Reslicing ECAT 6.3 files

Riku Klén

28th August 2003

Turku PET Centre Image Processing Report Series

Report TPCIMG 0004

Contents

1 Problem

ECAT 6.3 file is a three dimensional[1] image and that is why it has to be viewed from different angles. The problem is to decide which views are supported and make a program that allows to changes the view.

2 Solution

Three different views are supported: transaxial, sagittal and coronal view. For the used coordinate system x axis is from right to left (radiological convention), y axis is from anterior to posterior and z axis is from head to feet. In the figure below the third axis is away from the viewer, A = anterior, P = posterior, H = head, F = feet, R = right and L = left.

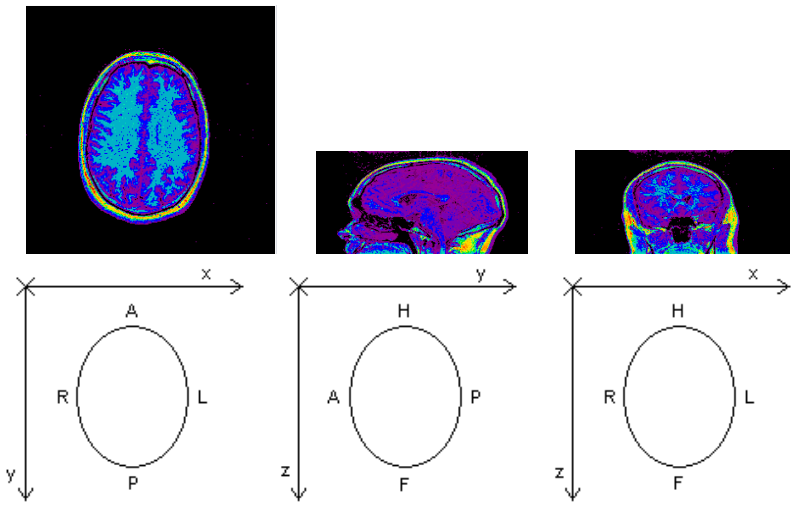

Figure 1: Transaxial view sagittal view coronal view

The program contains four basic transformations which are used to transform a view to another. These are implemented in file `slicer.c`. The actual program the allows to change the view is in file `reslice.c`.

[1] Actually ECAT 6.3 file is a four dimensional images. The fourth dimension is time (frames). If the file contains more than one frame then each frame is considered to be a three dimensional image.

2.1 How to use the program?

The name of the program is *reslice*. It can be used to transform ECAT file of one view to antoher. The program creates a new ECAT file. It takes four arguments:

`cti file`	name of file to be transformed
`new cti file`	name of new transformed file
`from view`	view of cti file (s,t or c)
`to view`	view of new cti file (s,t or c)

For example file `headS.img` is sagittal and it is wanted to transform into a coronal file `headC.img`. This can be done by

`reslice headS.img headC.img s c`.

Below there is a table of basic features.

transformation	`from view`	`to view`
from transaxial to sagittal	`t`	`s`
from transaxial to coronal	`t`	`c`
from sagittal to transaxial	`s`	`t`
from sagittal to coronal	`s`	`c`
from coronal to transaxial	`c`	`t`
from coronal to sagittal	`c`	`s`

2.2 Additional features

Besides changing the view, the program can be used to do any of the four basic transformation and to reduce the image size.

The first basic transformation is image rotation: for every plane the image is rotated 90 degrees. Rotation can be done either clockwise or counterclockwise. To rotate image `from view` should be `ro` and `to view` zero (two) (clockwise) or one (three) (counterclockwise).

The second basic transformation is reslicing: every 3D image for every frame is resliced so that it is seen from different direction. To reslice image `from view` should be `re` and `to view` zero (vertical reslicing) or one (horizontal reslicing).

Because `ecat63` supports only 256 planes the image size has to be checked and reduced before using reslicing method. Reducing image will shrink each plane dimension to half if it is more than 256 pixels. To reduce image `from view` should be `rd`. Argument `to view` can be arbitrary character.

The third basic transformation is inverting image: for each frame the planes are ordered so that the last plane will be the first, the second last will be the

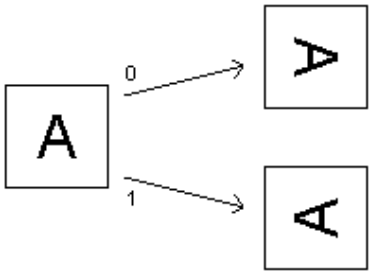

Figure 2: Image rotation.

