fspecial

Create predefined 2-D filter

Syntax

```
h = fspecial(type)
h = fspecial(type,parameters)
```

Description

`h = fspecial(`*`type`*`)` creates a two-dimensional filter h of the specified `type`. `fspecial` returns h as a correlation kernel, which is the appropriate form to use with `imfilter`. *`type`* is a string having one of these values.

Value	Description
`'average'`	Averaging filter
`'disk'`	Circular averaging filter (pillbox)
`'gaussian'`	Gaussian lowpass filter
`'laplacian'`	Approximates the two-dimensional Laplacian operator
`'log'`	Laplacian of Gaussian filter
`'motion'`	Approximates the linear motion of a camera
`'prewitt'`	Prewitt horizontal edge-emphasizing filter
`'sobel'`	Sobel horizontal edge-emphasizing filter
`'unsharp'`	Unsharp contrast enhancement filter

h = fspecial(*type*,parameters) accepts a filter `type` plus additional modifying `parameters` particular to the type of filter chosen. If you omit these arguments, `fspecial` uses default values for the`parameters`.

The following list shows the syntax for each filter type. Where applicable, additional parameters are also shown.

- `h = fspecial('average',hsize)` returns an averaging filterh of size `hsize`. The argument`hsize` can be a vector specifying the number of rows and columns in h, or it can be a scalar, in which case h is a square matrix. The default value for `hsize` is `[3 3]`.

- `h = fspecial('disk',radius)` returns a circular averaging filter (pillbox) within the square matrix of side `2*radius+1`. The default `radius` is 5.
- `h = fspecial('gaussian',hsize,sigma)` returns a rotationally symmetric Gaussian lowpass filter of size `hsize` with standard deviation `sigma` (positive). `hsize` can be a vector specifying the number of rows and columns in `h`, or it can be a scalar, in which case `h` is a square matrix. The default value for`hsize` is `[3 3]`; the default value for`sigma` is 0.5.
- `h = fspecial('laplacian',alpha)` returns a 3-by-3 filter approximating the shape of the two-dimensional Laplacian operator. The parameter`alpha` controls the shape of the Laplacian and must be in the range 0.0 to 1.0. The default value for `alpha` is 0.2.
- `h = fspecial('log',hsize,sigma)` returns a rotationally symmetric Laplacian of Gaussian filter of size `hsize` with standard deviation `sigma` (positive). `hsize` can be a vector specifying the number of rows and columns in `h`, or it can be a scalar, in which case `h` is a square matrix. The default value for`hsize` is `[5 5]` and 0.5 for `sigma`.
- `h = fspecial('motion',len,theta)` returns a filter to approximate, once convolved with an image, the linear motion of a camera by`len` pixels, with an angle of `theta` degrees in a counterclockwise direction. The filter becomes a vector for horizontal and vertical motions. The default`len` is 9 and the default `theta` is 0, which corresponds to a horizontal motion of nine pixels.
- `h = fspecial('prewitt')` returns the 3-by-3 filter h (shown below) that emphasizes horizontal edges by approximating a vertical gradient. If you need to emphasize vertical edges, transpose the filter `h'`.

  [1 1 1
  0 0 0
  -1 -1 -1]

  To find vertical edges, or for*x*-derivatives, use`h'`.
- `h = fspecial('sobel')` returns a 3-by-3 filter h (shown below) that emphasizes horizontal edges using the smoothing effect by approximating a vertical gradient. If you need to emphasize vertical edges, transpose the filter `h'`.

  [1 2 1
  0 0 0
  -1 -2 -1]
- `h = fspecial('unsharp',alpha)` returns a 3-by-3 unsharp contrast enhancement filter. `fspecial` creates the unsharp filter from the negative of the Laplacian filter with parameter`alpha`. `alpha` controls the shape of the Laplacian and must be in the range 0.0 to 1.0. The default value for`alpha` is 0.2.

> **Note** Do not be confused by the name of this filter: an unsharp filter is an image sharpening operator. The name comes from a publishing industry process in which an image is sharpened by subtracting a blurred (unsharp) version of the image from itself.

