Image Processing Toolbox Release Notes

The "Image Processing Toolbox 5.0 Release Notes" on page 1-7 summarize the changes introduced in the latest version of the Image Processing Toolbox. The following topics are discussed in these Release Notes:

- "New Features" on page 1-8
- "Enhancements and Changes to Existing Features" on page 1-16
- "Major Bug Fixes" on page 1-27
- "Upgrading from an Earlier Release" on page 1-28
- "Known Issues" on page 1-29

The Image Processing Toolbox Release Notes also provide information about these recent versions of the product, in case you are upgrading from a previous release.

- "Image Processing Toolbox 4.2 Release Notes" on page 2-1
- "Image Processing Toolbox 4.1 Release Notes" on page 3-1
- "Image Processing Toolbox 4.0 Release Notes" on page 4-5
- "Image Processing Toolbox 3.2 Release Notes" on page 5-1
- "Image Processing Toolbox 3.1 Release Notes" on page 6-1
- "Image Processing Toolbox 2.2.2 Release Notes" on page 7-1

Printing the Release Notes

If you would like to print the Release Notes, you can link to a PDF version.

7
Image Processing Toolbox 5.0 Release Notes 7

New Features 8
New Image Exploration and Enhancement Tool 8
New Modular Interactive Tools 9
Hough Transform 12
Texture Analysis 13
New ICC Color Profile Export Function 13
New Demos 14

Enhancements and Changes to Existing Features 16
New DICOM Anonymizer Function 16
New Integer Lookup Table Function 16
New Toolbox Utility Functions 16
Updates to the imshow Function 17
Changes to Toolbox Preferences 21
Changes to Existing Functions 22
Performance Improvements 24
Improved Memory Usage 25
Obsoleted and Removed Functions 26

Major Bug Fixes 27

Upgrading from an Earlier Release 28

Known Issues 29
General Issues 29
Issues Specific to the Linux Platform 32
Issues Specific to the Macintosh Platform 32
1-1
Image Processing Toolbox 4.2 Release Notes 1-1

New Features 1-2
Enhanced DICOM support 1-2

Major Bug Fixes 1-3

Upgrading from an Earlier Release 1-4

2-1
Image Processing Toolbox 4.1 Release Notes 2-1

New Features 2-2
Reading and Writing Data with JPEG Lossless Compression 2-2
Reading ICC Profiles Embedded in TIFF Files 2-2
Reading and Writing L*a*b* Color Data 2-2

Major Bug Fixes 2-4
applycform Fixes 2-4
Compiling Spatial Transformation Functions 2-4
3-5
Image Processing Toolbox 4.0 Release Notes 3-5

New Features 3-6
New Image Viewer 3-6
Enhanced Color Space Functions 3-7
New Image Enhancement Methods 3-8
Enhanced DICOM Support 3-8
Fan Beam Projection Transforms 3-8
Boundary Tracing Functions 3-9
Unsigned Integer Lookup Tables 3-9
Optimized Image Arithmetic Functions 3-9
Performance Improvements 3-9

Minor Enhancements 3-10
Changes to Sample Images Included 3-11
Obsolete Functions 3-11

Major Bug Fixes 3-12
4-1
Image Processing Toolbox 3.2 Release Notes 4-1

New Features 4-2
Writing DICOM Files 4-2
Representing Binary Images 4-2
Changes to Existing Functions 4-3

Major Bug Fixes 4-4

Upgrading from an Earlier Release 4-5
Change to Data Type of Output Binary Images 4-5
Change to Interpretation of Input Images 4-5
Converting Binary Images to an Integer Data Type 4-5
5-1
Image Processing Toolbox 3.1 Release Notes 5-1

New Features 5-2
Morphology 5-3
Spatial Transformations 5-4
Image Registration 5-5
Integer Image Arithmetic 5-6
Integer Image Filtering 5-7
Deconvolution/Deblurring 5-7
Support for DICOM Files 5-8
Miscellaneous New Functions 5-8
New Demos 5-9

Major Bug Fixes 5-11
Version 3.1 Bug Fixes 5-11
6-1
Image Processing Toolbox 2.2.2 Release Notes 6-1

New Features 6-2
New Demo 6-2
Support For Function Handles 6-2
Documentation Enhanced 6-2

Major Bug Fixes 6-3
imshow Fixes 6-3
bwlabel Segmentation Violation Eliminated 6-3
dilate And erode Return Correct Answers 6-3
freqz2 Fixes 6-3
fspecial Function’s ‘LoG’ Option 6-3
Improved Display for imcrop, improfile, and roipoly 6-4

Image Processing Toolbox 5.0 Release Notes

"New Features" on page 1-8	Provides brief descriptions of the major new features added to the toolbox in Version 5.
"Enhancements and Changes to Existing Features" on page 1-16	Describes changes and new features added to existing toolbox functions.
"Major Bug Fixes" on page 1-27	Describes major bug fixes since the last release.
"Upgrading from an Earlier Release" on page 1-28	Describes issues you might encounter upgrading from a previous release.
"Known Issues" on page 1-29	Describes known problems with this release.

