Parameters

graphNameStr can be "" to refer to the top graph window.

When identifying a subwindow with *graphNameStr*, see **Subwindow Syntax** on page III-92 for details on forming the window hierarchy.

separatorStr should contain a single ASCII character such as "," or ";" to separate the names.

An image name is defined as the name of the 2D wave that defines the image with an optional #ddd suffix that distinguishes between two or more images that have the same wave name. Since the image name has to be parsed, it is quoted if necessary.

Examples

The following command lines create a very unlikely image display. If you did this, you would want to put each image on different axes, and arrange the axes such that they don't overlap. That would greatly complicate the example.

```
Make/O/N=(20,20) jack,'jack # 2';
Display;AppendImage jack
AppendImage/T/R jack
AppendImage 'jack # 2'
AppendImage/T/R 'jack # 2'
Print ImageNameList("",";")
```

prints jack;jack#1;'jack # 2';'jack # 2'#1;

See Also

Another command related to images and waves: **ImageNameToWaveRef**.

For commands referencing other waves in a graph: **TraceNameList**, **WaveRefIndexed**, **XWaveRefFromTrace**, **TraceNameToWaveRef**, **CsrWaveRef**, **CsrXWaveRef**, **ContourNameList**, and **ContourNameToWaveRef**.

ImageNameToWaveRef

ImageNameToWaveRef(*graphNameStr*, *imageNameStr*)

The ImageNameToWaveRef function returns a wave reference to the 2D wave corresponding to the given image name in the graph window or subwindow named by *graphNameStr*.

Parameters

graphNameStr can be "" to refer to the top graph window.

When identifying a subwindow with *graphNameStr*, see **Subwindow Syntax** on page III-92 for details on forming the window hierarchy.

The image is identified by the string in *imageNameStr*, which could be a string determined using **ImageNameList**. Note that the same image name can refer to different waves in different graphs.

See Also

The **ImageNameList** function.

For a discussion of wave references, see **Wave Reference Functions** on page IV-197.

ImageRegistration

ImageRegistration [*flags*] **[testMask=*testMaskWave*] [refMask=*refMaskWave*] testWave=*imageWave1*, refWave=*imageWave2***

The ImageRegistration operation adjusts the test image wave, testWave, to match the reference image wave, refWave, possibly subject to auxiliary mask waves. The registration may involve offset, rotation, scaling or skewing.

Image data may be in two or three dimensions.

ImageRegistration is designed to find accurate registration for relatively small variation on the order of a few degrees of rotation and a few pixels offset between the reference and test images.

All the input waves are expected to be single precision float (SP) so you may have to redimension your images before using ImageRegistration.

ImageRegistration does not tolerate NaNs or INFs; use the masks if you need to exclude pixels from the registration process.

Parameters

refMask=*refMaskWave* — Specifies an optional ROI wave used to mask refWave. Omit refMask to use all of the refWave.

The wave must have the same dimensions as refWave. refMask is a single precision floating point wave with nonzero entries marking the "ON" state. Note that the operation modifies this wave and that you should not use the same wave for both the reference and the test masks.

refWave=*imageWave2* — Specifies the name of the reference image wave used to adjust testWave.

testMask=*testMaskWave* — Specifies an optional ROI wave used to mask testWave. Omit testMask to use all of the testWave.

The wave must have the same dimensions as refWave. testMask is a single precision floating point wave with nonzero entries marking the "ON" state. Note that the operation modifies this wave and that you should not use the same wave for both the reference and the test masks.

testWave=*imageWave1* — Specifies the name of the image wave that will be adjusted to match refWave.

testMaskWave and *refMaskWave* are optional ROI waves. The waves must have the same dimensions as testWave and refWave respectively. They must be single precision floating point waves with nonzero entries marking the "ON" state. If you need to include the whole region described by testWave or the whole region described by refWave you can omit the respective mask wave

Flags

/ASTP=*val* — Sets the adaptation step for the Levenberg-Marquardt algorithm. Default value is 4.

/BVAL=*val* — Enables clipping and sets the background values to which masked out voxels of the test data will be set.

/CONV=*val* — Sets the convergence method.

- *val*=0: Gravity, use if the difference between the images is only in translation. This option is frequently useful as a first step when the test and reference data are too far apart for accurate registration. The result of this registration is then passed to a subsequent ImageRegistration with /CONV=1.
- *val*=1: Marquardt.