Class Support

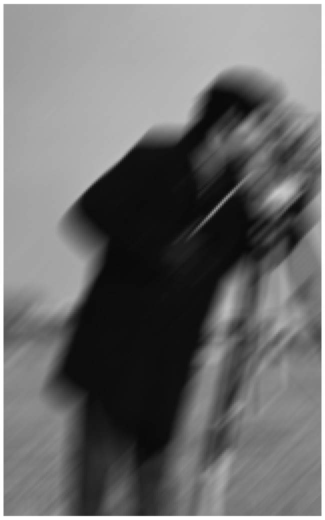
h is of class `double`.

Example

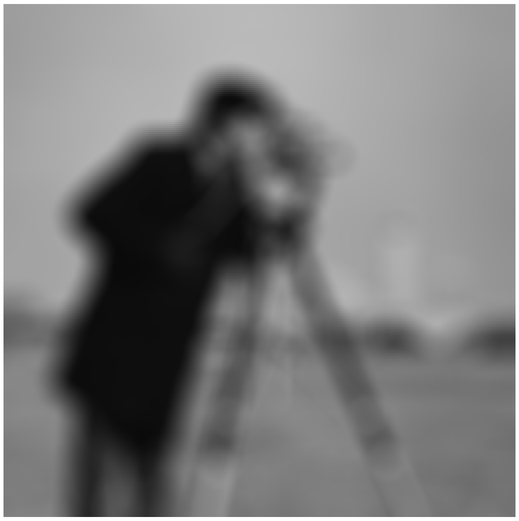
```
I = imread('cameraman.tif');
subplot(2,2,1);
imshow(I); title('Original Image');

H = fspecial('motion',20,45);
MotionBlur = imfilter(I,H,'replicate');
subplot(2,2,2);
imshow(MotionBlur);title('Motion Blurred Image');

H = fspecial('disk',10);
blurred = imfilter(I,H,'replicate');
subplot(2,2,3);
imshow(blurred); title('Blurred Image');

H = fspecial('unsharp');
sharpened = imfilter(I,H,'replicate');
subplot(2,2,4);
imshow(sharpened); title('Sharpened Image');
```

Image Courtesy of MIT

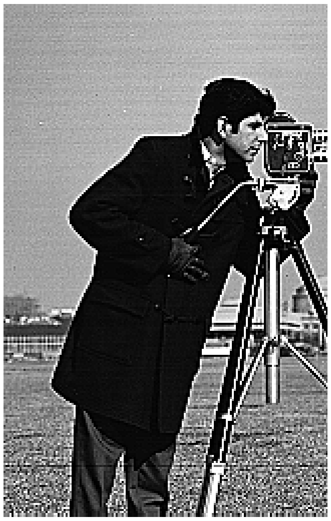
Original Image



Motion Blurred Imag



Blurred Image



Sharpened Image

Algorithms

`fspecial` creates Gaussian filters using

$$h_g(n_1, n_2) = e^{-(n_1^2 + n_2^2)/(2\sigma^2)}$$

$$h(n_1, n_2) = \frac{h_g(n_1, n_2)}{\sum_{n_1}\sum_{n_2} h_g}$$

`fspecial` creates Laplacian filters using

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

$$\nabla^2 = \frac{4}{(\alpha+1)}\begin{bmatrix} \frac{\alpha}{4} & \frac{1-\alpha}{4} & \frac{\alpha}{4} \\ \frac{1-\alpha}{4} & -1 & \frac{1-\alpha}{4} \\ \frac{\alpha}{4} & \frac{1-\alpha}{4} & \frac{\alpha}{4} \end{bmatrix}$$

`fspecial` creates Laplacian of Gaussian (LoG) filters using

$$h_g(n_1, n_2) = e^{-(n_1^2 + n_2^2)/(2\sigma^2)}$$

$$h(n_1, n_2) = \frac{(n_1{}^2 + n_2{}^2 - 2\sigma^2)h_g(n_1, n_2)}{2\pi\sigma^6 \sum_{n_1}\sum_{n_2} h_g}$$

`fspecial` creates averaging filters using

```
ones(n(1),n(2))/(n(1)*n(2))
```

`fspecial` creates unsharp filters using

$$\frac{1}{(\alpha+1)}\begin{bmatrix} -\alpha & \alpha-1 & -\alpha \\ \alpha-1 & \alpha+5 & \alpha-1 \\ -\alpha & \alpha-1 & -\alpha \end{bmatrix}$$

See Also

`conv2`, `edge`, `filter2`, `fsamp2`, `fwind1`, `fwind2`, `imfilter`

`del2` in the MATLAB Function Reference

◀ fsamp2 ftrans2 ▶