New Features

Version 5 of the Image Processing Toolbox includes the following new features.

- Open-architecture image exploration and enhancement tool
- Modular interactive tools for image exploration and manipulation that can also be combined into custom applications
- Texture analysis functions
- Hough Transform functions
- ICC profile export
- Additional `int16` and `single` support
- Two new IPT demos

For information about updates to existing toolbox functions, see "Enhancements and Changes to Existing Features" on page 1-16.

New Image Exploration and Enhancement Tool

The Image Processing Toolbox includes a new open-architecture, image exploration, and enhancement tool, called the Image Tool.

The Image Tool replaces the Image Viewer, providing all the display and exploration capabilities of its predecessor. For example, you can use the Image Tool to display an image, view general information about the image, get information about individual pixels or regions of pixels in the image, and navigate large images using the Overview navigation window, scroll bars, and magnification tools.

In addition, the Image Tool introduces several new tools, including an Adjust Contrast tool which you can use to adjust the brightness and contrast of an image interactively, a Choose Colormap tool which allows you to change the colormap for indexed and intensity images to any of MATLAB's colormap functions or to a user-defined colormap variable, menus that expose all toolbar feature plus tools to Import from Workspace, Export to Workspace, and Open from a file.

Unlike the Image Viewer, the Image Tool is built using standard features of MATLAB Handle Graphics. This enables the Image Tool to provide access the image being displayed using standard Handle Graphics techniques. For example, you can use annotations and overlay vector graphics on images displayed in the Image Tool. You can use `imtool` as an example to follow or as a base to use to create your own application.

To start the Image Tool, use the `imtool` function

```
imtool('moon.tif')
```

New Modular Interactive Tools

The toolbox includes several new modular interactive tools that you can activate from the command line and use with images displayed in a MATLAB figure window, called the *target image* in this documentation. The tools are modular because they can be used independently or in combination to create custom graphical interfaces for image processing applications. The Image Tool uses these modular tools.

The following table lists the modular tools available with the functions you use to create them.

Modular Tool	**Description**
Adjust Contrast tool	Display histogram of image pixel values in the target image and enable interactive adjustment of contrast and brightness by manipulating the range. Use `imcontrast` to create the tool in a separate figure window and associate it with an image.
Display Range tool	Displays a text string identifying the display range values of the target image. Use `imdisplayrange` to create the tool, associate it with an image, and embed it in a figure or uipanel.

Modular Tool	Description
Image Information tool	Displays basic information about an image along with any metadata the image might contain. Use `imageinfo` to create the tool. To collect image information, use `imattributes`, `imfinfo`, and `dicominfo`.
Magnification tool	Creates a text edit box containing the current magnification of the target image. Users can type in the desired magnification. Use `immagbox` to create the tool, associate it with an image, and embed it in a figure or uipanel. **Note**: The target image must be contained in a scroll panel.
Overview tool	Displays the target image in its entirety with the portion currently visible in the scroll panel outlined by a rectangle superimposed on the image. Moving the rectangle changes the portion of the target image that is currently visible in the scroll panel. Use `imoverview` to create the tool in a separate figure window and associate it with an image. Use `imoverviewpanel` to create the tool in a uipanel that can be embedded within another figure uipanel. **Note:** The target image must be contained in a scroll panel.

Modular Tool	Description
Pixel Information tool	Displays information about the pixel the mouse is over in the target image. Use `impixelinfo` to create the tool, associate it with an image, and display it in a figure or uipanel. If you want to display only the pixel values, without the text label, use `impixelinfoval`.
Pixel Region tool	Display pixel values for a specified region in the target image. Use `impixelregion` to create the tool in a separate figure window and associate it with an image. Use `impixelregionpanel` to create the tool as a uipanel that can be embedded within another figure or uipanel.
Scroll Panel tool	Display target image in a scrollable uipanel with scroll bars. Use `imscrollpanel` to add a scroll panel to an image displayed in a figure window.

