

REGISTER

This command is used to find the transformation that need to be applied to a second image in order to get the best overlap with a first image. It can be used, for example, to find the transformation that must be applied to a follow-up image to fit it to a baseline image in order to determine locations of bone loss or gain. It can also be used to 'stitch' two partly overlapping datasets.

This command is generally followed by the 'transformation' command to perform the actual transformation. Appendix A lists a script file that can be used to perform a full registration and transformation.

Background

This image registration program tries to find the transformation T that maps image 2 to image 1 (Fig. 1). It does so by searching for the translations and rotations that minimize the difference between image 1 and the transformed image 2. The implemented search procedures try to find this minimum by applying stepwise trial rotations and translations to image 2. Based on the results, new trial rotations and translations are calculated that further reduce the difference. This procedure is repeated until no further improvement is obtained.

A problem with this approach is that there can be many 'local minima' (Fig. 2). Rather than finding the transformation that results in the best fit between the images, the procedure might 'hang' at one of these local minima. To avoid this, the search can be performed first at a much courser resolution. At this courser resolution level, many details, hence, many local minima, will be lost and the global minimum is found. This minimum then can be used as a starting point for the search at a more refined level. An additional advantage of this multi-level approach is that it can save a lot of cpu-time. The search at the levels with a reduced resolution will be much faster than the search at the most refined level. By taking the courser-level solution as the starting point for the next, more refined, level, the search at the more refined level can be restricted such that the final solution will be found faster. A restriction of the 'search-space' (the space in which the rotations and translations will transform the second image) can be obtained by specifying a different typical trial step-size for each of the resolution levels.

Features

With the IPL 'register' command either the full images can be registered or the registration can be limited to a specific part of the images. In the latter case, gobj files are required to identify the regions to be registered. Limiting the registration to specific regions (e.g. bone) can considerably improve the registration of those regions in case the image contains other elements (e.g. soft tissues) that might otherwise affect the registration. The use of gobj files can also considerably reduce the cputime needed to register two images since, when gobj files are specified, the registration search is performed only for the part of the images within the gobj regions.

If no initial guess for the rotations and translations is specified, the program calculates an initial guess for the transformation based on the centers of mass and moments of inertia. If gobj files are specified,

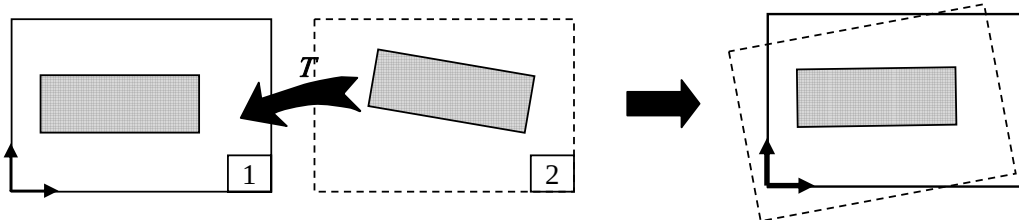


Fig. 1: The registration procedure tries to find the transformation T that results in the best fit of image 1 and the transformed image 2.

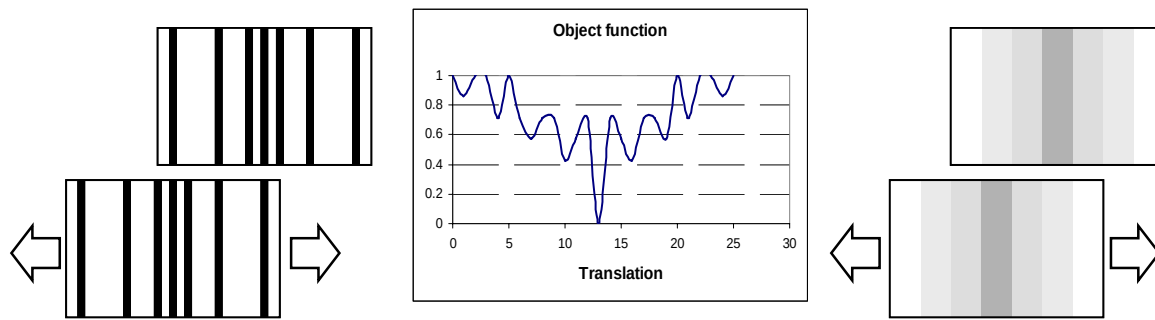


Fig 2 An example of the existence of local minima when registering two images. When trying to register the images drawn at the left, an 'object function' describing the difference between both images as a function of the horizontal translation is found as indicated by the graph in the center. This function shows many local minima at which the registration might 'hang'. If the resolution of the images is much reduced, as conceptually indicated by the images drawn at the right, the local minima disappear and the global minimum is found.

