

---

Subject: Re: arbitrary rotation of 3-d arrays  
Posted by [morisset](#) on Fri, 11 Jun 1999 07:00:00 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

The use of t3d will perform transformation on a coordinate cube, but will not 'rotate' the datas. It's like when you want to compare 2 images with one turned in respect to the other one. Or if you want to make a projection using total, but on an axis other than x or y. Then you have to use the ROT idl function to perform interpolation.

I wrote a turn3d code (see below) that perform rotation of the datas in a 3D cube. It uses ROT slide by slide.  
It (seems to ;-)) works also with vector fields.  
Hope it helps, send (bug) reports to (witout blanck):  
morisset @ astrsp-mrs.fr

```
-----Cut here -----
function turn_3d,a_in,x_angle,y_angle,z_angle,RESIZE = resize,$
  CONSERV = conserv,VERBOSE = verbose,HELP=help,_extra= _extra,vect=vect
;+
; NAME:
; turn_3d
;
; CALLING SEQUENCE:
; result = turn_3d(a,x_angle,y_angle,z_angle)
;
; PURPOSE:
; Rotate a 3D array. It applys the ROT IDL function to each
; 2D sub_array of A. The computation is done in a 50% bigger
; cube to assure that nothing will be losed.
; If A is a structure, apply recurively turn_3d on all the
; tags of the structure.
;
; INPUT PARAMETERS:
; A = The 3D array to be rotated. This array may be of any type
; exepcted string. Can be a structure.
; X, Y, Z_ANGLE =
; 3 angles of rotation in degrees CLOCKWISE.
; WARNING: in case of multiple rotation, the order is Z, X and Y.
; KEYWORDS:
; VECT : Setting this keyword if A is a 3D vector field
; _extra will be passed to ROT:
;
; RESIZE: Setting this keyword to resize the result to the maximum
; size (x,y or z-one) of A. The resizing is NOT a rebining,
; it extracts a 3D sub-array of the big 3D array in wich
; the computation is done.
; If A is a structure, RESIZE is set.
```

```

; CONSERVE: Setting this keyword to assure that
; total(result) = total(A).
;
; VERBOSE: Setting this keyword will print the ratio of the
; sizes of the input array and the result. Works only if
; RESIZE not set. If A is a structure, will say what is
; being rotated.
; HELP: print the calling sequence
;
; LIMITATIONS: They are those of ROT... For small dimensions arrays,
; a rotation of +10deg followed by a rotation of -10deg will NOT
; give you back the input data.
;
; BUGS: If A is a structure of arrays NON cubics (s(1) = s(2) = s(3),
; then it crash!
;
; AUTHOR
; Christophe MORISSET, 1997. morisset @ iagusp.usp.br
;
; HISTORY:
; 15-9-97 Post for me by D. Fanning on comp.lang.idl-pwwave
; 19-9-97 Add the HELP keyword
; 26-9-97 Add the possibiliy for A to be a structure
; Suppretion of CUBIC keyword
; 28-4-98 pass all the _extra to rot
; 19-1-99 add the vector facility (and keyword)
;-

```

```

if keyword_set(help) then begin
  print,'function turn_3d,a,x_angle,y_angle,z_angle,INTERP =
interp,'+ $
  'MISSING = missing,PIVOT = pivot, RESIZE = resize,'+ $
  'CONSERV = conserv,VERBOSE = verbose,HELP=help,vect=vect'
  return,0
endif

```

```

if (size(a_in))(n_elements(size(a_in))-2) eq 8 then begin ; a is a
structure
  if keyword_set(verbose) then print,' turn_3d: structure'
  b = a_in
  names = tag_names(a_in)
  for i = 0,n_tags(a_in)-1 do begin
    if keyword_set(verbose) then print,'turning ',names(i)
    b.(i) = turn_3d(a_in.(i),x_angle,y_angle,z_angle, $
      _extra=_extra,$
      RESIZE = 1,CONSERV = conserv)
  endfor

