ImageSeedFill returns a "bad seed pixel specification" if the seed pixel location derived from the various keywords above satisfies one or more of the following conditions:

- The computed integer pixel/voxel is outside the image.
- The value stored in the computed integer pixel/voxel location does not satisfy the min/max or fuzzy conditions. This is the most common condition when srcWave has wave scaling. To avoid this difficulty you should use the keywords seedP, seedQ, and seedR.

Examples

Using Cursor A position and value to supply parameter inputs for a 2D seedFill (**Warning**: command wrapped over two lines):

```
ImageSeedFill
   seedP=pcsr(a),seedQ=qcsr(a),min=zcsr(a),max=zcsr(a),target=0,srcwave=image0
```

Using the fuzzy algorithm for a 3D wave (**Warning**: command wrapped over two lines):

```
ImageSeedFill seedX=232,seedY=175,seedZ=42,target=1,fillNumber=10,fuzzyCenter=.25,
   fuzzyWidth=1,fuzzyScale=1,fuzzyProb=0.4,srcWave=ddd
```

See Also

For an additional example see **Seed Fill** on page III-377. To display the result of the operation for 3D waves it is useful to convert the 3D wave M_SeedFill into an array of quads. See ImageTransform **vol2surf**.

ImageSnake

ImageSnake [***flags***] ***srcWave***

The ImageSnake operation creates or modifies an active contour/snake in the grayscale image srcWave. The operation iterates to find the "lowest total energy" snake. The energy is defined by a range of optional flags, each corresponding to an individual term in the total energy expression. Iterations terminate by reaching the maximum number of iterations, when the snake does not move between iterations or when the user aborts the operation.

Flags

/ALPH=*alpha* — Sets the coefficient of the energy term arising from the "tightness" of the snake.

/BETA=*beta* — Sets the coefficient of the energy term corresponding to curvature of the snake. A high value for beta makes the snake more rounded.

/DELT=*delta* — Sets the coefficient of the repulsion energy. A high value of *delta* keeps nonconsecutive snake points far from each other.

/EPS=*num* — Sets the maximum number of vertices which are allowed to move in one iteration. If the number of vertices which move during an iteration is smaller than *num* then iterations terminate.

/EXEF=*eta* — Sets the coefficient of the optional external energy component. By default this value is set to zero and there is no external energy contribution to the snakes energy. Note, this component is referred to as "external" because it is completely up to the user to specify both its coefficient and the value associated with each pixel. It should not be confused with what is called external snake energy in the literature, which usually applies to energy proportional to the gradient image (see /GAMM and /GRDI).

/EXEN=*wave* — Specify a wave that contains energy values that will be added to the snakes energy calculation. The wave must have the same dimensions as *srcWave* and must be single precision float. Each pixel value corresponds to user defined energy which will be multiplied by the /EXEF coefficient and added to the sum which the snake minimizes. Note that when /EXEF is set to zero this component is ignored. An external energy wave may be useful, for example, if you want to attract the snake to the picture boundaries. In that case you can set:

```
Duplicate/O srcWave,extWave
Redimension/S extWave
Variable rows=DimSize(srcWave,0)-1
Variable cols=DimSize(srcWave,1)-1
extWave=(p==0 || q==0 || p==rows || q==cols) ? 0:1
```

/GAMM=*gamma* Sets the coefficient of the energy term corresponding to the gradient. A high value of gamma makes the snake follow lines of high image gradient.

/GRDI=*gWave* Specify the gradient image. This wave must have the same dimensions as *srcWave* and it must be single precision float. The wave corresponds to the quantity `abs(grad(gauss**srcWave))`, where grad is the gradient operator and ** denotes convolution of the source wave with a Gaussian kernel. It is best to run the operation the first time without specifying this wave. When the operation executes, it creates the wave M_GradImage which can then be used in subsequent executions of this operation. If you want to modify the wave to express some other form of energy that you want the operation to minimize, you should use the /EXEN and /EXEF flags.

/ITER=*iterations* Sets the maximum number of iterations. Convergence can be achieved if the value specified by /EPS is met. You can also terminate the process earlier by pressing the **User Abort Key Combinations**.

/N=*snakePts* Specify the number of vertices in the snake or the number of snake points. Note that if you are providing snake waves in /SX and /SY, you do not need to specify this flag. If you do not specify this flag the default value is 40.

/SIG=*sigma* Sets the size of the Gaussian kernel that is used to convolve the input image when creating the gradient image. Note that you do not need to use this flag if you provide a gradient image. *sigma* is 3 by default. You can use larger odd integers for larger Gaussian kernels which would correspond to a stronger blur.

/STRT={*centerX*, *centerY*, *radius*}

Sets the starting snake to be a circle with the given center and radius. If you use this flag you should also provide the number of snake points using the /N flag.

/STEP=*pixels* Sets the maximum radius of search. By default the radius of the search is 6 pixels and the search follows a clockwise pattern from radius of 1 pixel to maximum radius specified by this flag. Note: the search radius should be smaller than the dimension of a typical feature in the image. If the radius is larger the snake may encompass more than one object. Larger radius is also less efficient because many of the pixels in that range would result in a snake that crosses itself and hence get rejected in the process.

/SX=*xSnake* Provide an X-wave for the starting snake. You must also provide an appropriate Y-wave using /SY.

/SY=*ySnake* Provide a Y-Wave for the starting snake. Must work in combination with /SX.

/UPDM=*mode* Sets the update mode using any combination of the following:

Value	Update
0	Once when the operation terminates.
1	Once at the end of every iteration.
2	Once after every snake vertex moves.
4	Once for every search position.

/Q Quiet mode; don't print information in the history.

/Z Don't report any errors.

Details

A snake is a two-dimensional, usually closed, path drawn over an image. The snake is described by a pair of XY waves consisting of N vertices (sometimes called "snake elements" or "snaksels"). In this implementation it is assumed that the snake is closed so that the last point in the snake is connected to the first. Snakes are used in image segmentation, when you want to automatically select a contiguous portion of the plane based on some criteria. Unlike the classic contours, snakes do not have to follow a constant level. Their structure (or path) is found by associating the concept of energy with every snake configuration and attempting to find the configuration for which the associated energy is a minimum. The search for a