the center of mass and moments of inertia are calculated only for the selected regions.

Starting from this first estimate, a search algorithm is used to find the optimal transformation. Two different search methods are implemented. The first is a simplex method. This is a relatively simple method that usually requires a large number of iteration to find the optimum solution. The other method is a Powell's method. This method requires much less iterations. However, the cpu-time needed per iteration is much increased. As a result of this, the simplex method will generally be 3 to 4 times faster. An advantage of Powell's method, however, is that it is less sensitive for the chosen input parameters (initial guess and trial step size) and thus might find a solution in those cases where the simplex method fails.

The search can be performed at up-to three different resolution levels. In order to reach voxel-accurate registration, the last level should be the full image resolution. For each of the levels, the trial step size can be controlled. In this way it is possible to narrow the search at the more refined levels to reduce the number of iterations and cpu-time.

The search algorithms, in fact, try to find the optimal fit by minimizing an 'object function'. Two different object functions can be specified. The first is a correlation coefficient based function. With this function a correlation coefficient (CC) is calculated based on the grey-levels of both images and the value (1-CC) is minimized. This object function is recommended for most situations. The second object function is based on a mutual information approach. With this approach, both images are posterized to reduce the number of gray levels to 255 and the correlation is based on the posterized images. This object function can be recommended only if the images origin from different modalities (e.g. micro-CT versus MR).

The regis command can be applied to images with anisotropic voxel sizes and can register images which differ in resolution. It is recommended, however, to use images with near isotropic voxels of about the same size.

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>in1</i>	string	in1		name of base AIM input object
<i>gobj_file_name_in1</i>	string	none		GOBJ file for base AIM input object
<i>in2</i>	string	in2		name of AIM object to be registered to in1
<i>gobj_file_name_in2</i>	string	none		GOBJ file for AIM object to be registered
<i>Tmat_file_name</i>	string	out		name of file to which the transformation data is written
<i>initial_rotation</i>	3×float	0.0 0.0 0.0	$-\pi - \pi$	initial guess for rotations around the x- y- and z-axes [radians]
<i>initial_translation</i>	3×float	0.0 0.0 0.0		initial guess for translations in x- y- and z-direction [mm]
<i>delta_rotation</i>	3×float	0.1 0.1 0.5	$-\pi - \pi$	rotational steps size [radians]
<i>delta_translation</i>	3×float	0.1 0.1 0.5		translational steps size [mm]
<i>resolution_scaling</i>	3×integer	16 4 1	≥ 0	resolution reduction for the different search levels
<i>delta_scaling</i>	3×float	1.0 0.1 0.1	> 0	level scaling for the delta values
<i>tolerance</i>	float	1.0E-5	1.0E-6 – 1.0E-2	Stop criterion for the minimization procedure
<i>min_corr_coeff</i>	float	0.50	0.0 – 0.999	Min. required value for the correlation coefficient
<i>min_method</i>	integer	1	1, 2, 3	Minimalization method: 1= Simplex 2= Powell 3= Combined
<i>object_func</i>	integer	1	1, 2	Object function: 1=Correlation 2=Mutual Information
<i>max_nr_iter</i>	integer	1000	> 0	Max number of iterations