Modular Tool Utility Functions

In addition to the modular tools listed above, the toolbox includes a number of new utility functions that make GUI building easier. The following table lists these utility functions in alphabetical order. The tools reside in the `$MATLAB/toolbox/images/imuitools` directory, where `$MATLAB` represents your MATLAB installation directory.

`getimagemodel`	Retrieve imagemodel objects from image handles
`imattributes`	Return information about image attributes
`imgca`	Get handle to current image axes

`imgcf`	Get handle to current image figure
`imgetfile`	Image Open File dialog box
`imhandles`	Get all image handles
`impositionrect`	Create position rectangle
`iptaddcallback`	Add function handle to callback list
`iptcheckhandle`	Check validity of handle
`iptgetapi`	Get Application Programmer Interface from a handle
`ipticondir`	Directories containing IPT and MATLAB icons
`iptremovecallback`	Delete function handle from callback list
`iptwindowalign`	Align figure windows

Hough Transform

The toolbox now includes three new functions that provide support for the Hough transform.

- `hough`
- `houghpeaks`
- `houghlines`

The `hough` function implements the Standard Hough Transform (SHT). The Hough transform is designed to detect lines, using the parametric representation of a line:

```
rho = x*cos(theta) + y*sin(theta).
```

The variable `rho` is the distance from the origin to the line along a vector perpendicular to the line. `theta` is the angle between the x-axis and this vector. The `hough` function generates a parameter space matrix whose rows and columns correspond to these `rho` and `theta` values, respectively.

The `houghpeaks` functions finds peak values in this space, which represent potential lines in the input image.

The `houghlines` function finds the endpoints of the line segments corresponding to peaks in the Hough transform and it automatically fills in small gaps.

Texture Analysis

The toolbox now supports a set of functions that you can use for texture analysis. These functions include

- `entropy` — Calculates the entropy of an intensity image
- `entropyfilt` — Calculates the local entropy of an intensity image
- `graycomatrix` — Computes the gray-level co-occurrence matrix from an image
- `graycoprops` — Extracts properties from a gray-level co-occurrence matrix
- `rangefilt` — Calculates the local range of an image
- `stdfilt` — Calculates the standard deviation of an image

Texture analysis refers to the characterization of regions in an image by their texture content. Texture analysis attempts to quantify intuitive qualities described by terms such as rough, silky, or bumpy in the context of an image. In this case, the roughness or bumpiness refers to variations in the brightness values, or gray levels.

Some of the most commonly used texture measures are derived from the Grey Level Co-occurrence Matrix (GLCM). The GLCM is a tabulation of how often different combinations of pixel brightness values (grey levels) occur in a pixel pair in an image. You can use the `graycomatrix` function to create a GLCM and then use `graycoprops` to extract feature information (e.g. contrast, correlation, energy, and homogeneity) from the GLCM.

The texture analysis support also includes several new functions that filter using standard statistical measures, such as range, standard deviation, and entropy. (Entropy is a statistical measure of randomness.) To see an example of using these filtering functions, view the "Texture Segmentation Using Texture Filters" demo. Use the `iptdemos` function to access toolbox demos.

New ICC Color Profile Export Function

The toolbox includes a new function, `iccwrite`, that you can use to write International Color Consortium (ICC) color profile data to a file. ICC profiles provide color management systems with the information necessary to convert color data between native device color spaces and device independent color spaces, called Profile Connection Space (PCS).

The toolbox also includes a function, `isicc`, that can verify if the data is a valid ICC profile. The `iccwrite` function can output profile data in accordance with both version 2 (ICC.1:2001-04) and version 4 (ICC.1:2001-12) of the ICC specification. For more information about the changes between version 2 and version 4 of the specification, go to the ICC Web site `www.color.org`.

In addition, `iccread`, `makecform`, and `applycform` all now work with Version 4 profiles as well as Version 2 profiles, and with color profiles with more than four channels.

Changes to iccread

To add color profile export support, and to accommodate Version 4 of the specification, there are some differences in the way data is returned by `iccread`. In the structure returned by `iccread` some of the fields that contained text strings in previous releases are now structures. Accessing the text string in the fields requires an additional level of dereferencing. For example, the value of the Description field was a text string.

```
P.Description

ans =

sRGB IEC61966-2.1 991203
```

Now, the value of this field is a structure with two fields: String and Optional. To access the text string, you must access the String field in this structure.

```
P.Description.String

ans =

sRGB IEC61966-2.1 991203
```

New Demos

The toolbox includes two new demos:

- Texture Segmentation Using the Text Filters
- Analyzing a Multispectral LANDSAT Image

In addition, some of the existing demos have been reorganized into new categories. The Color Segmentation and Morphological Segmentation demos have been moved to the new Image Segmentation category.

