Displaying Images Using the imshow Function

On this page…

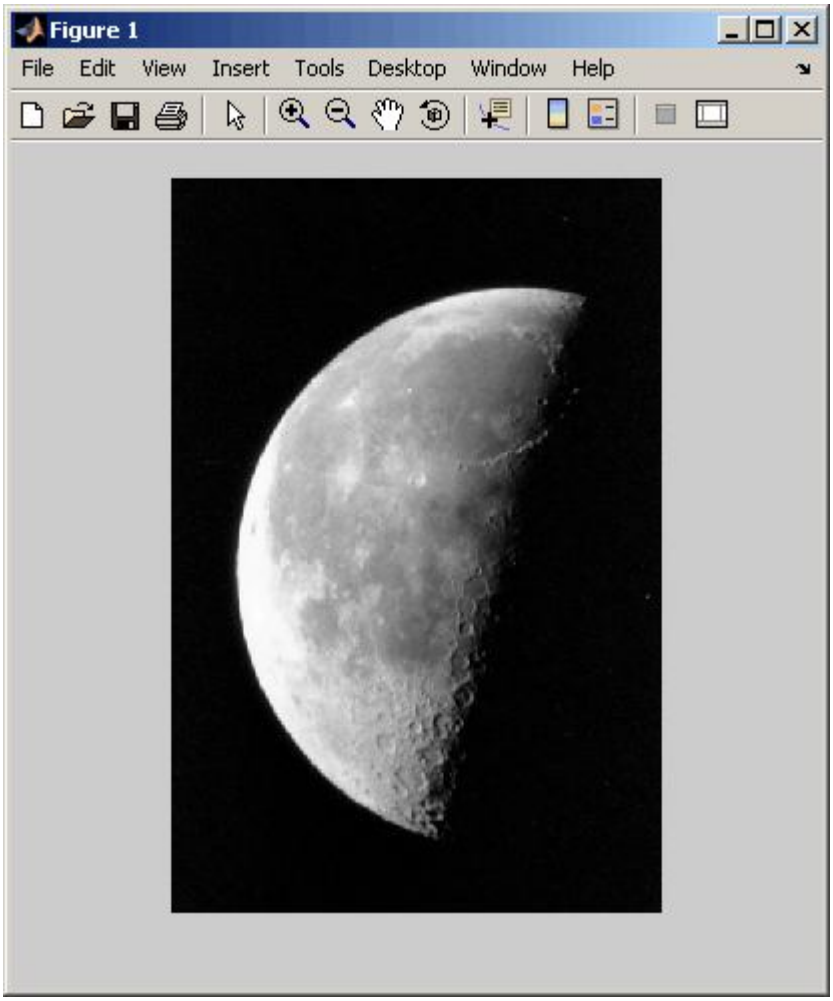
Overview

To display image data, use the `imshow` function. The following example reads an image into the MATLAB workspace and then displays the image in a MATLAB figure window.

```
moon = imread('moon.tif');
imshow(moon);
```

The `imshow` function displays the image in a MATLAB figure window, as shown in the following figure.

Image Displayed in a Figure Window by imshow



You can also pass `imshow` the name of a file containing an image.

```
imshow('moon.tif');
```

This syntax can be useful for scanning through images. Note, however, that when you use this syntax, imread does not store the image data in the MATLAB workspace. If you want to bring the image into the workspace, you must use the `getimage` function, which retrieves the image data from the current Handle Graphics image object. This example assigns the image data from `moon.tif` to the variable `moon`, if the figure window in which it is displayed is currently active.

```
moon = getimage;
```

For more information about using `imshow` to display the various image types supported by the toolbox, see Displaying Different Image Types.

Back to Top

Specifying the Initial Image Magnification

By default, `imshow` attempts to display an image in its entirety at 100% magnification (one screen pixel for each image pixel). However, if an image is too large to fit in a figure window on the screen at 100% magnification, `imshow` scales the image to fit onto the screen and issues a warning message.

To override the default initial magnification behavior for a particular call to `imshow`, specify the `InitialMagnification` parameter. For example, to view an image at 150% magnification, use this code.

```
pout = imread('pout.tif');
imshow(pout, 'InitialMagnification', 150)
```

`imshow` attempts to honor the magnification you specify. However, if the image does not fit on the screen at the specified magnification, `imshow` scales the image to fit and issues a warning message. You can also specify the text string `'fit'` as the initial magnification value. In this case, `imshow` scales the image to fit the current size of the figure window.

To change the default initial magnification behavior of `imshow`, set the `ImshowInitialMagnification` toolbox preference. To set the preference, open the Image Processing Toolbox Preferences dialog by calling `iptprefs` or by selecting **Preferences** from the MATLAB Desktop **File** menu.

When `imshow` scales an image, it uses interpolation to determine the values for screen pixels that do not directly correspond to elements in the image matrix. For more information, see Specifying the Interpolation Method.