Detailed description of each parameter

<i>Parameter</i>	Description
<i>in1</i>	The name of the aim object that represents the 'baseline' image. 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. For the best possible registration, however, the use of gray-level images is recommended.
<i>gobj_filename_in1</i>	It is possible to perform the registration for only a part of the image. This part can be identified by a gobj file. If a gobj file is specified, the part of the image outside the selected region is not taken into account when calculating the best transformation of <i>in2</i> to <i>in1</i> .
<i>in2</i>	The name of the aim object that represents the 'follow-up' image. This is the image that will be transformed to <i>in1</i> . 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. For the best possible registration, however, the use of gray-level images is recommended.
<i>gobj_filename_in2</i>	It is possible to perform the registration for only a part of the image. This part can be identified by a gobj file. If a gobj file is specified, the part of the image outside the selected region is not taken into account when calculating the best transformation of <i>in2</i> to <i>in1</i> .
<i>Tmat_file_name</i>	The name of the file to which the transformation data is written. This file can be read later by the IPL 'transform' command to perform the actual transformation of image 2
<i>initial_rotation</i>	It is possible to specify an initial guess for the rotations around the x- y- and z-axes [radians]. If all initial rotations and translations are left at zero, the program will calculate an initial guess based on the center of mass and the moments of inertia of the image or of the gobj-defined selected regions therein.
<i>initial_translation</i>	It is possible to specify an initial guess for the translations in the x- y- and z-direction [mm]. If all initial rotations and translations are left at zero, the program will calculate an initial guess based on the center of mass and the moments of inertia of the image or of the gobj-defined selected regions therein.

<i>Parameter</i>	<i>Description</i>
<i>delta_rotation</i>	These values determine the typical trial step-size that is taken for the rotations during the minimization search around the x-, the y- and z-axis respectively [radians]. Selecting larger values will enable the program to search in a wider range while selecting small values will narrow the search. It is possible to specify different values for the rotation around the different axes to account for the fact that larger deviations are expected around one or more of the axis. For example, with the default setting a 5 times larger value is specified for the rotation around the z-axis. This is because, in many situations, the rotation around the longitudinal z-axis will be larger than those around the transversal x- and y-axes. Note: for small values the rotation corresponds to a 1 voxel shift over a length of $1/\textit{delta_rotation}$ voxels. As an example: a value of 0.01 corresponds to a 1 voxel shift over a length of 100 voxels. Note: setting a value of <i>delta_rotation</i> to a very small value effectively suppresses rotations around the corresponding axis.
<i>delta_translation</i>	These values determine the typical trial step-size that is taken for the translations during the minimization search in the x-, y- and z-direction [mm]. Selecting larger values will enable the program to search in a wider range while selecting small values will narrow the search. It is possible to specify different values for the translation in the different directions to account for the fact that larger deviations are expected in one or more directions. For example, with the default setting a 5 times larger value is specified for the translation in the z-direction. This is because, in many situations, the translation in the longitudinal z-direction will be larger than that in the transversal x- and y-direction. Note: setting a value of <i>delta_translation</i> to a very small value effectively suppresses translations in the corresponding direction.
<i>resolution_scaling</i>	Defines the resolution reduction factors. Up-to three levels at which the transformation search will take place can be defined. The first level should have the highest reduction factor (i.e. lowest resolution), the last one the lowest value. The value to choose for these factors can depend on the image size and resolution. If less than 3 levels are required, a zero-value can be specified for the last level(s). To get voxel-accurate results, the last level should have the <i>resolution_scaling</i> set to 1. If all scaling factors are set to zero, no search is performed and the final transformation reflects the values specified by the <i>initial_rotation</i> and the <i>initial_translation</i>.