/CSNR=*val* — Determines if the operation calculates the signal to noise ratio (SNR)

- *val*=0: The SNR is not calculated.
- *val*=1: The SNR is calculated (default).

Skipping the SNR calculation saves time and may be particularly useful when performing the registration on a stack of images.

/FLVL=*val* — Specifies the finest level on which the optimization is to be performed.

If this is the same as /PRDL, then only the coarsest registration calculation is done.

If /FLVL=1 (default), then the full multiresolution pyramid is processed. You can use this flag to terminate the computation at a specified coarseness level greater than 1.

/GRYM — Optimizes the gray level scaling factor.

It is sometimes dangerous to let the program adjust for gray levels because in some situations it might result in a null image.

/GRYR — Renders output using the gray scaling parameter. This is more meaningful if the operation computes the optimal gray scaling (see /GRYM).

/GWDT={*sx*,*sy*,*sz*} Sets the three fields to the half-width of a Gaussian window that is used to smooth the data when computing the default masks. Defaults are {1,1,1}. See /REFM and /TSTM for more details.

/INTR=*val* Sets the interpolation method.

- *val*=0: Nearest neighbor. Used when registering the center of gravity of the test and reference images.
- *val*=1: Trilinear.
- *val*=2: Tricubic (default).

/ISOS Optimizes the isometric scaling. This option is inappropriate if voxels are not cubic.

/ISR Computes the multiresolution pyramid with isotropic size reduction.

If the flag is not specified, the size reduction is in the XY plane only.

/MING=*val* Sets the minimum gain at which the computations will stop. Default value is zero, but you can use a slightly larger value to stop the iterations earlier.

/MSKC=*val* Sets mask combination value. During computation the masks for the test data and the mask for the reference data are also transformed. This flag determines how the two masks are to be combined. The registration criteria are computed for the combination of the two masks.

- *val*=0: or.
- *val*=1: nor.
- *val*=2: and (default).
- *val*=3: nand.
- *val*=4: xor.
- *val*=5: nxor.

/PRDL=*depth* Specifies the depth of the multiresolution pyramid. The finest level is *depth*=1. Each level of the pyramid decreases the resolution by a factor of 2. By default, the pyramid *depth*=4, which corresponds to a resolution reduction by a factor of $2^{(depth-1)}$=8.

The algorithm starts by computing the first registration on large scale features in the image (deepest level of the pyramid). It then makes small corrections to the registration at each consecutive pyramid level.

For best results, the coarsest representation the data should be between 30 and 60 pixels on a side. For example, for an image that is *H* by *V* pixels, you should choose the depth such that $H/2^{(depth-1)} \approx 30$.

/PSTK When performing registration of a stack of images, use this flag to apply the registration parameters of the previous layer as the initial guess for the registration of each layer after the first in the 3D stack.

/Q Quiet mode; no messages printed in the history area.

/REFM=*val* Sets the reference mask.

- *val*=0: To leave blank and then every pixel is taken into account.
- *val*=1: Value will be set if a valid reference mask is provided.
- *val*=2: The test mask is computed (default).

When computing the reference mask it is assumed that brighter features are more important. This is done by using a low pass filter on the data (using the parameters in /GWDT) which is then converted into a binary mask. Note that you do not need to specify /REFM=1 if you are providing a reference mask wave. See also /TSTM.

/ROT={*rotX*,*rotY*,*rotZ*}

Determines if optimization will take into account rotation about the X, Y, or Z axes. A value of one allows optimization and zero prevents optimization from affecting the corresponding rotation parameter. Defaults are {0,0,1}, which are the appropriate values for rotating images.

/SKEW={*skewX*,*skewY*,*skewZ*}

Determines if optimization will take into account skewness about the X, Y, or Z axes. A value of one allows optimization and zero prevents optimization from affecting the corresponding skewness parameter. Defaults are {0,0,0}. Note that skewing and rotation or isometric scaling are mutually exclusive operations.

/STCK Use /STCK to perform the registration between a 2D reference image and each of the layers in a 3D image. The number of rows and columns of the refWave must match exactly the number of rows and columns in testWave. The transformation parameters are saved in the wave M_RegParams where each column contains the parameters for the corresponding layer in testWave.

