Subject: Re: Rotate volumes
Posted by marc schellens[1] on Wed, 19 Sep 2001 11:19:16 GMT
View Forum Message <> Reply to Message

Martin Downing wrote:
>
> Marc,
>
> Thats a neat way of doing it - just watch out for the effect of wrap-around
> if you use translate (caused by the use of SHIFT) on matching two images.
> The method I posted actually uses linear interpolation (I misread the help!)
> and is a little quicker (20sec c.f. 29sec) for a general rotation on a PIII
> 800). The rotation orders are not the same but could easily be made so.
>
> cheers
>
> Martin
>
I used the attached test routine (just quickly put together for this
test)
because I remembered, that then plane-wise method was faster for me
(IDL 5.4.1, Linux mandrake 8.0):

156^3 bytarr:
8s vs. 11s

256^3 bytarr:
35s vs. 184s

As I mentioned, the bigger the array, the larger your index list.
In the 256^3 case, my machine started swapping (with over 600MB free
memory!)
which explains the huge time-difference in the second case.
With the shift you are right. For MRI images its most often ok to use
nevertheless,
as the head is surrounded by a nonused black frame.
I assume you use windows. Interesting, that the performance of the two
methods
is so much OS-dependent.

cheers,
:-) marc

```
function undefined,i
return,n_elements(i) eq 0
```

```
end

function transform2, im, rotation = phi,$
scale=scale,translate=translate, centre_rot=centre_rot
; translate an image volume using interplote
s = size(im)
; for clarity:
sx=s(1)
sy=s(2)
sz=s(3)
if undefined(phi) then phi =[0,0,0]
if undefined(centre_rot) then centre_rot
=[(sx-1)/2.0,(sy-1)/2.0,(sz-1)/2.0]
if undefined(translate) then translate =[0,0,0]
if undefined(scale) then scale =[1,1,1]

print,'X...'
if phi[0] ne 0.0 then begin
    for x=0,sx-1 do begin

 im[x,*,*]=rot(/INTERP,reform(im[x,*,*],sy,sz),-phi[0],MISSIN G=0)
    endfor
endif
print,'Y...'
if phi[1] ne 0.0 then begin
    for y=0,sy-1 do begin

 im[*,y,*]=rot(/INTERP,reform(im[*,y,*],sx,sz),phi[1],MISSING =0)
    endfor
endif
print,'Z...'
if phi[2] ne 0.0 then begin
    for z=0,sz-1 do begin
        im[*,*,Z]=rot(/INTERP,im[*,*,Z],-phi[2],MISSING=0)
    endfor
endif

return,im
end

function transform1, im, rotation = rot,$
scale=scale,translate=translate, centre_rot=centre_rot
; translate an image volume using interplote
s = size(im)
; for clarity:
sx=s(1)
sy=s(2)
sz=s(3)
```

```
if undefined(rot) then rot =[0,0,0]
if undefined(centre_rot) then centre_rot
=[(sx-1)/2.0,(sy-1)/2.0,(sz-1)/2.0]
if undefined(translate) then translate =[0,0,0]
if undefined(scale) then scale =[1,1,1]
;generate image coordinates
i = lindgen(sx*sy*sz) ; temp array = vector indices
coords = [ [i mod sx],[(i / sx) mod (sy)],[i
/(sx*sy)],[replicate(1,sx*sy*sz)]]
; generate transform (or add your own)
t3d, /reset
t3d,trans= -centre_rot
t3d, rot=rot
t3d, trans= centre_rot + translate
t3d, scale=scale
; calc new sample positions of voxels
coords = coords#!p.t
; use these to interpolate voxels (note this is only SAMPLED)
im     = reform( interpolate(im, coords(*,0), coords(*,1), coords(*,2),$
missing=0),sx,sy,sz)
return, im
end


pro test

d=156
a=1.0

b=byte(indgen(d,d,d))

s=systime(1)
print,'ix'
r=transform1(b,rot=[a,a,a])
print,systime(1)-s


b=byte(indgen(d,d,d))

s=systime(1)
print,'planes'
r=transform2(b,rot=[a,a,a])
print,systime(1)-s

end
```