<i>Parameter</i>	<i>Description</i>
<i>delta_scaling</i>	It is possible to scale the delta values specified by <i>delta_rotation</i> and <i>delta_translation</i> to a different value for each search level. The actual <i>delta_rotation</i> and <i>delta_translation</i> used for each level are set to: $\text{delta_scaling} \times \text{delta_rotation}$ and $\text{delta_scaling} \times \text{delta_translation}$. In this manner it is possible to allow a wide range of rotation/translation at the coarsest level but to narrow the search-space down to smaller values at the finest level. Reducing the search-space can considerably reduce the number of iteration required to find the final solution. However, when the search-space is too narrow, the quality of the fit will be affected.
<i>tolerance</i>	Tolerance level for the search procedure. The search for the best fit will be continued until improvements in the result are smaller than the specified value. Note that specifying a low value does not guarantee a good fit; it indicates that the fit can not be further improved. The actual quality of the fit can be judged from the correlation coefficient (CC) that is printed to the screen during the registration search.
<i>min_corr_coef</i>	Specifies the minimum requested value for the correlation coefficient. If the search at any of the specified level results in a correlation coefficient less than this specified value, the program will abort with an error message. When fitting two very similar images one of which is rotated/translated, the <i>min_corr_coef</i> should be in the range of 0.95 - 0.99. When fitting 'follow-up' images to 'baseline' images the best correlation that can be reached will obviously depend on the changes between both images and a lower value might be required. The value can also depend on the chosen object function.
<i>min_method</i>	Specifies the minimization method to be used for the search procedure. If set to 1 the simplex method is used, if set to 2 Powell's method is used. If set to 3 Powell's method is used for all levels, except for the finest level, for which the simplex method is used. The rationale behind this choice is that the more reliable Powell's method might be the method of choice for the coarser levels where cpu-time plays only a minor role.
<i>object_func</i>	Defines the object function to be used by the minimization method. If set to 1 a correlation coefficient (CC) based function is used, if set to 2 a mutual information (MI) based function is used. For images from the same source (i.e. micro-CT), the CC function will generally be the best choice. The use of the MI function can be recommended only in case images of different modalities are used (e.g. micro-CT versus MR).

Parameter	Description
<i>max_nr_iteration</i>	Defines the maximum number of iterations for the minimization method. This number will depend on the chosen method and on the specified <i>tolerance</i> . When the simplex method is selected (<i>min_method</i> =1) a value of 1000 is recommended. When Powell's method is selected (<i>min_method</i> =2) a value of 20 is recommended. The search will abort with an error message if the number of iterations exceeds the specified maximum.

Important notes:

- 1) The final transformation will account for the position of image 1 and image 2 in their original CT-scan. This implies that the transformation is valid as well for other sub-images from the same CT-scan of which image 2 is obtained (e.g. a mask file). It will not be valid, however, for images from other CT-scans that may have different positions.
- 2) Care should be taken that the values for parameter *resolution_scaling* are chosen such that the reduced images are at least 10 voxels in size. For example, if the images are 256×256×256 in size, the highest allowable value for this parameter would be 26.

FAQ

1) What to do if the program has converged but the results is not satisfactory?

You probably arrived at a local minimum. Change the input parameters and try again. In order to speed-up this process, it is recommended to try the registration first for the courser levels. For example, set the *reduction_factor* parameter to 16 4 0. It is recommended as well to set the *interpol_option* to 1 during the test-phase, to speed-up the final transformation.

Then: check the correlation coefficients obtained at these courser levels and the resulting transformed image. If using the simplex method, switch to Powell's method first to investigate if this significantly improves the correlation.

If correlations are still too low, change the *delta_rotation* and *delta_translation* parameters as well as the *delta_scaling* parameter until satisfying correlations are found.

In some cases, it can be necessary as well to change the *resolution_scaling* settings.

To speed-up the transformation,

2) How to avoid that the first guess is based on the fit of the mass-center and the moments of inertia?

In some cases a first guess based on the fit of the mass-center and the moments of inertia might not be effective. This first guess can be omitted by specifying a small non-zero value (i.e. 1.0E-6) for one of the initial rotations or initial translations. If any of these values is non-zero, the initial guess will be skipped.

3) How to stitch two partly overlapping images?

Stitching overlapping images can be useful in case the standard stacking of images provide unsatisfying results. This can be the case, for example, if (small) movement artifacts are present during in-vivo scanning. In such situations, the generated image stacks might no longer precisely fit together. This problem can be reduced by creating two or more stacks that partly overlap and use image registration to find the best fit between these stacks.

In this situation it is essential to provide a good estimate for the initial translation that needs to be applied to image 2. The image registration then can be performed in the standard manner.

Appendix A: Sample script to register two images

The script below will register two images, perform the transformation of the second image, and finally create a segmented output image that holds a value '40' for voxels only appearing in the first image, a value of '80' for voxels appearing in the second image, and a value of '120' for voxels that appear in both images. In the 3D viewer different colors can be assigned to these values to visualize changes between the images.

