 dev_im = (image - mean_im)^2
 var_im = smooth(dev_im, n)/(n-1)
-------------------------------------
```

This is not the true variance of the pixels under the box mask, as each pixel in the mask is having a different mean subtracted.
i.e (read this as a formula if you can!)
Pseudo_Variance = SUM ij ((I(x+i,y+j) - MEAN(x+i,y+j)^2) /(n-1)

instead of true variance:
Variance = SUM ij ((I(x+i,y+j) - MEANxy)^2) /(n-1)
which can be reduced to : {(SUM ij ((I(x+i,y+j)^2) - (SUM ij I(x+i,y+j)) ^2)/n }/(n-1)
hence the non loop method we use below:

```
---------------------------
; calc box mean
mean_im = smooth(image, n)
; calc box mean of squares
msq_im = smooth(image^2, n)
; hence variance
 var_im = ( msq_im - mean_im^2) * (n/(n-1.0))
---------------------------------
```

cheers

Martin

PS: Sorry about my before-and-after-coffee postings this morning, outlook

decided to post my replies whilst I was still pondering - how kind - I've killed that *feature* now :)