1 Program 3dvolreg

1.1 Purpose

Registers each 3D sub-brick from the input dataset to the base brick.

1.2 Usage

3dvolreg [options] dataset
'dataset' may contain a sub-brick selector list.

1.3 Options

-verbose Print progress reports. Use twice for LOTS of output.

Interpolation options:
-Fourier Perform the alignments using Fourier interpolation.
-heptic Use heptic polynomial interpolation.
-quintic Use quintic polynomical interpolation.
-cubic Use cubic polynomial interpolation.
Default = Fourier [slowest and most accurate interpolator]

-clipit Clips the values in each output sub-brick to be in the same range as the corresponding input volume. The interpolation schemes can produce values outside the input range, which is sometimes annoying.

-prefix fname Use 'fname' for the output dataset prefix. The program tries not to overwrite an existing dataset.
Default = 'volreg'.

-base n Sets the base brick to be the 'n'th sub-brick from the input dataset (indexing starts at 0).
Default = 0 (first sub-brick).

-base 'bset[n]' Sets the base brick to be the 'n'th sub-brick from the dataset specified by 'bset', as in

-base 'elvis+orig[4]'

The quotes are needed because the '[]' characters are special to the shell.

-dfile dname Save the motion parameters in file 'dname'. The output is in 9 ASCII formatted columns:

n roll pitch yaw dS dL dP rmsold rmsnew

where:

n = sub-brick index

$$\left.\begin{array}{lcl} \text{roll} & = & \text{rotation about the I-S axis} \\ \text{pitch} & = & \text{rotation about the R-L axis} \\ \text{yaw} & = & \text{rotation about the A-P axis} \end{array}\right\}\text{degrees CCW}$$

$$\left.\begin{array}{lcl} \text{dS} & = & \text{displacement in the Superior direction} \\ \text{dL} & = & \text{displacement in the Left direction} \\ \text{dP} & = & \text{displacement in the Posterior direction} \end{array}\right\}\text{mm}$$

rmsold = RMS difference between input brick and base brick

rmsnew = RMS difference between output brick and base brick

N.B.: If the '-dfile' option is not given, the parameters aren't saved.

N.B.: The motion parameters are those needed to bring the sub-brick back into alignment with the base. In `3drotate`, it is as if the following options were applied to each input sub-brick:

-rotate <roll>I <pitch>R <yaw>A -ashift <dS>S <dL>L <dP>P

Algorithm: Iterated linearized weighted least squares to make each sub-brick as like as possible to the base brick. This method is useful for finding SMALL MOTIONS ONLY. See program 3drotate for the volume shift/rotate algorithm. The following options can be used to control the iterations:

-maxite m = allow up to 'm' iterations for convergence [default = 9].

-x_thresh x = iterations converge when maximum movement is less than 'x' voxels [default=0.050000],

-rot_thresh r = and when maximum rotation is less than 'r' degrees [default=0.070000].

-delta d = distance, in voxel size, used to compute image derivatives using finite differences [default=0.700000].

-final mode = do the final interpolation using the method defined by 'mode', which is one of the strings 'cubic', 'quintic', 'heptic', or 'Fourier' [default=mode used to estimate parameters].

-weight 'wset[n]' = set the weighting applyed to each voxel proportional to the brick specified here [default=smoothed base brick].

Warning: This program can consume very large quantities of memory.

Use of '-verbose -verbose' will show the amount of workspace, and the steps used in each iteration.