Enhancements and Changes to Existing Features

In addition to the features described in "New Features" on page 1-8, the following sections describe enhancements and other changes to existing toolbox functions.

- "New DICOM Anonymizer Function" on page 1-16
- "New Integer Lookup Table Function" on page 1-16
- "New Toolbox Utility Functions" on page 1-16
- "Updates to the imshow Function" on page 1-17
- "Changes to Toolbox Preferences" on page 1-21
- "Changes to Existing Functions" on page 1-22
- "Performance Improvements" on page 1-24
- "Improved Memory Usage" on page 1-25
- "Obsoleted and Removed Functions" on page 1-26

New DICOM Anonymizer Function

The toolbox now includes a new function, `dicomanon`, that can remove all confidential data from a DICOM file.

New Integer Lookup Table Function

The toolbox now includes a new function, intlut, that can convert arrays of `uint8`, `uint16`, and `int16` integer values using a lookup table.

New Toolbox Utility Functions

The toolbox includes several new utility functions that can help with input argument parsing. These functions check the validity of arguments and issues standard error messages, if the argument is invalid. The toolbox includes other utility functions to get the dynamic range of an image and convert a positive integer to an ordinal string.

The following table lists these functions in alphabetical order. The functions reside in the `$MATLAB/toolbox/images/iptutils` directory, where `$MATLAB` represents your MATLAB installation directory.

`getrangefromclass`	Check dynamic range of image
`iptcheckconn`	Check validity of connectivity argument
`iptcheckinput`	Check validity of input arguments
`iptcheckmap`	Check validity of colormap argument
`iptchecknargin`	Check number of arguments
`iptcheckstrs`	Check validity of string arguments
`iptnum2ordinal`	Convert positive integer to ordinal string

Updates to the imshow Function

There have been several changes to the behavior and syntaxes supported by the `imshow` function. The following sections detail these changes.

- "Filenames No Longer Displayed as a Title in Figure Window Border" on page 1-17
- "Nondefault Spatial Coordinate Syntax Changed" on page 1-18
- "Initial Image Magnification DISPLAY_OPTION Syntax Changed" on page 1-18
- "New Image Scaling Algorithm Determines Image Display" on page 1-19
- "New Image Display Range Syntax" on page 1-20
- "Number-of-Gray-levels Syntax Obsoleted" on page 1-20

Filenames No Longer Displayed as a Title in Figure Window Border

The `imshow` function when called with a filename

```
imshow(filename)
```

no longer automatically displays the filename as a title in the figure window border. In previous releases, the gray space was proportional to image size. Often, this left too little space for small images and too much for big images, making positioning the title in the figure window problematic.

Changes to the algorithm `imshow` uses to calculate the border size when the `'ImshowBorder'` preference is set to `'loose'` should allow plenty of room for title, axes tick marks, and axes labels in a way that is independent of the image size. (See “New Image Display Range Syntax” on page 1-20 for more information.)

Nondefault Spatial Coordinate Syntax Changed

The `imshow` syntax for specifying nondefault spatial coordinates has changed to use parameter/value pairs. The old syntax

```
imshow(x,y,...)
```

is now

```
imshow(...,'XData',x,'YData',y)
```

`imshow` still accepts the old syntax, automatically translating it to the new syntax and issuing the following warning.

```
IMSHOW(x,y,...) is an obsolete syntax. Use
IMSHOW(...,'XData',x,'YData',y) instead.
```

Initial Image Magnification DISPLAY_OPTION Syntax Changed

The `imshow` `display_option` syntax is obsolete. The old syntax,

```
imshow(...,display_option)
```

where `display_option` was either `'truesize'` or `'notruesize'` has been replaced by the following parameter/value pair syntax.

```
imshow(...,'InitialMagnification',initial_mag)
```

As the value, you can specify a numeric magnification percentage value or the text string `'fit'`.

imshow still accepts the old syntax, automatically translating the old display_option values to the new syntax, as shown in the following table. The table also includes the text of the warning message issued by imshow.

