

TRANSFORM

This command is used for rigid transformation of an image. It can be used, for example, in combination with the ‘register’ command to register two images.

The transformation matrix that defines the transformation is read from a file, hence this command must be preceded by a command/procedure that defines the transformation.

Features

- The following interpolation algorithms are available:
 - Nearest neighbor: the most simple and fastest approach, might give ‘jagged’ edges. Generally not recommended.
 - Linear interpolation: more advanced interpolation, recommended for most situations.
 - Cubic interpolation: most advanced interpolation, but also the slowest and not necessarily better than linear interpolation
- The ‘transform’ command can be applied to images with anisotropic voxel size
- It is possible to define a voxel size for the newly generated transformed image that is different from the voxel size of the original image. Changing the resolution can be useful in case of registering two images with differences in resolution. The output image resolution then can be set to the same value as the resolution of the image to which it should be registered.

Limitations

- Only char-type (1 byte per voxel) and short-type (2 bytes per voxel) aims are supported.
- Only a rigid transformation is performed

Overview input parameters

parameter name	type	default value	relevant range	short description
<i>in</i>	string	in		name of base AIM input object
<i>out</i>	string	out		name of output AIM object
<i>Tmat_file_name</i>	string	out		name of file from which the transformation data is read
<i>img_interpol_option</i>	integer	1	0, 1, 2,	Interpolation option: 0= Nearest neighbor 1= Linear interpolation 2= Cubic interpolation
<i>el_size_mm_out</i>	3×float	-1, -1, -1	>0	Resolution of output [mm]; -1 =same as input

Detailed description of each parameter

<i>Parameter</i>	Description
<i>in</i>	The name of the aim object that represents the image to which the transformation will be applied. The object should be either of char-type (1 byte per voxel) or short-type (2 bytes per voxel). It can be either a gray-level or a segmented image. The use of gray-level images, however, is highly recommended since segmented images will tend to get ‘jagged’ edges after transformation
<i>out</i>	The name of the aim object that holds the transformed image. The type of this image will be the same as that of the input aim object. Its resolution, however, can be different (see parameter <i>el_size_mm_out</i> below)
<i>Tmat_file_name</i>	The name of the file from which the transformation data is read.
<i>img_interpol_option</i>	Interpolation option (see features section above): 0= Nearest neighbor 1= Linear interpolation 2= Cubic interpolation.
<i>el_size_mm_out</i>	Resolution of output [mm]; -1 =same as input.