Common Image Filters

This reading describes the image filters that can be created with the `fspecial` function.

Table of Contents

Test Image

The filters in this reading will all be applied to the `rice.png` image.

```
img = imread("rice.png");
```

Blurring Filters for Noise and Detail Removal

Average Filter

This is the simplest blurring filter. It calculates the average value of the pixels covered by the filter.

The average filter has one optional input

- `hsize`, an integer (or vector of two integers for non-square filters) specifying the size of the filter matrix. The default value is 3.

```
fAverage = fspecial("average",[3,3])
imgAveraged = imfilter(img, fAverage);
montage({img, imgAveraged})
```

Disk Filter

This creates a disk-shaped averaging filter. The disk shape includes only physically close pixels, and tapers to zero near the edge of the disk.

The disk filter has one optional input

- `radius`, an integer specifying the radius of the disk-shaped filter. The filter is a square matrix of size `2*radius+1`. The default value is 5.

```
fDisk = fspecial("disk",3)
```

```
imgDisk = imfilter(img, fDisk);
montage({img, imgDisk})
```

Gaussian Filter

Gaussian filters are the preferred method of blurring an image before performing other processing steps, such as edge detection. The Gaussian filter is edge preserving, meaning smaller details can be removed without degrading the larger details, such as the edges of items.

The `imgausfilt` function creates and applies the filter in a single step. You need to provide two inputs:

- The standard deviation, sigma, for the guassian filter
- The filter size

Use the sliders to experiment with different size filters and different standard deviations.

```
sigma = 0.5;
filterSize = 3;
imgGaussFilt = imgaussfilt(img, sigma, FilterSize=filterSize);
montage({img,imgGaussFilt})
```

Blurring Filter for General Image Manipulation

Motion

This filter simulates an image blur caused by linear motion of the camera. The shape of the filter changes to accommodate the direction of motion, with horizontal and vertical motions resulting in row and column, vectors respectively.

The motion filter has two optional inputs

- `len`: an integer specifying the length of the filter matrix
- `theta`: the angle of the motion in degrees, measured counterclockwise from the horizontal. The default value is 0.

```
fMotion = fspecial("motion",9,0)
imgMotion = imfilter(img, fMotion);
montage({img, imgMotion})
```

Edge Detection Filters

These filters are used in edge detection algorithms. These are created and applied by the `edge` function, similar to the `imgaussfilt` described above. Creating and applying the filters as shown below allows you to investigate intermediate results.

Sobel

This filter is used in many edge detection algorithms. It is equivalent to a second-order accurate finite difference approximation to the image gradient. By default, the vertical gradient is calculated. The transpose of the filter is used to approximate the horizontal gradient.

The Sobel filter has no optional inputs.

```
fSobel = fspecial("sobel")
imgSobel = imfilter(img, fSobel);
montage({img, imgSobel})
```

Prewitt

This filter is also used in edge detection. It is equivalent to a first-order accurate finite difference approximation to the image gradient. By default, the vertical gradient is calculated. The transpose of the filter is used to approximate the horizontal gradient.

The Sobel filter has no optional inputs.

```
fPrewitt = fspecial("prewitt")
imgPrewitt = imfilter(img, fPrewitt);
montage({img, imgPrewitt})
```

Laplacian

The Laplacian filter combines the horizontal and vertical gradient approximations. With `alpha` = 0, it is equivalent to a second-order accurate finite difference approximation to the two-dimensional Laplacian operator.

The Laplacian filter has one optional argument

- `alpha`: the shape of the Laplacian. Takes values in the range [0, 1], with a default of 0.2.

```
fLaplacian = fspecial("laplacian",0.2)
imgLaplacian = imfilter(img, fLaplacian);
montage({img, imgLaplacian})
```

Blob Detection

Laplacian of Gaussian

Blob detection is a computer vision technique that generalizes image segmentation. It detects regions in an image that have similar properties, such as color or intensity. The Laplacian of Gaussian filter is a common first step in blob detection.

The Laplacian of Gaussian filter has two optional inputs

- `hsize`: an integer (or vector of two integers for non-square filters) specifying the size of the filter matrix. The default value is 3.
- `sigma`: the standard deviation of the Gaussian function used to create the filter. The default value is 0.5.

```
fLog = fspecial("log",[3,3],0.3)
```

```
imgLog = imfilter(img, fLog);
montage({img, imgLog})
```