Old Syntax	Automatic Translation to New Syntax

`imshow(...,'truesize')`	`imshow(...,'InitialMagnification',100)`
	where 100% magnification specifies the same one-image-pixel to one-screen-pixel magnification achieved by the `'truesize'` option. imshow also issues the following warning message.
	`Warning: IMSHOW(...,'truesize') is an obsolete syntax.`
	`Use IMSHOW(...,'InitialMagnification',100) instead.`

`imshow(...,'notruesize')`	`imshow(...,'InitialMagnification','fit')`
	where the behavior is similar to the `'notruesize'` behavior. imshow issues the following warning message.
	`Warning: IMSHOW(...,'notruesize') is an obsolete syntax.`
	`Use IMSHOW(...,'InitialMagnification','fit') instead.`

New Image Scaling Algorithm Determines Image Display

When you specify a numeric magnification percentage value, imshow performs the following processing to determine how to display the image.

1 Calculate the gutter dimensions (gray space around image) and figure window decoration dimensions.

Note imshow only has to calculate these dimensions once per MATLAB session because they are independent of the image being displayed and depend on the system running the display function.

---

2 Determine the screen dimensions for the image at the requested display magnification.

3 Add the results of Step 1 and Step 2.

4 Determine if the figure fits on the screen at the specified magnification.

   If the image in the figure window fits on the screen, display it.

   If the image in the figure window does not fit on the screen, `imshow` reduces the magnification until the image fits on the screen and displays it. `imshow` issues a warning message that the image has been scaled, including the magnification value in the message.

Note `imshow` now uses the same magnification increments as the zoom tool (100%, 67%, 50%, 33%,...).

New Image Display Range Syntax

The `imshow` syntax in which you specify the display range of the image

```
imshow(I,[LOW HIGH])
```

has been augmented to use a parameter/value pair syntax

```
imshow(...,'DisplayRange',[LOW HIGH])
```

`imshow` still accepts the old syntax.

Note, however, that with the new parameter/value syntax, you can specify the target image as a filename, as in the following example.

```
imshow(filename,'DisplayRange'[LOW HIGH])
```

If you want to specify the display range for an intensity image specified by a filename, you must use the `'DisplayRange'` parameter.

Number-of-Gray-levels Syntax Obsoleted

The `imshow` syntax in which you specified the number of gray levels used to display the image

```
imshow(I,N)
```

has been obsoleted. If you use this syntax, `imshow` outputs the following warning message:

```
IMSHOW(I,N) is an obsolete syntax. Your intensity image will be
displayed using 256 shades of gray.
```

Changes to Toolbox Preferences

The names of several Image Processing Toolbox preferences have changed in Version 5. The following table lists these preferences with the new name, where available.

Obsolete Preference	New Preference
`'ImshowTruesize'`	`'ImshowInitialMagnification'`
`'ImviewInitialMagnification'`	`'ImtoolInitialMagnification'`

`'TruesizeWarning'`	No replacement.
	
	Use the MATLAB `warning` function to control whether you see the warning that appears when displaying a large image with `imshow`.
	
	`warning off Images:initSize:adjustingMag`
	
	`warning on Images:initSize:adjustingMag`

Changes to Existing Functions

The following tables lists toolbox functions that have been changed in Version 5.

Function	Enhancement
`blkproc`	The `fun` argument must be specified as a function handle; it can no longer be specified as an `inline` function or text string. The syntax `blkprc(...,fun,P1,P2...)` is no longer supported. Use an anonymous function instead.
`colfilt`	See the entry for `blkprc` in this table. It describes the change made to this function.
`deconvblind`	See the entry for `blkprc` in this table. It describes the change made to this function.
`dicomread`	No longer supports the `'Dictionary'` and `'Raw'` parameters.
`edge`	Supports new syntaxes that return the gradient components when gradient-based methods are used (Sobel, Prewitt, Roberts).
`getimage`	No longer accepts a handle to a texture-mapped surface as an input argument or returns a texture-mapped surface as an image. `getimage` now only returns data for image objects. `getimage` also returns a new flag identifying a binary image.
`graythresh`	Now implements Otsu's class separability metric, which measures the effectiveness of a threshold computation. For this metric, the lower bound of zero represents a monotone image and the upper bound of 1 represents a two-valued image.
`ifanbeam`	See the entry for `iradon` in this table.

Function	Enhancement
`im2bw`	Might produce different results when used in conjunction with the `graythresh` function. The `graythresh` function uses Otsu's method which, by definition, splits the pixels in an image into two classes based on the calculated threshold. All the pixels up to and including the pixels equal to the threshold belong to the first class and the remaining pixels belong to the other class. To match this algorithm, the `im2bw` function now uses the greater-than operator (>) instead of the greater-than-or-equal operator(>=). Thus `im2bw` may produce different results from previous releases.
`imfilter`	Now automatically detects and exploits filter separability to speed up the filter computation. This change in the computational algorithm may result in small differences in the output values because of a combination of floating-point round-off differences and integer rounding effects.
`iradon`	Now supports all interpolation types supported by interp1. Previously, only `'linear'`, `'nearest'`, and `'spline'` were supported.
`makelut`	See the entry for `blkprc` in this table. It describes the change made to this function.
`nlfilter`	See the entry for `blkprc` in this table. It describes the change made to this function.
`qtdecomp`	See the entry for `blkprc` in this table. It describes the change made to this function.

