Subject: Re: DICOM directory
Posted by Anne[1] on Thu, 16 Apr 2009 15:21:07 GMT
View Forum Message <> Reply to Message

I'm also posting the procedure that reads the images and sorts them into [x,y,z,time] format. Sorry about pasting this - I don't know if it's possible to add attachments in this news group,
Anne

```
; docformat = 'rst'
;
;+
; Reads in dicom series and arranges into 3D or 4d format as
appropriate.
; Uses dialog_pickfile and allows multiple files to be selected.
; This is based on the slice location and also allows trigger time to
be read.
; At present does not check that images all belong to same series
; but assumes that files have already been sorted into separate
directories and that user selects all files.
; If a slice is missing in a 4d sequence then blank images are
inserted at appropriate points
; Returned image is in form [x,y,z,time]
;-
;+
; :Keywords:
; path : in, optional, type=string
;      can pass a string containing the path name to start searching
from
; fNames : in, optional, type=stringArray
;       ordered list of filenames. This could be used to accept sorted
output from dicomsortfiles
; info : out, optional, type=structure
;     this is  a structure containing slice information
;       info: {
;               xPixelSize;
;               yPixelSize;
;               sliceThickness;
;               slicelocation (array);
;               sliceTime (array);
;               time increment;
;               origin (array) -->added by Grace
;              }
; getpath : out, optional, type=string
;      returns the path string. Useful if need to make another call to
same directory
; title : in, optional, type=string
;      String is used as a title for the dialog_pickfile box.
```

```
; list :
; :Examples: myImage = read4ddicom(p='~\home',info=info)
; :Categories: DICOM, fileIO
;
; :History: Written by Anne Martel, http://medbio.utoronto.ca/faculty/martel.html
;     modified Nov 07 to return sorted list of files.
;     Now automatically flips images so origin should be at bottom
left.
;     Modified March 2009 to cope with missing images.
;     Modified April 2009 to take list structure as a keyword
; :todo: need to check what effect flipping image has on origin
;-
function read4dDicom,fNames=fNames, info=sizeParams, $
         path=path, getpath=getpath, $
         title = title, $
         serieslist = serieslist

; There are several mechanisms for selecting file names.
; First check to see if list of fNames has been set.
; If not then check to see if serieslist has been set (this is
generated by the routine dicom_sortfiles)
; If neither of these keywords are set then user is prompted for
filenames

if n_elements(fNames) eq 0 then begin
  nSeries = n_Elements(seriesList)
  if (nSeries eq 0) then begin
     if N_ELEMENTS(title) EQ 0 then begin
         fNames=dialog_pickfile(title='select DICOM images', /
multiple, /must_exist, $
                     path=path,get_path=getpath)
     endif else begin
         fNames=dialog_pickfile(title=title, /multiple, /must_exist, $
                     path=path,get_path=getpath)
     endelse
  endif else begin ; serieslist keyword is set so use this to select
filenames
    ;print list of series available
    for i = 0,nseries - 1 do begin
      tmp=seriesList[i]
      print, strcompress(i), ': ', tmp.patname, ', ',tmp.se_desc
    endfor
    read,'Enter series number:',userSelection
    seriesNo=fix(userSelection)
    fNames = *(seriesList[seriesNo]).file_list
  endelse
endif
```

```
  nFiles=n_elements(fNames)
  imageNo=intarr(nFiles)

  sizeParams={xPixelSize:1.0 , $
      yPixelSize:1.0, $
      sliceThickness:1.0, $
      timeIncrement:0L, $
      sliceLocation:fltarr(nfiles), $
      sliceTime:fltarr(nfiles), $
      ;--> added by Grace
      spaceBetweenSlices: 1.0, $
      origin:fltarr(3) $
      ;--> end of addition
      }

   ; get fixedparameters for first image
   im=obj_new('IDLffDICOM',fNames[0])
   ptrSeriesUID = im->getvalue('0020'x,'000E'x, /no_copy)
   seriesUID = *ptrSeriesUID[0]
   PixelSpacing = im->getvalue('0028'x,'0030'x, /no_copy)
   spacing = strsplit(*PixelSpacing[0],'\',/extract)