/STRT=*val* Sets the first value of the adaptation parameter in the Levenberg-Marquardt algorithm.

The default value of this parameter is 1.

/TRNS={*transX*,*transY*,*transZ*}

Determines if optimization will take into account translation about the X, Y, or Z axes. A value of one allows optimization and zero prevents optimization from affecting the corresponding translation parameter. Defaults are {1,1,0}, which are appropriate for finding the X and Y translations of an image.

/TSTM=*val* Sets the test mask.

val=0: To leave blank and then every pixel is taken into account.

val=1: This value will be set if a valid reference mask is provided.

val=2: The reference mask is computed. This is the default value.

The test image mask is computed in the same way as the reference image mask (see /REFM) using the same set of smoothing parameters. Note that you do not need to specify /TSTM=1 if you are providing a test mask wave.

/USER=*pWave* Provides a user transformation that will be applied to the input testWave in order to create the trasnsformed image. *pWave* must be a double precision wave which contains the same number of rows as W_RegParams.

Note: If you use a previously created W_RegParams make sure to change its name as it is overwritten by the operation.

If *pWave* has only one column and testWave contains multiple layers, then the same transformation applies to all layers. If *pWave* contains more than one column, then each layer of testWave is processed with corresponding column. If there are more layers than columns the first column is used in place of the missing columns.

/ZMAN Modifies the test and reference data by subtracting their mean values prior to optimization.

Details

ImageRegistration will register images that have sufficiently similar features. It will not work if key features are too different. For example, ImageRegistration can handle two images that are rotated relative to each other by a few degrees, but cannot register images if the relative rotation is as large as 45 degrees. The algorithm is capable of subpixel resolution but it does not handle large variations between the test image and the reference image. If the centers of the two images are too far from each other, you should first try ImageRegistration using /CON=0 to remove the translation offset before proceeding with a finer registration of details.

The algorithm is based on an iterative processing that proceeds from coarse to fine detail. Optimization uses a modified Levenberg-Marquardt algorithm and produces an affine transformation for the relative rotation

and translation as well as for isometric scaling and contrast adjustment. The algorithm is most effective with square images where the center of rotation is not far from the center of the image.

When using gravity for convergence, skew parameters can't be evaluated (only translation is supported). Skew and isoscaling are mutually exclusive options. Mask waves are defined to have zero entries for pixels outside the region of interest and nonzero entries otherwise. If a mask is not provided, every pixel is used.

ImageRegistration creates the waves M_RegOut and M_RegMaskOut, which are both single precision waves. In addition, the operation creates the wave W_RegParams which stores 20 double precision registration parameters. M_RegOut contains the transformed (registered) test image and M_RegMaskOut contains the transformed mask (which is not affected by mask combination). ImageRegistration ignores wave scaling; images are compared and registered based on pixel values only.

The results printed in the history include:

dx, dy, dz	translation offsets measured in pixels.
a*ij*	Elements in the skewing transformation matrix.
phi	Rotation angle in degrees about the X-axis. Zero for 2D waves.
tht	Rotation angle in degrees about the Y-axis. Zero for 2D waves.
psi	Rotation angle in degrees about the Z-axis.
det	Absolute value of determinant of the skewing matrix (a*ij*).
err	Mean square error defined as $\frac{1}{N}\sum\left(x_i - y_i\right)^2$, where x_i is the original pixel value, y_i the computed value, and N is the number of pixels.
snr	Signal to noise ratio in dB. It is given by: $10\log\left(\frac{\sum x_i^2}{\sum\left(x_i - y_i\right)^2}\right)$.

These parameters are stored in the wave W_RegParams (or M_RegParams in the case of registering a stack). Angles are in radians. Dimension labels are used to describe the contents of each row of the output wave. Each column of the wave consists of the following rows (also indicated by dimension labels):

Point	Contents	Point	Contents	Point	Contents	Point	Content
0	dx	6	a21	12	gamma	17	origin_x
1	dy	7	a22	13	phi	18	origin_y
2	dz	8	a23	14	theta	19	origin_z
3	a11	9	a31	15	psi	21	MSE
4	a12	10	a32	16	lambda	21	SNR
5	a13	11	a33				

You can view the output waves with dimension labels by executing:

```
Edit W_RegParams.ld
```

See Also

The ImageInterpolate **Warp** operation.