Function	Enhancement
`regionprops`	Now calculates the perimeter of each labeled region in a label matrix.
`roifilt2`	The `fun` argument must be specified as a function handle; it can no longer be specified as an `inline` function or text string. The syntax `roifilt(...,fun,P1,P2...)` is no longer supported. Use anonymous function instead.

Performance Improvements

The performance of several existing toolbox functions has been improved in this release, including:

- `regionprops` (6 times faster than previous version)
- `imfilter` (faster for separable filters)

The `dicomread` function has a 5 to 10 times performance improvement over the previous version.

The performance of the following morphology functions has been improved when used with large rectangular structuring elements.

- `imdilate`
- `imerode`

Functions that call `imdilate` and `imerode`, specifying large rectangular structuring elements might also see a speed improvement, i.e., `imopen`, `imclose`, `imtophat`, and `imbothat`.

The performance of the following image type conversion functions and color space conversion functions has been improved with data of classes `uint8` and `uint16` by using the `intlut` function.

- `imadjust`
- `imcomplement`
- `ind2gray`
- `rgb2gray`
- `rgb2ntsc`
- `rgb2ycbcr`

- `rcbcr2rgb`

Improved Memory Usage

The memory usage of the following deblurring functions has been improved by clearing temporary variables as the algorithms proceed.

- `deconvblind`
- `deconvlucy`
- `deconvreg`
- `deconvwnr`

edgetaper

The `edgetaper` function now uses single precision in its calculations for images with an integer data type in order to reduce memory useage. This means the `edgetaper` returns an answer that is slightly different that the answer returned by previous version of the toolbox. If you want `edgetaper` to use double precision for integer data types, convert your image to `double` before calling `edgetaper`.

improfile

The `improfile` function used to cast input images to `double` if it met these criteria:

1. logical array
2. non-double image using an interpolation method other than nearest-neighbor.

It now casts them to single to reduce memory overhead. In these cases, the output of `improfile` might differ slightly from previous versions of the toolbox.

Obsoleted and Removed Functions

The following tables lists toolbox functions that have been obsoleted or removed in Version 5.

Function	Enhancement
`dctmtx2`	This function, which was obsoleted in previous release, has been removed from the toolbox.
`im2mis`	This function, which was obsoleted in previous release, has been removed from the toolbox.
`imview`	This function is obsolete. It now warns and passes its arguments to the `imtool` function.
`imzoom`	This function, which was obsoleted in previous release, has been removed from the toolbox.
`uintlut`	This function, which only accepted `uint8` and `uint16` data now warns and passes arguments to the `intlut` function which also handles `int16` data.

Major Bug Fixes

The Image Processing Toolbox 5 includes the following bug fix.

Canny Edge Detector Handles Constant-valued (flat) Images

The Canny edge detector now accepts monotone images, also called constant-value images or flat images. Instead of issuing an error when an input image was single-valued (monotone), the function now returns an output image containing all zeros, indicating that no edges were found.

imcomplement returns correct answer for signed integer input

imcomplement was incorrectly calculating the complement of an image with signed integer data type. This problem has been fixed.

imlincomb correctly handles int16 data combined with scalar having a 0.5 fractional part

imlincomb was giving an incorrect answer when combining int16 data with a scalar that had 0.5 as a fractional part. This problem has been fixed.

iradon introduced a vertical shift of one pixel

The iradon function now correctly calculates the vertical origin of the input projections. Previously the calculated origin was off by one for inputs with an even number of projection samples. The effect of this problem could be observed by computing the Radon transform (using radon) of a test pattern containing horizontal edges, followed by computing the inverse Radon transform (using iradon). Careful comparison of the test pattern with the output of iradon showed a vertical misregistration of one pixel.

imshow now correctly renders indexed images with colormaps having more than 256 colors

On the Windows platform, imshow now sets the figure's 'Renderer' property to 'zbuffer' for indexed images with associated colormaps having more than 256 entries, so they now render correctly. Previously they would appear black.

Upgrading from an Earlier Release

Note The issues mentioned here are all described in more detail in previous sections.