```

```

return,b

endif                                ; case a is a structure

if keyword_set(vect) then begin
  if keyword_set(verbose) then print,' turn_3d: vector'
  a_out = a_in

  if z_angle ne 0. then begin
    a_tmp = a_out
    t3d,/reset,rotate=[0.,0.,z_angle]
    for i= 0,2 do a_out[*,*,i] = $
      a_tmp[*,*,0] * !p.t[i,0] + $
      a_tmp[*,*,1] * !p.t[i,1] + $
      a_tmp[*,*,2] * !p.t[i,2]
    for i= 0,2 do a_out[*,*,i] = turn_3d(a_out[*,*,i], $
0.,0.,z_angle,_extra=_extra, $

resize=resize,verbose=verbose)
endif
if x_angle ne 0. then begin
  a_tmp = a_out
  t3d,/reset,rotate=[x_angle,0.,0.]
  for i= 0,2 do a_out[*,*,i] = $
    a_tmp[*,*,0] * !p.t[i,0] + $
    a_tmp[*,*,1] * !p.t[i,1] + $
    a_tmp[*,*,2] * !p.t[i,2]
  for i= 0,2 do a_out[*,*,i] = turn_3d(a_out[*,*,i], $
x_angle,0.,0.,_extra=_extra, $

resize=resize,verbose=verbose)
endif
if y_angle ne 0. then begin
  a_tmp = a_out
  t3d,/reset,rotate=[0.,y_angle,0.]
  for i= 0,2 do a_out[*,*,i] = $
    a_tmp[*,*,0] * !p.t[i,0] + $
    a_tmp[*,*,1] * !p.t[i,1] + $
    a_tmp[*,*,2] * !p.t[i,2]
  for i= 0,2 do a_out[*,*,i] = turn_3d(a_out[*,*,i], $
0.,y_angle,0.,_extra=_extra, $

resize=resize,verbose=verbose)
endif
return,a_out

```

```

endif                                ; case a is a vector (4D)

if keyword_set(verbose) then print, ' turn_3d: simple case'
a = reform(a_in)
if (size(a))(0) ne 3 then stop, ' A must be 3D'

x_size = (size(a))(1)
y_size = (size(a))(2)
z_size = (size(a))(3)

max_size = x_size > y_size > z_size

; let's do a 50% larger 3D array containing the input 3D array at his
"center"

new_size = fix(max_size*1.5) + 1
b = congrid(a*0.,new_size,new_size,new_size)

b[(new_size-x_size)/2:(new_size-x_size)/2+x_size-1,$
  (new_size-y_size)/2:(new_size-y_size)/2+y_size-1,$
  (new_size-z_size)/2:(new_size-z_size)/2+z_size-1] = a

; Z-rotation
  if z_angle ne 0. then begin
    for z = 0,new_size-1 do b[:,*,z] =
rot(reform(b[:,*,z]),z_angle,$
                                     _extra=_extra)
  endif

; X-rotation
  if x_angle ne 0. then begin
    for x = 0,new_size-1 do b[x,*,*] =
rot(reform(b[x,*,*]),x_angle,$
                                     _extra=_extra)
  endif

; Y-rotation
  if y_angle ne 0. then begin
    for y = 0,new_size-1 do b[:,y,*] =
rot(reform(b[:,y,*]),-y_angle,$
                                     _extra=_extra)
  endif

if keyword_set(resize) then b = $
b[(new_size-x_size)/2:(new_size-x_size)/2+x_size-1,$
  (new_size-y_size)/2:(new_size-y_size)/2+y_size-1,$

```

```
(new_size-z_size)/2:(new_size-z_size)/2+z_size-1] $  
else if keyword_set(verbose) then $  
  print,' Size changed by: ',float(new_size) / float(max_size)  
  
  if keyword_set(conserv) then b = b / total(b) * total(a)  
  
  return,b  
  
end
```

Sent via Deja.com <http://www.deja.com/>  
Share what you know. Learn what you don't.

---