Subject: IDL reader for medical images
Posted by Georges Limbert on Tue, 02 Mar 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Dear all,

I would like to know if anybody knows if it exists
a IDL subroutine that can read medical images from
the Visible Human Project. I would like to read the
CT scan images (512 x 512, 16 bit, Header size=3416)
in order to extract the QCT informations.
(Quantitative Computed Tomography)
There is one image per file.
I need to create a text file for each slice (image)
where I would have the voxel positions in space, that
is, the coordinates of the center of "gravity" for
each voxel and the QCT number associated with them.

The text file would look like this:

```
************************************
            xg   |  yg    |  QCT

voxel 1        |
voxel 2        |
- - - - - - - - - - - - - - - - - -
voxel (512x512) |
************************************
```

Any comment or help would be much appreciated.

Best regards.

Georges Limbert

|---|
G e o r g e s L im b e r t (Ph.D. student)
Computational Engineering and Design Centre
Department of Mechanical Engineering
University of Southampton, Highfield
SO17 1BJ Southampton, England
United Kingdom
Phone: +44 (0)1703 59 51 94

Fax: +44 (0)1703 59 32 30
Email: g.limbert@soton.ac.uk
WWW: http://www.soton.ac.uk/~gl4

Subject: Re: IDL reader for medical images
Posted by David Foster on Tue, 09 Mar 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Georges Limbert wrote:
>
> Dear all,
>
> I would like to know if anybody knows if it exists
> a IDL subroutine that can read medical images from
> the Visible Human Project. I would like to read the
> CT scan images (512 x 512, 16 bit, Header size=3416)
> in order to extract the QCT informations.
> (Quantitative Computed Tomography)
> There is one image per file.
> I need to create a text file for each slice (image)
> where I would have the voxel positions in space, that
> is, the coordinates of the center of "gravity" for
> each voxel and the QCT number associated with them.
>
> The text file would look like this:
> ---
>
> ************************************
> xg | yg | QCT
>
> voxel 1 |
> voxel 2 |
> - - - - - - - - - - - - - - - - - -
> voxel (512x512) |
> ************************************
>
> Any comment or help would be much appreciated.
>
> Best regards.
>
> Georges Limbert

Georges -

I've included below my READ_IMG.PRO which can read any medical image so long as it is square, dimensions of 512/256/128/64, raw 8-bit or 16-bit data, and the header is smaller than the image. The code is old and ugly, but it works! I also included READ_IMG.DOC which documents it.

Once you read an image with something like:

```
 status = read_img(filename, image, dimension, header)
```

You can write out the values to a text file using something like this (adding code to check for overwrite, parameter checking, etc.):

```
 function img2txt, image, filename, status_msg=status_msg

 status = 0
 status_msg = ''

 openw, unit, filename, /get_lun, error=err
 if (err ne 0) then begin
  status = -1
  status_msg = 'Error creating file: ' + filename
  message, status_msg, /continue
 endif else begin
  on_ioerror, IOERROR
  for j = 0, dimension - 1 do begin
   for i = dimension - 1 do begin
    printf, unit, format='(3i12)',  i, j, $
     image[i,j]
   endfor
  endfor
  free_lun, unit
 endelse
 goto, NOERROR

 IOERROR:
    status = -1
    status_msg = 'Error writing to file: ' + filename
    message, status_msg, /continue

 NOERROR:

 return, status
 end
```

You might want to check out the other routines I have available at:

ftp://bial8.ucsd.edu pub/software/idl/share

since many are relevant to medical imaging work.