- Changes to `imshow` syntax and to toolbox preferences. Old syntaxes and preferences will still work as expected, but they will now warn.
- The function `imview` is now obsolete. It warns and calls `imtool`.
- The functions `edgetaper`, `im2bw`, `imfilter`, and `improfile` may give slightly different answers from previous releases in certain cases.
- The DICOM data dictionary changed so that some words are different now than in previous releases. This is due to updates in the data dictionary as defined by the DICOM standards committee. Note: This actually changed for version 4.2 of the toolbox but was not release noted at that time.

Known Issues

There are some known issues with this beta release.

- "General Issues" on page 1-29
- "Issues Specific to the Linux Platform" on page 1-32
- "Issues Specific to the Macintosh Platform" on page 1-32

General Issues

The following are known issues.

- "Image Tool Cursor Interactions" on page 1-30
- "imagemodel, imageinfo, and imattributes Return Incorrect Data Type" on page 1-31
- "Image Information Tool Cannot Be Resized" on page 1-31
- "Pixel Region Affects Positioning of Image in Subplot" on page 1-30

imoverview, imoverviewpanel, imscrollpanel and imtool Performance with Large Intensity Images

There is a performance problem with the Image Tool and its related navigation tools when used with large intensity images.

Possible workarounds:

- For images of any class, the image can be treated as an RGB image via `repmat`.

  ```
  I = imread('concordorthophoto.png');
  imtool(repmat(I,[1 1 3]))
  ```

- For images of class `uint8` or `uint16`, the image can be treated as an indexed image.

  ```
  I = imread('concordorthophoto.png');
  n = double(intmax(class(I)));
  map = gray(n);
  imtool(I,map)
  ```

Both workarounds will make `imcontrast` unusable and pixel reporting will be for the wrong image type. The first workaround uses more memory than the second workaround.

Image Tool Cursor Interactions

If you run the Image Tool and activate one of the navigational tools (zoom in, zoom out, pan) prior to turning on the Pixel Region tool, the navigational tool will be turned off and will be replaced by the mouse behavior of `imcontrast`. Conversely, if you run imtool, start the Adjust Contrast tool, and then start a navigation tool, the mouse behavior of `imcontrast` is turned off.

Adjust Contrast Tool Does Not React to Changes in CLim, CData, or CDataMapping

If you turn on the Adjust Contrast tool using imcontrast or imtool, the then set the axes `CLim`, or the image `CData` or `CDataMapping` properties, the tool does nothing to react. Once you click on the tool, it updates based on changes to the `CLim` values.

Image in Pixel Region tool doesn't update when you use the Adjust Contrast tool

If you use `imtool` to display a grayscale image, and you launch the Pixel Region tool, and then you adjust the contrast using the Adjust Contrast tool, the pixel colors in the Pixel Region tool don't update properly.

Pixel Region Affects Positioning of Image in Subplot

If you call impixelregion after plotting images in subplots, the position of the subplot containing the target image gets messed up.

impixelregionpanel Might Interfere with ButtonDown events in Parent Figure

If you create an `impixelregionpanel` and another `imscrollpanel` in the same figure, the `impixelregionpanel` can interfere with `ButtonDown` events throughout the figure. This can, for example, prevent the user from clicking and dragging a position rectangle located in an `imscrollpanel` elsewhere in the figure.

Workaround: After creating all of the panels needed for your GUI, execute this code:

```
  uistack(hPixelRegionPanel, 'bottom')
```

where `hPixelRegionPanel` is the handle returned by `impixelregionpanel`.

imagemodel, imageinfo, and imattributes Return Incorrect Data Type

The `imagemodel`, `imageinfo`, and `imattributes` functions return class `double` for `int16` or `single` images. These functions determine the data type by querying the image object's CData. For `int16` and `single` images, the image object converts its CData to class `double`.

For example,

```
h = imshow(int16(ones(10)));
class(get(h,'CData'));
```

returns `'double'`. Consequently, `imageinfo` and `imattributes` would return a class type of `double` and calling the image model object's `getClassType` method returns double.

Image Information Tool Cannot Be Resized

The Image Information Tool (`imageinfo`) cannot be resized.

Compiling functions that depend on the `imagemodel` may cause warning messages. Running compiled versions of `imtool` and/or some of the functions in the `imuitools` directory may generate the following warning messages:

```
Warning: Objects of graphics.linkprop class exist - not clearing
this class
or any of its super-classes.
Warning: An object instance still exists.
Use the objectdirectory command to see a count of existing
instances.
```

These warning messages will only appear when you exit your application, and the messages can be safely ignored.

