Subject: Re: faster convol on local subsets?
Posted by ben.bighair on Mon, 05 Dec 2011 01:19:13 GMT
View Forum Message <> Reply to Message

On Dec 4, 7:37 pm, Andre <note....@gmail.com> wrote:
> Hello experts,
>
> Maybe somebody has an easy solution for this?
> I have a 2D array (img) and want the filter response from kernels that vary according to the image position. In a second array (loc, same dimensions as img) I have the information which kernel should be used at each pixel. My current approach is to first convolve the full image with the j-th kernel and take the response only at the positions with the current j indexed in the loc array:
>
```
>  for j=0, n do begin
>       kernel=kernel_store[*,*,j]
>       response_temp = convol(img, kernel, /edge_zero, /NAN)
>       index=where(loc eq j)
>       if (index[0] gt -1)then response[index]=response_temp[index]
>  endfor
```
>
> I works fine, but it is relatively slow and I wonder if there is a smarter (faster) to apply only the convolutions that are really needed?
>
> Thanks in advance for any help!

Hi,

Since the value of loc doesn't change, you might try precomputing the
values of index in loc for each step, j.  That would save a two full
scans of you loc array at each iteration (one for loc eq j and one for
WHERE).  Instead, use HISTOGRAM with its REVERSE_INDICES to get them
sorted and tagged.  Then for each iteration step use David Fanning's
REVERSEINDICES to grab the correct indices in loc.

So, it might look like...

```
locHist = HISTOGRAM(loc, min = 0, REVERSE_INDICES = ri)

for j=0, n do begin
      kernel=kernel_store[*,*,j]
      response_temp = convol(img, kernel, /edge_zero, /NAN)
      index = REVERSEINDICES(ri, j, COUNT =
count)
      if (count GT 0)then response[index]=response_temp[index]
endfor
```

Cheers,

Ben

http://www.idlcoyote.com/programs/reverseindices.pro

---