Subject: Re: Array Tiling - The IDL Way
Posted by Michael Galloy on Mon, 12 Mar 2012 18:54:59 GMT
View Forum Message <> Reply to Message

On 3/12/12 11:31 AM, Percy Pugwash wrote:
> I have a large array that I'd like to break up into tiles (square tiles of side T). I would like to have those tiles in the form of a stack, such that my array goes from dimensions [Nx*T,Ny*T] to [T,T,Nx*Ny].
>
> Is there any way I can do this using only functions like REFORM and TRANSPOSE and no for-loops? Ideally I'd like to do it in-place too. I've been racking my brain for a nice IDLesque way to do this, but no luck thus far...
>
> P

I think this is what you are trying to do:

```
IDL> a = indgen(10, 10)
IDL> b = transpose(reform(a, 2, 5, 2, 5), [1, 3, 0, 2])
```

This will break the image a into 5 tiles of size 2 in each dimension. So you can do the following, e.g., to retrieve the tile at (4, 4):

```
IDL> print, reform(b[4, 4, *, *])
      88      89
      98      99
```

Mike
--
Michael Galloy
www.michaelgalloy.com
Modern IDL, A Guide to Learning IDL: http://modernidl.idldev.com
Research Mathematician
Tech-X Corporation

Subject: Re: Array Tiling - The IDL Way
Posted by rogass on Mon, 12 Mar 2012 19:28:01 GMT
View Forum Message <> Reply to Message

On 12 Mrz., 18:31, Percy Pugwash <percy.pugw...@gmail.com> wrote:
> I have a large array that I'd like to break up into tiles (square tiles of side T). I would like to have those tiles in the form of a stack, such that my array goes from dimensions [Nx*T,Ny*T] to [T,T,Nx*Ny].
>
> Is there any way I can do this using only functions like REFORM and TRANSPOSE and no for-loops? Ideally I'd like to do it in-place too. I've been racking my brain for a nice IDLesque way to do this, but no luck thus far...

>
> P

Hi,
just have a try with this. Note that your tile size must be an integer
part of the original image.

```
function cr_tile,im,wx,wy
s = size(im,/dim)
nel = n_elements(im)/(wx*wy)
ind = rebin((lindgen(wx,wy) mod wx) + $
            rebin(lindgen(1,wy)*s[0],wx,wy,/sample),wx,wy,nel,/
sample)
ind     += rebin(reform((lindgen(s))[0:*:wx,0:*:wy],1,1,nel,/
over),wx,wy,nel)
return,im[temporary(ind)]
end
```

Cheers

CR

Subject: Re: Array Tiling - The IDL Way
Posted by rogass on Mon, 12 Mar 2012 19:32:36 GMT
View Forum Message <> Reply to Message

On 12 Mrz., 19:54, Michael Galloy <mgal...@gmail.com> wrote:
> On 3/12/12 11:31 AM, Percy Pugwash wrote:
>
>> I have a large array that I'd like to break up into tiles (square tiles of side T). I would like to have those tiles in the form of a stack, such that my array goes from dimensions [Nx*T,Ny*T] to [T,T,Nx*Ny].
>
>> Is there any way I can do this using only functions like REFORM and TRANSPOSE and no for-loops? Ideally I'd like to do it in-place too. I've been racking my brain for a nice IDLesque way to do this, but no luck thus far...
>
>> P
>
> I think this is what you are trying to do:
>
>   IDL> a = indgen(10, 10)
>   IDL> b = transpose(reform(a, 2, 5, 2, 5), [1, 3, 0, 2])
>
> This will break the image a into 5 tiles of size 2 in each dimension. So
> you can do the following, e.g., to retrieve the tile at (4, 4):
>

> IDL> print, reform(b[4, 4, *, *])
> 88 89
> 98 99
>
> Mike
> --
> Michael Galloywww.michaelgalloy.com
> Modern IDL, A Guide to Learning IDL:http://modernidl.idldev.com
> Research Mathematician
> Tech-X Corporation

Uh, it's a 'little' bit more efficient than my one .... :)

Cheers

CR

Subject: Re: Array Tiling - The IDL Way
Posted by Percy Pugwash on Sat, 17 Mar 2012 16:13:54 GMT
View Forum Message <> Reply to Message

Thanks for the suggestions.

Unfortunately in this case it seems that the nested for-loop method is actually much faster! I think the reason is that the arrays I'm using are so large (gigabytes in size) that the amount of memory manipulation involved in the TRANSPOSE operation dwarfs the time spend on running a for-loop.

Ah well.

P

On Monday, 12 March 2012 18:54:59 UTC, Mike Galloy wrote:
> On 3/12/12 11:31 AM, Percy Pugwash wrote:
>> I have a large array that I'd like to break up into tiles (square tiles of side T). I would like to have those tiles in the form of a stack, such that my array goes from dimensions [Nx*T,Ny*T] to [T,T,Nx*Ny].
>>
>> Is there any way I can do this using only functions like REFORM and TRANSPOSE and no for-loops? Ideally I'd like to do it in-place too. I've been racking my brain for a nice IDLesque way to do this, but no luck thus far...
>>
>> P
>
> I think this is what you are trying to do:
>
> IDL> a = indgen(10, 10)
> IDL> b = transpose(reform(a, 2, 5, 2, 5), [1, 3, 0, 2])

>
> This will break the image a into 5 tiles of size 2 in each dimension. So
> you can do the following, e.g., to retrieve the tile at (4, 4):
>
> IDL> print, reform(b[4, 4, *, *])
> 88 89
> 98 99
>
> Mike
> --
> Michael Galloy
> www.michaelgalloy.com
> Modern IDL, A Guide to Learning IDL: http://modernidl.idldev.com
> Research Mathematician
> Tech-X Corporation