   sizeParams.xPixelSize = float(spacing[0])
   sizeParams.yPixelSize = float(spacing[1])
   thickness=im->getvalue('0018'x,'0050'x, /no_copy)
   sizeParams.sliceThickness = *thickness[0]

   ;--> added by Grace
   parameter = im->getvalue('0018'x,'0088'x, /no_copy)
   if ptr_valid(parameter) then sizeParams.spaceBetweenSlices =
*parameter[0]

   parameter = im->getvalue('0020'x,'0032'x, /no_copy)
   if ptr_valid(parameter) then begin
     origin = strsplit(*parameter[0],'\',/extract)
     sizeParams.origin[0] = float(origin[0])
     sizeParams.origin[1] = float(origin[1])
     sizeParams.origin[2] = float(origin[2])
   endif
   ;print, origin
   ;--> addition ends

   ; get variable parameters for first image
   imageID=im->getvalue('0020'x,'0013'x, /no_copy)
   if ptr_valid(imageID) then imageNo[0]=*imageID[0]
   imSlice = im->getvalue('7FE0'x,'0010'x, /no_copy)
   location = im->getvalue('0020'x,'1041'x, /no_copy)
   if ptr_valid(location[0]) then sizeParams.slicelocation[0] = float
```

```
(*location[0])
   time = im->getvalue('0018'x,'1060'x, /no_copy)
   if ptr_valid(time[0]) then sizeParams.sliceTime[0] = float(*time
[0])
   imSlice = im->getvalue('7FE0'x,'0010'x, /no_copy)
   image = rotate(*imslice[0],7)
   Counter = 1
   obj_destroy,im

   ; now read in remaining images - counter will equal number of valid
files
   for i = 1,nFiles-1 do begin
     im=obj_new('IDLffDICOM',fNames[i])
        ptrCurrentUID = im->getvalue('0020'x,'000E'x, /no_copy)
        ; check that images belong to same series
        if *ptrCurrentUID[0] eq seriesUID then begin
           imageID=im->getvalue('0020'x,'0013'x, /no_copy)
           if ptr_valid(imageID) then imageNo[Counter]=*imageID[0]
           imSlice = im->getvalue('7FE0'x,'0010'x, /no_copy)
           location = im->getvalue('0020'x,'1041'x, /no_copy)
           if ptr_valid(location[0]) then sizeParams.slicelocation
[Counter] = float(*location[0])
           time = im->getvalue('0018'x,'1060'x, /no_copy)
           if ptr_valid(time[0]) then sizeParams.sliceTime[Counter] =
float(*time[0])
           imSlice = im->getvalue('7FE0'x,'0010'x, /no_copy)
            image=[[[image]],[[rotate(*imslice[0],7)]]]
            counter = counter + 1
        endif else begin
           tmp = dialog_message(fnames[i]+': file not part of
series')
       endelse
       obj_destroy,im
  endfor

; need to cope with misplaced files
imageOrder = indgen(nFiles)
imageOrder[0:counter-1] = sort(imageNo[0:counter-1])
image = image[*,*,imageOrder[0:counter-1]]
fNames = fNames[imageOrder[0:counter-1]]
sizeParams.slicelocation = sizeParams.slicelocation(imageOrder)
sizeParams.sliceTime = sizeParams.sliceTime(imageOrder)
r=where(sizeparams.slicelocation eq sizeparams.slicelocation
[0],nRepetitions)

nslices = floor(counter/nRepetitions)
imageSize=size(image[*,*,0])
; if number of images is exact multiple of nslices then simply reform
```

```
matrix
; otherwise have to fill in 4d array one slice at a time
if (nRepetitions gt 1) and (nSlices*nRepetitions eq counter) then
begin
    sizeParams.timeIncrement = sizeParams.sliceTime[nSlices] -
sizeParams.sliceTime[0]
    image = reform(temporary(image),imageSize[1],imageSize
[2],nSlices,nRepetitions)
endif else begin
  locations = sizeParams.slicelocation(sort(sizeParams.slicelocation))
  slicePos = locations[uniq(locations)]
  ; now make sure that don't reverse the slice order within volume
  if sizeParams.slicelocation[0] gt sizeParams.slicelocation[1] then
begin
    slicePos=reverse(slicePos)
  endif
  nslices= n_elements(slicePos)
  nRepetitions = ceil(1.0*counter/nSlices)
  tempVol = intarr(imageSize[1],imageSize[2],nSlices,nRepetitions)
  currentIm = 0
  for vol = 0,nRepetitions-1 do begin
    for slice = 0, nslices-1 do begin
        if sizeParams.slicelocation[currentIm] eq slicePos[slice] then
begin
          tempVol[*,*,slice,vol]=image[*,*,currentIm]
          currentIm++
        endif else begin
          print,"missing vol:" , strcompress(vol), ",
slice:",strcompress(slice)
        endelse
    endfor
 endfor
 image=tempVol
endelse

return,image

end
```