Subject: Re: Gaussian Fit to background of image for subtraction
Posted by Vince Hradil on Tue, 06 Feb 2007 21:09:23 GMT
View Forum Message <> Reply to Message

On Feb 6, 11:25 am, "rpert...@gmail.com" <rpert...@gmail.com> wrote:
> Hello,
> I am doing some image analysis, and my image consists of several
> bright spots that I need to detect. I was able to write a program that
> would do just that...find the pixels that are larger than a threshold,
> group close pixels together and label different blobs as different
> spots by marking a 'plus sign' on the spot. Except, it does not 'see'
> all the spots, and lowering the threshold results in 'seeing' spots
> that are not there. Therefore, I am considering some filtering that I
> need to do to my background as it is not uniform and was suggested to
> perform a gauss 1d or 2d to the background to subtract it (and exclude
> the spots as I do that), and then see if i can 'see' all the spots....
>
> I am not sure how to do a gauss fit to background though...any
> suggestions?
> Thanks!
> rp

You could try a high-pass filter in the fourier domain?
Or try fitting a 2d gaussian by brute-force - or 2d-gaussian plus
spots. Is there a reason to believe that the background is gaussian
or was that just suggested because it is "smooth"? If it's the
latter, then I would try the fourier filtering...

Subject: Re: Gaussian Fit to background of image for subtraction
Posted by Karsten Rodenacker on Tue, 06 Feb 2007 21:47:04 GMT
View Forum Message <> Reply to Message

You could also use morphological operations. E.G. to detect your blobs
apply a morph_tophat and an appropriate threshold. kernel or structuring
element should be slightly larger than your blobs. tophat consits of a
morphological smoothing (open) to generate so to say the background which
is than subtracted from the original.
Tophat is relatively unknown but surprisingly effective.

Regards
karsten

Am Tue, 06 Feb 2007 18:25:24 +0100 schrieb rpertaub@gmail.com
<rpertaub@gmail.com>:

> Hello,

> I am doing some image analysis, and my image consists of several
> bright spots that I need to detect. I was able to write a program that
> would do just that...find the pixels that are larger than a threshold,
> group close pixels together and label different blobs as different
> spots by marking a 'plus sign' on the spot. Except, it does not 'see'
> all the spots, and lowering the threshold results in 'seeing' spots
> that are not there. Therefore, I am considering some filtering that I
> need to do to my background as it is not uniform and was suggested to
> perform a gauss 1d or 2d to the background to subtract it (and exclude
> the spots as I do that), and then see if i can 'see' all the spots....
>
> I am not sure how to do a gauss fit to background though...any
> suggestions?
> Thanks!
> rp
>

--
Erstellt mit Operas revolutionï¿œrem E-Mail-Modul: http://www.opera.com/m2/

Subject: Re: Gaussian Fit to background of image for subtraction
Posted by Brian Larsen on Wed, 07 Feb 2007 17:05:41 GMT
View Forum Message <> Reply to Message

In the past I have taken a different tack at this. I knew something about the shape and size of my blobs and about the character of the background. This gives all sorts of advantages in the background removal process and the blog detection.

In my example the blobs were cells in a microscope picture and so they have a definite character, the camera was really bad making signal to noise like 1.2 or so. I needed to find the centers of the cells. I accomplished this by a Gaussian convolution over the image since the cell look kinda Gaussian and background did not.

```
; convolve the data with a gaussian kernel to look FOR gaussian like
; things cells are close enough for this
; Simple Gaussian kernel
kernel = [$
  [ 1,  8, 15,  8, 1], $
  [ 8, 63,127, 63, 8], $
  [15,127,255,127,15], $
  [ 8, 63,127, 63, 8], $
  [ 1,  8, 15,  8, 1]]
result = CONVOL( dat1, kernel, 4 )
```

This has the affect of making the cells really bright and the
background really dim. I could then subtract the background at 2
sigma.
bkgd = mean(result, /nan)
bkgd_std = stddev(result, /nan)
;; all the noise should be less than mean+2stddev
result -= (bkgd+2*bkgd_std)

Leaving pretty close to just the cells.

Then let label_region do all the work:
mask = a ge 150
;; this names connected regions 0, 1, 2
regions = label_region(mask)
ind = where(regions eq 1)
;; find the center
center = [mean(ind mod 256), mean(ind / 256)]

and bang I had the centers really well. This seemed to work without
fail on these images. Could be worth a look.