Back to Top

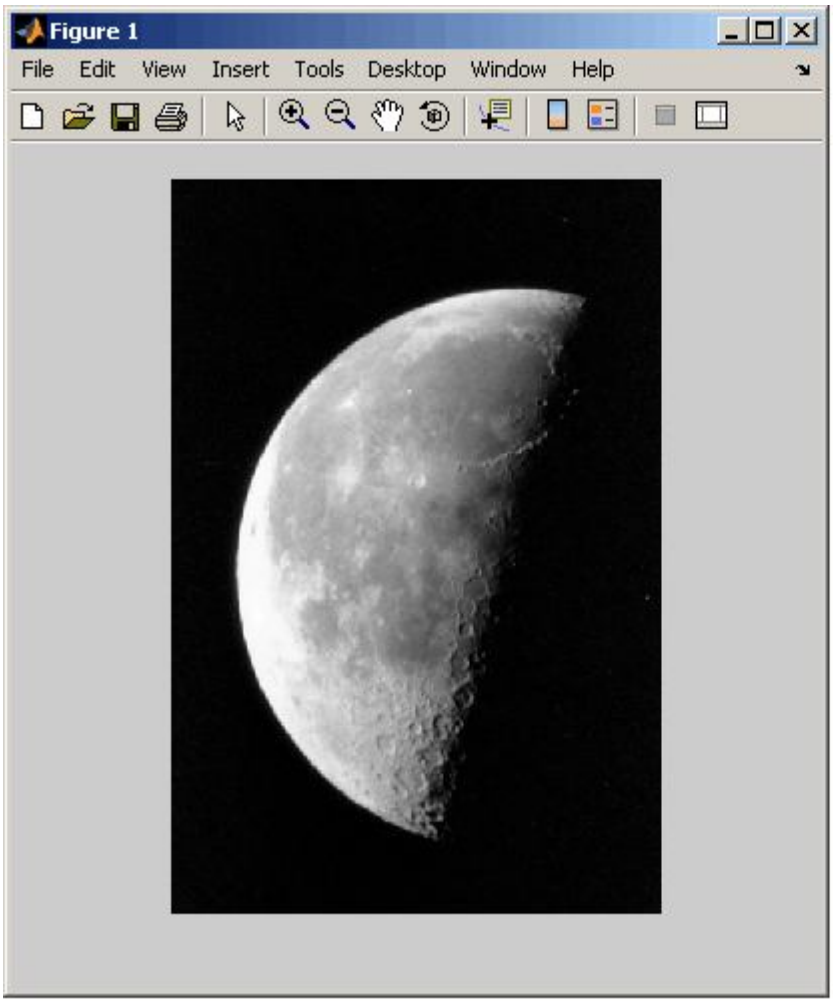
Controlling the Appearance of the Figure

By default, when `imshow` displays an image in a figure, it surrounds the image with a gray border. You can change this default and suppress the border using the `'border'` parameter, as shown in the following example.

```
imshow('moon.tif','Border','tight')
```

The following figure shows the same image displayed with and without a border.

Image Displayed With and Without a Border



'loose'

'tight'

The `'border'` parameters affect only the image being displayed in the call to `imshow`. If you want all the images that you display using `imshow` to appear without the gray border, set the Image Processing Toolbox `'ImshowBorder'` preference to `'tight'`. You can also use preferences to include visible axes in the figure. For more information about preferences, see `iptprefs`.

Back to Top

Displaying Each Image in a Separate Figure

The simplest way to display multiple images is to display them in separate figure windows. MATLAB does not place any restrictions on the number of images you can display simultaneously.

`imshow` always displays an image in the current figure. If you display two images in succession, the second image replaces the first image. To view multiple figures with `imshow`, use the `figure` command to explicitly create a new empty figure before calling `imshow` for the next image. For example, to view the first three frames in an array of grayscale images `I`,

```
imshow(I(:,:,:,1))
figure, imshow(I(:,:,:,2))
figure, imshow(I(:,:,:,3))
```

Back to Top

Displaying Multiple Images in the Same Figure

You can use the `imshow` function with the MATLAB `subplot` function or the MATLAB `subimage` function to display multiple images in a single figure window. For additional options, see Viewing Image Sequences.

> **Note** `imtool` does not support this capability.

Dividing a Figure Window into Multiple Display Regions

`subplot` divides a figure into multiple display regions. The syntax of `subplot` is

```
subplot(m,n,p)
```

This syntax divides the figure into an `m`-by-`n` matrix of display regions and makes the `p`th display region active.