Dave Foster

```
========== READ_IMG.PRO ==========================================
;  READ_IMG.PRO   3-14-97   DSFoster
;
; Routine to read square image file into array of bytes or integers.
Possible
; sizes for the array are: (4096, 8192, 16384, 32768, 65536, 131072,
262144,
; 524288), depending on dimensions of image: 64x64, 128x128, 256x256 or
; 512x512 (must be square), and upon depth of pixels (8- or 16-bit).
;
; Any existing image header is returned as argument. Note that this
routine
; assumes that an image file's header-length will not be greater than
; the length of the image data.
;
;   Arguments:
;       filename: full pathname of image file
;       image:    NxN matrix holding image of dimension N
;       dim:      dimensions of image, calculated and returned
;       header:   header preceeding image data, if any;
;                   filled and returned
;
; Modifications:
;
; 11-30-94 DSF Initialize Header to null so that if no header is read,
;             returns BYTE('0'). Argument Header is returned as BYTARR.
;  5-16-96 DSF Improve coding.
;  2-04-97 DSF Change error messages.
;  3-14-97 DSF Create generic version for general use.


FUNCTION read_img, filename, image, dim, header

status = 0
header = byte('0')              ; Initialize header

sizes=lonarr(8)                 ; Array of possible image sizes
for i = 0, 7 do $
    sizes(i) = 4096L * (2^i)

openr, unit, filename, /GET_LUN, ERROR=err
```

```
if ( err ne 0 ) then begin
   message, 'File not found or error opening: ' + filename, /continue
   status = -1
   hdr = ''
endif else begin
   on_ioerror, IO_ERROR

   ; Read header if it exists, and then raw image

   fileinfo = fstat( unit )
   ielement = -1

   for i = 0, 7 do begin                        ; Find dimensions
of image
      if ( fileinfo.size ge sizes(i) and fileinfo.size le sizes(7) )
then $
         ielement = i
   endfor

   if ( ielement eq -1 ) then begin
      message, 'Unrecognized image-file format/size: ' + filename, $
 /continue
      print, '       (Size = ', fileinfo.size, ')'
      free_lun, unit
      status = -1
   endif else begin

      result = (ielement + 1) mod 2        ; Is image 1- or 2-byte?
      if ( result eq 0) then begin
         dim = long( sqrt(sizes(ielement)/2) )
         image = intarr(dim, dim)
      endif else begin
         dim = long( sqrt(sizes(ielement)) )
         image = bytarr( dim, dim )
      endelse

      ; Get header size and if nonzero, read from file

      hlength = fileinfo.size mod (dim^2)
      if ( hlength ne 0 ) then begin
         header = bytarr(hlength)
         readu, unit, header
      endif

      ; Read image

      readu, unit, image
      free_lun, unit
```

```
   endelse
endelse

return, status

IO_ERROR:

   message, !ERR_STRING, /CONTINUE          ; Action on I/O errors
   free_lun, unit
   return, -1

END
```

========== READ_IMG.DOC ==========

READ_IMG

The READ_IMG function reads a raw image file and fills a two-dimensional array with the image contents. The dimensions and data-type of the image array are set appropriately (see Restrictions below). The image file may have a header, which is returned as argument. The dimension of the image is also returned.

This routine was written to read medical images such as MRI, PET, and SPECT. See Restrictions below.

Calling Sequence

Result = READ_IMG(filename, image, dimension, header)

Arguments

Filename

The full pathname of the image file.

Image

The 2D image array. This is created and filled by READ_IMG().

Dimension

The dimension of the image, which is assumed to be square. Currently limited to 64, 128, 256 and 512.

Header

The contents of the image header, if any, as BYTARR. If there is no header this will be byte('0').

Outputs

Returns 0 if no error, -1 if either an I/O error occured or an invalid image file was read.

Argument Image is created as 2D array of the appropriate type and dimension, and is filled with the image data.

Argument Dimension is set to be the dimension of the image (assumed square).

Argument Header is created as BYTARR and is filled with header information. If there is no header this will be byte('0').

Restrictions

The image data can only be of type BYTE or INT; the allowable dimensions are 64, 128, 256 or 512, and the image must be square. It is assumed that the data is in raw binary format, as 8- or 16-bit pixels.

If there is an initial header it must contain fewer bytes than the image data.

Example

```
fname = "/usr/image/3184/I.023"
result = READ_IMG(fname, image, dim, header)
```

--

~~~~~~ ~~~~~~
~~~~~~

David S. Foster Univ. of California, San Diego
Programmer/Analyst Brain Image Analysis Laboratory
foster@bial1.ucsd.edu Department of Psychiatry
(619) 622-5892 8950 Via La Jolla Drive, Suite 2240
La Jolla, CA 92037

~~~~~~ ~~~~~~~
~~~~~~