Brian

---------- ---------
Brian A. Larsen
Dept. of Physics
Space Science and Engineering Lab (SSEL)
Montana State University - Bozeman
Bozeman, MT 59717

On Feb 6, 2:47 pm, "Karsten Rodenacker" <karsten.rodenac...@gsf.de>
wrote:
> You could also use morphological operations. E.G. to detect your blobs
> apply a morph_tophat and an appropriate threshold. kernel or structuring
> element should be slightly larger than your blobs. tophat consits of a
> morphological smoothing (open) to generate so to say the background which
> is than subtracted from the original.
> Tophat is relatively unknown but surprisingly effective.
>
> Regards
> karsten
>
> Am Tue, 06 Feb 2007 18:25:24 +0100 schrieb rpert...@gmail.com

> <rpert...@gmail.com>:
>
>
>
>> Hello,
>> I am doing some image analysis, and my image consists of several
>> bright spots that I need to detect. I was able to write a program that
>> would do just that...find the pixels that are larger than a threshold,
>> group close pixels together and label different blobs as different
>> spots by marking a 'plus sign' on the spot. Except, it does not 'see'
>> all the spots, and lowering the threshold results in 'seeing' spots
>> that are not there. Therefore, I am considering some filtering that I
>> need to do to my background as it is not uniform and was suggested to
>> perform a gauss 1d or 2d to the background to subtract it (and exclude
>> the spots as I do that), and then see if i can 'see' all the spots....
>
>> I am not sure how to do a gauss fit to background though...any
>> suggestions?
>> Thanks!
>> rp
>
> --
> Erstellt mit Operas revolutionärem E-Mail-Modul:http://www.opera.com/m2/

Subject: Re: Gaussian Fit to background of image for subtraction
Posted by Karsten Rodenacker on Wed, 07 Feb 2007 22:00:28 GMT
View Forum Message <> Reply to Message

Linear operation have the disadvantage to do always something. The
problems with your method occur with neighbored cells and cells with
varying intensity profile. Or think about a ramp intensity background with
cells sitting on it. You might experience surprising results.
As long you need only a cell centre and not an exact mask of the cell
there might be no problem. However, take a look to the non-linear
operators.
KR

Am Wed, 07 Feb 2007 18:05:41 +0100 schrieb Brian Larsen
<balarsen@gmail.com>:

> In the past I have taken a different tack at this. I knew something
> about the shape and size of my blobs and about the character of the
> background. This gives all sorts of advantages in the background
> removal process and the blog detection.
>
> In my example the blobs were cells in a microscope picture and so they
> have a definite character, the camera was really bad making signal to

> noise like 1.2 or so. I needed to find the centers of the cells. I
> accomplished this by a Gaussian convolution over the image since the
> cell look kinda Gaussian and background did not.
>
> ; convolve the data with a gaussian kernel to look FOR gaussian like
> ; things cells are close enough for this
> ; Simple Gaussian kernel
> kernel = [$
> [1, 8, 15, 8, 1], $
> [8, 63,127, 63, 8], $
> [15,127,255,127,15], $
> [8, 63,127, 63, 8], $
> [1, 8, 15, 8, 1]]
> result = CONVOL(dat1, kernel, 4)
>
> This has the affect of making the cells really bright and the
> background really dim. I could then subtract the background at 2
> sigma.
> bkgd = mean(result, /nan)
> bkgd_std = stddev(result, /nan)
> ;; all the noise should be less than mean+2stddev
> result -= (bkgd+2*bkgd_std)
>
> Leaving pretty close to just the cells.
>
> Then let label_region do all the work:
> mask = a ge 150
> ;; this names connected regions 0, 1, 2
> regions = label_region(mask)
> ind = where(regions eq 1)
> ;; find the center
> center = [mean(ind mod 256), mean(ind / 256)]
>
> and bang I had the centers really well. This seemed to work without
> fail on these images. Could be worth a look.
>
> Brian
>
> ---------- ---------
> Brian A. Larsen
> Dept. of Physics
> Space Science and Engineering Lab (SSEL)
> Montana State University - Bozeman
> Bozeman, MT 59717
>
>
>
>

>
> On Feb 6, 2:47 pm, "Karsten Rodenacker" <karsten.rodenac...@gsf.de>
> wrote:
>> You could also use morphological operations. E.G. to detect your blobs
>> apply a morph_tophat and an appropriate threshold. kernel or structuring
>> element should be slightly larger than your blobs. tophat consits of a
>> morphological smoothing (open) to generate so to say the background
>> which
>> is than subtracted from the original.
>> Tophat is relatively unknown but surprisingly effective.
>>
>> Regards
>> karsten
>>
>> Am Tue, 06 Feb 2007 18:25:24 +0100 schrieb rpert...@gmail.com
>> <rpert...@gmail.com>:
>>
>>
>>
>>> Hello,
>>> I am doing some image analysis, and my image consists of several
>>> bright spots that I need to detect. I was able to write a program that
>>> would do just that...find the pixels that are larger than a threshold,
>>> group close pixels together and label different blobs as different
>>> spots by marking a 'plus sign' on the spot. Except, it does not 'see'
>>> all the spots, and lowering the threshold results in 'seeing' spots
>>> that are not there. Therefore, I am considering some filtering that I
>>> need to do to my background as it is not uniform and was suggested to
>>> perform a gauss 1d or 2d to the background to subtract it (and exclude
>>> the spots as I do that), and then see if i can 'see' all the spots....
>>
>>> I am not sure how to do a gauss fit to background though...any
>>> suggestions?
>>> Thanks!
>>> rp
>>
>> --
>> Erstellt mit Operas revolutionï¿œrem E-Mail-Modul:http://www.opera.com/m2/
>
>

--
Erstellt mit Operas revolutionï¿œrem E-Mail-Modul: http://www.opera.com/m2/