Subject: particle detection - a way to speed up things?
Posted by Ingo von Borstel on Wed, 28 Nov 2007 13:40:00 GMT
View Forum Message <> Reply to Message

Hi there,

I run an algorithm which tries to detect particles on image sequences. The most time consuming operation (more than half of the processing time) is to find the centre of all detected particles. I calculate the centre of each particle separatedly by supplying an image where only the i-th particle is present to "schwerpunkt2". Is there a faster way to do this? I put the outline of the calling routine and "schwerpunkt2" below for reference.

Best regards,
Ingo

```
PRO schwerpunkt2, image, xpos, ypos, img_total=img_total, dims=dims

; Procedure returns the centre of weight (xpos, ypos) of a
; 2D-array (image). In order to speed up calculation, the total of the
; supplied 2D array (img_total) and its dimensions (dims) can be
; supplied, should they already be known.

IF SIZE(image,/N_DIMENSIONS) NE 2 THEN BEGIN
   MESSAGE, "Number of dimensions must be exactly two.",/CONTINUE
   xpos = 0
   ypos = 0
   RETURN
ENDIF

IF NOT KEYWORD_SET(dims) THEN $
  dims = SIZE(image,/DIMENSIONS)

IF NOT KEYWORD_SET(img_total) THEN $
  img_total = TOTAL(image)

xs = dims[0] & ys = dims[1]

xvec = indgen(xs)
yvec = indgen(ys)

xpos = TOTAL(xvec * TOTAL(image,2))/img_total
ypos = TOTAL(yvec * TOTAL(image,1))/img_total

END ; schwerpunkt2
```

```
PRO detect_particles, filename, area, pos, brightness, minintbright,
maxsize, minsize
image = READ_IMAGE(filename)
 ; Now do proper noise reduction and particle enhancement using edge
detection, and filtering with proper structuring elements such that
particles most probable don't overlap anymore and are separated by zeros
in the image.
gray_image = enhance_image(image)

particle_image =
 WATERSHED(255-gray_image,CONNECTIVITY=8,/LONG,nregions=n_par ticles)
dims =SIZE(particle_image,/DIMENSIONS)

pos = DBLARR(n_particles,2)
area = DBLARR(n_particles)
brightness = DBLARR(n_particles)

 ; Now determine properties of all detected particles
FOR i=0,n_particles-1 DO BEGIN
 bin_thisparticle = particle_image EQ i
 gray_thisparticle = particle_image * bin_thisparticle
 xpos = 0 & ypos = 0
 area[i-1] = TOTAL(bin_thisparticle)
 brightness[i-1] = TOTAL(gray_thisparticle)
 schwerpunkt2, gray_thisparticle, xpos,
ypos,img_total=brightness[i-1],dims=dims
 pos[i-1,0] = xpos
 pos[i-1,1] = ypos
 IF area[i-1] LT minsize OR area[i-1] GT maxsize OR brightness[i-1] LT
minintbright THEN BEGIN
  real_particle[i-1] = 0
  gray_image *= particle_image NE i
 ENDIF
ENDFOR
END ;detect_particles
```

--
Ingo von Borstel <newsgroups@planetmaker.de>
Public Key: http://www.planetmaker.de/ingo.asc

If you need an urgent reply, replace newsgroups by vgap.

Subject: Re: particle detection - a way to speed up things?
Posted by Vince Hradil on Fri, 30 Nov 2007 14:02:28 GMT
View Forum Message <> Reply to Message

On Nov 30, 2:22 am, Ingo von Borstel <newsgro...@planetmaker.de>
wrote:
> Hi,
>
>> If I'm reading your program correctly, you have a big image consisting
>> (presumably) of
>> a lot of empty space and numerous particles that you have identified
>> in some way.
>
> Yes, correct. Black image with an arbitrary amount (<= 1000) of small
> blobs with a value different than zero; those blobs numbered using
> either label_region or watershed.
>
>> My guess is that the main reason your program is slow is that for each
>> particle you
>> are summing over the entire image.
>
> Probably. Half the time is consumed by the determination of the position
> for each particle in the overall image.
>
>> I can see two ways to speed things up:
>> 1) Create subsets of the image for each particle and only sum only
>> over the subset containing the particle.
>
> In order to calculate the absolute position of a particle, I need to
> know where on an image it resides. I think I'll loose this information
> with this approach.
>
>> 2) Use something like HISTOGRAM or a multi-dimensional historgam with
>> the REVERSE_INDICES keyword (or equivalent)
>> to get the indices associated with each particle and sum over those.
>> The histogram command would be applied to your
>> particle_image field with a binsize of 1 and starting at 0. See
>> http://www.dfanning.com/tips/histogram_tutorial.html
>> for ideas on how to approach this problem using histograms.
>
> The latter might have potential. I'll give it a try and report back then.
>
> Thanks a lot for your input.
>
> Cheers,
> Ingo
>
> --
> Ingo von Borstel <newsgro...@planetmaker.de>
> Public Key:http://www.planetmaker.de/ingo.asc
>
> If you need an urgent reply, replace newsgroups by vgap.

Have you looked at the LABEL_REGION example in the help?

Subject: Re: particle detection - a way to speed up things?
Posted by dcleon@gmail.com on Fri, 30 Nov 2007 16:02:30 GMT
View Forum Message <> Reply to Message

>> In order to calculate the absolute position of a particle, I need to
>> know where on an image it resides. I think I'll loose this information
>> with this approach.
>

Not necessarily. You just pass the position of the corner of the
subset to
the routine that calculates the center, or you could add the position
of
the corner of the subset to the value returned by your routine that
computes
the position of the particle relative to the image subset. Its linear
so the
position of the center of the image with respect to subset + the
position of
the subset is the same as the position of the particle relative to the
whole image.

This approach would be easier to implement than the HISTOGRAM approach
I mentioned.

cheers
dave

Subject: Re: particle detection - a way to speed up things?
Posted by Dan Larson on Mon, 03 Dec 2007 16:30:26 GMT
View Forum Message <> Reply to Message

Hi,

Depending on the types of blobs you are detecting, FFT filtering can
often work much faster for the initial segmentation. My approach for
particle detection:
1. spatial bandpass to identify the regions of interest.
2. cut out the regions of interest to pass to the fitting routine (as
David suggested).
3. find the center using the algorithm of choice.

This last step can be any number of things, but if you choose a centroid approach, these lines work well for me, although I think they are similar to what you have:

```
a=size(pic)
x_dim = a[1]
y_dim = a[2]
array=lindgen(x_dim, y_dim)
xarr=array mod x_dim
yarr=array/x_dim
sum=double(total(pic))
xcenter=total(xarr*pic)/sum
ycenter=total(yarr*pic)/sum
return, [xcenter, ycenter, sum]
end
```

For fourier filtering, I particulary like a routine from Crocker and Grier: www.physics.emory.edu/~weeks/idl/kit/bpass.pro

If you are interested in detecting and fitting diffraction-limited spots in a fluorescence microscope, I have software which is specifically designed for that purpose. I could send that to you directly if you want to try it out.

best,
dan