A typical example of how to use this script is the following:

- The first image is the baseline image of a bone
- The second image represents a follow-up image
- The generated image then is colored in the 3D viewer such that a value of '40' is represented by 'red', a value of '80' by 'green' and a value of '120' by 'yellow'

In this situation, the final image shows locations where bone is lost in red, location where new bone is formed in green, and locations where bone remains in yellow.

The script is called by typing:

```
$! @register c00000001.aim c00000002.aim comb1-2.aim c00000001.gobj c00000002.gobj
```

where all names must be replaced by the correct file names and specification of the gobj files is optional

```
$!
$!      _/_/_/_   _/_/_/_   _/_
$!      _/_       _/_       _/_
$!      _/_       _/_       _/_
$!      _/_       _/_       _/_
$!      _/_       _/_       _/_
$!      _/_       _/_       _/_
$!
$!
$!
$!      IPL Batch Scanco
$!
$!
$!=====
$! call as:
$! @register c0000001.aim c0000002.aim comb1-2.aim [c0000001.gobj c0000002.gobj]
$!
$! Note: specification of the gobj files can be omitted
$! Also specify the following:
$!
$ sigma_gauss      := 0.7
$ support_gauss    := 1
$ threshold_seg     := 220
$ peel_iter_gobj   := 0
$ rotation_z       := 0
$
$=====
$ define in1_file   'p1'
$ define in2_file   'p2'
$ define test_file   'p3'
$
$ if p4 .eqs. ""
$ then
$   define gobj1_file "none"
$ else
$   define gobj1_file 'p4'
$ endif
$ if p5 .eqs. ""
$ then
$   define gobj2 file "none"
```

```

$ else
$     define    gobj2_file 'p5'
$ endif
$
$ show log in1_file
$ show log gobj1_file
$ show log in2_file
$ show log gobj2_file
$!
$!
$ ipl_sanco_prog := $um:ipl_sanco_m.exe
$!
$ ON ERROR THEN EXIT
$!
$ ipl_sanco_prog

/read
-name                in1
-filename            in1_file

/read
-name                in2
-filename            in2_file

/register
-in1                 in1
-gobj_filename_in1   gobj1_file
-in2                 in2
-gobj_filename_in2   gobj2_file
-Tmat_file_name      Tmat.dat
-initial_rotation     0.0 0.0 0.0
-initial_translation  0.0 0.0 0.0
-delta_rotation       0.10 0.10 0.50
-delta_translation    0.10 0.10 0.50
-resolution_scaling    16 4 1
-delta_scaling         1.00 0.10 0.10
-tolerance            0.00001
-min_corr_coef        0.5
-min_method           1
-object_func           1
-max_nr_iter          1000

/transform
-in                  in2
-out                 out
-Tmat_file_name      Tmat.dat
-img_interpol_option 1
-el_size_mm_out      -1 -1 -1

/del in2

! segment the baseline image; set value to 40

/sup
-input               in1
-supdim_numbers       4 4 1
-testoff_pixels       "support_gauss"
-suppos_pixels_local  -1 -1 -1
-subdim_pixels        -1 -1 -1

/seg_gauss
-input               in1
-output              in1_seg
-sigma               "sigma_gauss"
-support              "support_gauss"
-lower_in_perm        "threshold_seg"
-upper_in_perm        1000
-value_in_range       40

/del in1

! segment the transformed follow-up image; set value to 80

/sup

```

```

-input                out
-supdim_numbers      4 4 1
-testoff_pixels      "support_gauss
-suppos_pixels_local -1 -1 -1
-subdim_pixels       -1 -1 -1

/seg_gauss
-input                out
-output              out_seg
-sigma               "sigma_gauss
-support             "support_gauss
-lower_in_perm       "threshold_seg
-upper_in_perm       1000
-value_in_range      80

/del out

! add the segmented images; common regions will get a value of 120

/concat
-input1              in1_seg
-input2              out_seg
-output              out
-common_region_only  false
-add_not_overlay     true
-make_edge           false
-shift_ofin2         0 0 0
-turnangle           0.0
-turnpoint_global    -1 -1

/write out test_file

..
$ exit

```