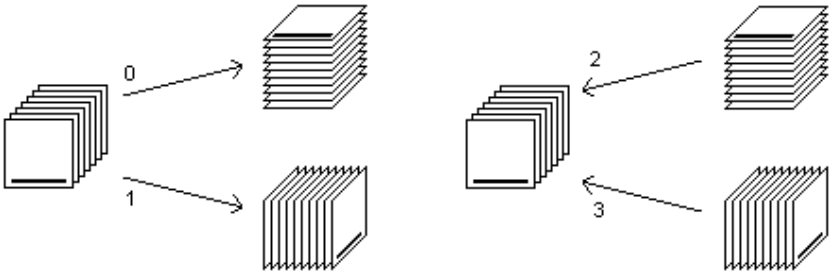

Figure 3: Image reslicing.

second and so on. To invert image `from view` should be `in`. Argument `to view` can be arbitrary.

The last basic transformation is flipping the image: for every plane the image is be flipped horizontally (which means that left (right) side of the image will become right (left) side and top (bottom) of the image will remain top (bottom) of the image). To flip image `from view` should be `fl`. Argument `to view` can be arbitrary character.

Below there is a table of additional features.[2]

[2] * means arbitrary character.

transformation	from view	to view
rotate clockwise	ro	0
rotate counterclockwise	ro	1
reslice vertically	re	0
reslice horizontally	re	1
reslice vertically	re	2
reslice horizontally	re	3
reduce	rd	*
invert	in	*
flip	fl	*

3 Analysis of the program

3.1 slicer.c

File `slicer.c` consists of six methods and one testmethod.

Method `void updateMainheader(ECAT63_mainheader*, ECAT63_mainheader*)` will update the mainheader of output file with the mainheader of input file. The first argument is a pointer to input ECAT main header and the second argument is a pointer to output ECAT main header. This method is used by the other five methods (the sixth method is for testing and it will be introduced later).

Method `void rotate90(IMG*, IMG*, int)` rotates every plane of each frame 90 degrees. The first argument is a pointer to input ECAT main header and the second argument is a pointer to output ECAT main header. The third argument is the direction of the rotation. If it is zero then rotation direction is clockwise. Otherwise rotation direction is counterclockwise. See Figure 2.

Method `void reduce(IMG*, IMG*)` reduces planes of ECAT file. If dimension of a plane is greater than 256 pixels it will be reduced to half (For example 300 will be reduced to 150 and 255 to 127). The first argument is a pointer to input ECAT main header and the second argument is a pointer to output ECAT main header. If dimensions are less or equal to 256 then input file is copied to output file.

Method `void reslice(IMG*, IMG*, int)` reslices ECAT file. Planes of three dimensional image will be shown from a different direction. The first argument is a pointer to input ECAT main header and the second argument is a pointer to output ECAT main header. The third argument is the direction of the reslicing. If it is zero or two then reslicing direction is vertical. Otherwise reslicing direction is horizontal. See Figure 3.

Method `void invert(IMG*, IMG*)` changes the order of planes in every

frame. The first plane will become the last, the second plane will become the second last and so on. The first argument is a pointer to input ECAT main header and the second argument is a pointer to output ECAT main header.

Method `void flip(IMG*, IMG*)` flips each plane horizontally. The left side of the plane will become right and the right side will become the left. The top of the plane will remain the top aswell as bottom will remain bottom. The first argument is a pointer to input ECAT main header and the second argument is a pointer to output ECAT main header.

The last five methods will create a new ECAT file and the input file for them has to exist and to be readable.

3.2 `reslice.c`

File `reslice.c` contains only the `main` method. The program uses methods presented in file `slicer.c`. It will work only if the number of string arguments is four.

First is checked that the third and the fourth arguments are of right lenght. The third argument should consist of one ore two characters and the fourth argument should consist of one character. If they are not of right length one is returned and an error message is printed on the screen.

Next the input and output files are checked. If the input file which is the first argument is not readable, then an error message is printed and two is returned. If the output file which is the argument exists then is asked if it can be overwritten.

Finally the third and the fourth arguments are read and needed procedure performed. If the arguments are incorrect then an error message is printed on the screen.

4 Testing the program

The program has been tested with real PET and MRI data aswell as with a test file named `cubes.img`. This test file can be created by file `cubes.c`. The file `slicer.c` also has a method `void test()` which can be used in testing the basic transformation methods. To use the test method line

```
test();
```

needs to be added in the very beginning of the `main` method in file `reslice.c`.