On Mar 10, 7:56 am, FÖLDY Lajos <fo...@rmki.kfki.hu> wrote:
> On Mon, 9 Mar 2009, rtk wrote:
>> On Mar 8, 6:17 pm, Jye <jye.sm...@gmail.com> wrote:
>>>   if i eq 0 then All_Images = image_slice else All_Images =
>>> [[[All_Images]], [[image_slice]]]
>
>> Forget too much Java, which, I agree, is a bad thing. I'd say you
>> have been doing too much Matlab programming. The vast majority of
>> your problem lies with the line above. If I run this code on my box:
>
>> pro ttt0
>>   s = systime(1)
>>   for i=0,899 do begin
>>   im = fix(dist(256))
>>   if (i eq 0) then begin
>>   all = im
>>   endif else begin
>>   all = [[[all]], [[im]]]
>>   endelse
>>   endfor
>>   print, 'runtime = ', systime(1)-s
>>   help, all
>> end
>
>> It takes 93 seconds. However, if I run this code:
>
>> pro ttt2
>>   all = intarr(256,256,900)
>>   s = systime(1)
>>   for i=0,899 do begin
>>   im = fix(dist(256))
>>   all[*,*,i] = im
>>   endfor
>>   print, 'runtime = ', systime(1)-s
>>   help, all
>> end
>
>> it takes 1.6 seconds. The only difference is that the output array is
>> allocated once, instead of 900 times with 899 ever-increasing array
>> copies to boot. Since you know the number of file in advance,
>> allocate All_Images and just assign into it.
>
>> Ron
>

```
>  You can get some more speedup by replacing 'all[*,*,i] = im' with
>  'all[0,0,i] = im':
>
>  ; cut here, ttt2.pro
>  pro ttt2
>      all = intarr(256,256,900)
>      im = fix(dist(256))
>
>      s = systime(1)
>      for i=0,899 do begin
>          all[*,*,i] = im
>      endfor
>      print, 'runtime = ', systime(1)-s
>
>      s = systime(1)
>      for i=0,899 do begin
>          all[0,0,i] = im
>      endfor
>      print, 'runtime = ', systime(1)-s
>  end
>  ; cut here
>
>  IDL> ttt2
>  % Compiled module: TTT2.
>  % Compiled module: DIST.
>  runtime =      0.20675302
>  runtime =      0.047688007
>
>  (This is not the bottleneck for the original problem, though.)
>
>  regards,
>  lajos- Hide quoted text -
>
>  - Show quoted text -
```

Hey I noticed you're also rotating each slice individually before
adding it to the group. It may work faster if you rotate the full
volume in one step at the end.
Of course IDL doesn't have anything to do this. But you can find a
link to the code that does it here. http://www.dfanning.com/math_tips/rotvolume.html.

Try building your volume first then rotating the whole thing in one
step. May or may not save you some time.