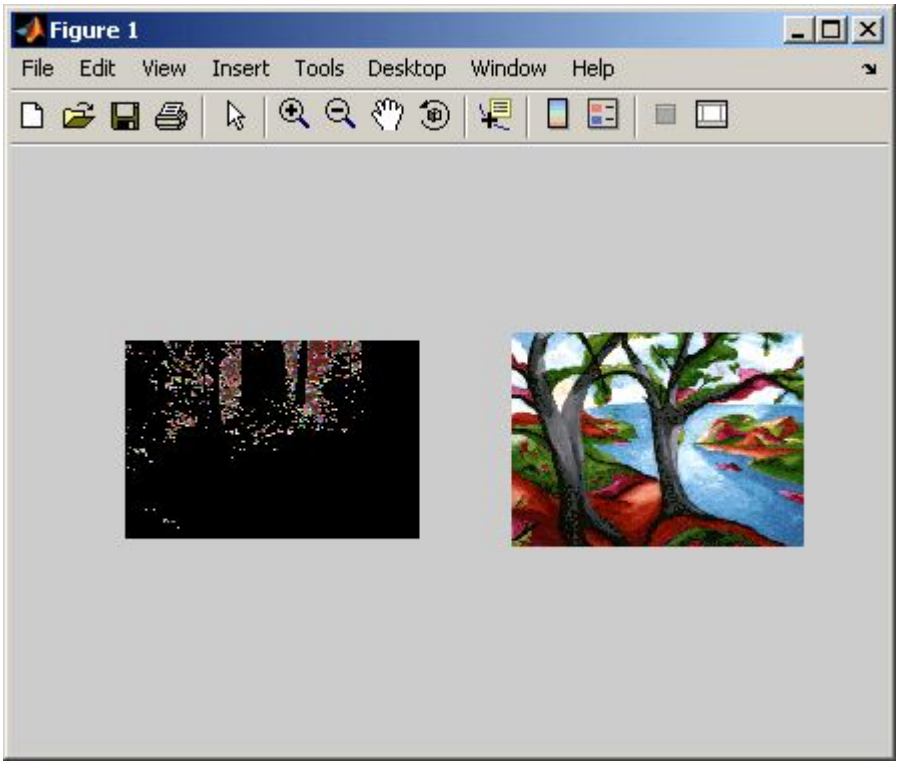
> **Note** When you use `subplot` to display multiple color images in one figure window, the images must share the colormap of the last image displayed. In some cases, as illustrated by the following example, the display results can be unacceptable. As an alternative, you can use the `subimage` function, described in Using the subimage Function to Display Multiple Images, or you can map all images to the same colormap as you load them.

For example, you can use this syntax to display two images side by side.

```
[X1,map1]=imread('forest.tif');
[X2,map2]=imread('trees.tif');
subplot(1,2,1), imshow(X1,map1)
subplot(1,2,2), imshow(X2,map2)
```

In the figure, note how the first image displayed, `X1`, appears dark after the second image is displayed.

Two Images in Same Figure Using the Same Colormap

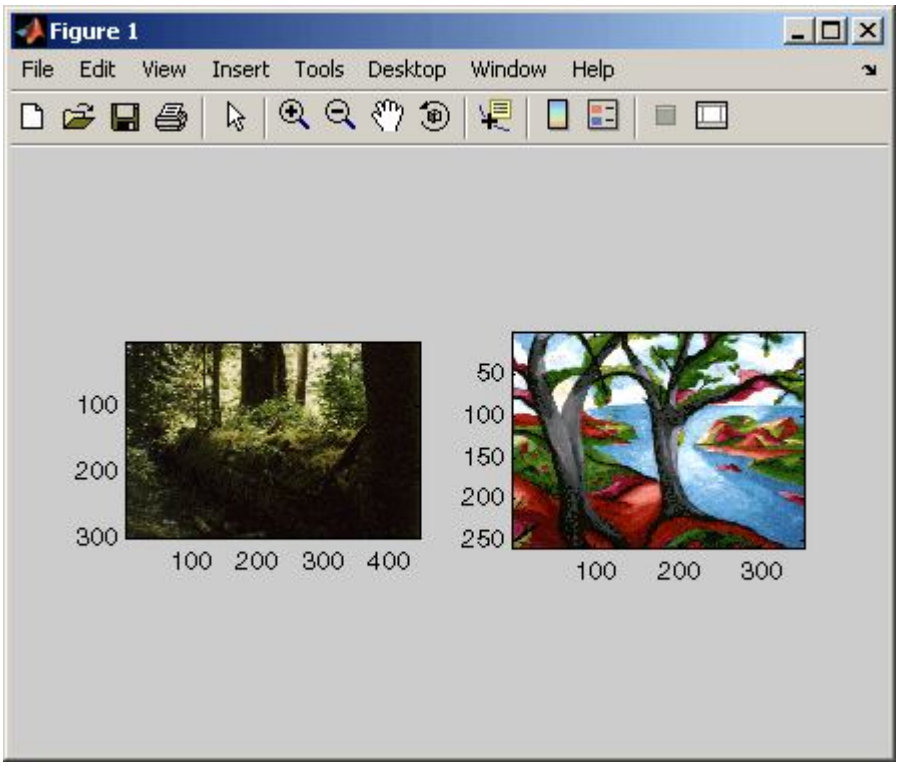


Using the subimage Function to Display Multiple Images

`subimage` converts images to truecolor before displaying them and therefore circumvents the colormap sharing problem. This example uses `subimage` to display the forest and the trees images with better results.

```
[X1,map1]=imread('forest.tif');
[X2,map2]=imread('trees.tif');
subplot(1,2,1), subimage(X1,map1)
subplot(1,2,2), subimage(X2,map2)
```

Two Images in Same Figure Using Separate Colormaps



Back to Top

Provide feedback about this page

Overview

Using the Image Tool to Explore Images