Choose Colormap Tool Does Not Save Selection in Compiled Applications

The colormap selection made in the Choose Colormap tool does not persist when running a compiled version of the Image Tool. That is, the colormap selected from the Choose Colormap tool is lost when you click the **OK** button instead of being saved.

Issues Specific to the Linux Platform

On the Linux platform, `imageinfo` sometimes is blank.This is due to a known issue in MATLAB that will be fixed in an upcoming release.

Alt+Click Does Not Work for Opposite Zoom in imtool

On the Linux platform, if you are using the Image Tool and choose the zoom in or zoom out tool, Alt+click should zoom in the opposite direction from the currently selected tool.

Issues Specific to the Macintosh Platform

The following issues are unique to Macintosh systems.

- "Image Tool Limitations" on page 1-32
- "Image Information Tool Not Supported" on page 1-32
- "Resize Behavior of impixelinfo and imdisplayrange" on page 1-32
- "Difference in Scroll bar Behavior" on page 1-33
- "Pixel Region Tool Slow on Macintosh Systems" on page 1-33

Image Tool Limitations

On Macintosh systems, the Image Tool (`imtool`) has the following limitations:

- The Magnification tool is not supported.
- The Image Information toolbar button is not functional.

Image Information Tool Not Supported

On Macintosh systems, the `imageinfo` function is not supported because it require Java figures, which are not available on this platform.

Resize Behavior of impixelinfo and imdisplayrange

On Macintosh systems, the `impixelinfo` and `imdisplayrange` functions do not resize correctly because they require Java figures, which are not available on this platform.

Difference in Scroll bar Behavior

On Macintosh systems, if you create a Scroll Panel (`imscrollpanel`) and drag the scroll bars, the image does not update until you release the mouse from the drag. On other platforms, the image updates continuously during the drag.

Pixel Region Tool Slow on Macintosh Systems

On Macintosh systems, the `impixelregion` and `impixelregionpanel` are slower than on other platforms.

Alt+Click Does Not Work for Opposite Zoom in imtool

On Macintosh systems, if you are using the Image Tool and choose the zoom in or zoom out tool, Alt+click should zoom in the opposite direction from the currently selected tool.

Image Processing Toolbox 4.2 Release Notes

New Features 1-2
Enhanced DICOM support 1-2

Major Bug Fixes 1-3

Upgrading from an Earlier Release 1-4

New Features

The Image Processing Toolbox 4.2 includes the following new features, added since Version 4.1.

Enhanced DICOM support

- The `dicomwrite` function can now write data in any modality that can be read using `dicomread`. Note, however, that when writing data in these modalities, `dicomwrite` does not verify the data or check to see if the correct amount of data is written.
- The toolbox can now read and write private metadata, even if they are not defined in the DICOM data dictionary. When private metadata is not in the data dictionary, `dicomread` uses generic names for the metadata fields, rather than the descriptive names available to attributes defined in the data dictionary.
- You can create a custom data dictionary that contains definitions of your private metadata, using the `dicomdict` function.

Major Bug Fixes

The Image Processing Toolbox 4.2 includes the following bug fixes made since Version 4.1. If you are upgrading from a release earlier than Release 13 with Service Pack 1, then you should also see “Major Bug Fixes” on page 3-4 of the Image Processing Toolbox 4.1 Release Notes.

- The `bwdist` function now produces correct results for very large images (such as 10000-by-5000).
- The `dicomread` function no longer produces the "Error using reshape" message when reading a DICOM file containing multiframe data and overlays stored in both the pixel data and in the metadata.
- The `dicomwrite` function can now correctly handle data attributes with multiple VMs (value multiplicity).
- The `dicomwrite` function now produces correct results for the RLE (run-length encoding) compression type when the number of bits per pixel is greater than 8.
- The `im2col` function, when called with the syntax `im2col(1:N, [1 N])`, now returns a column vector.
- The `imview` function, when called with the syntax `imview(I,[])`, where `I` is a constant image, no longer produces a warning message about a badly conditioned polynomial.
- The `montage` function no longer displays an extra blank row when displaying certain multiples of images.
- The `poly2mask` function no longer errors when trying to close certain polygons, specifically polygons where the last element of vector `X` doesn't match the first element, `x(1) ~= x(end)`, and the last element of vector `Y` matches the first, `y(1) == y(end)`, or vice versa.
- The `stretchlim` function now uses more bins to achieve better results for input images of class `uint16` or `double`.

Upgrading from an Earlier Release

There are no upgrade issues if you are upgrading to Version 4.2 from Version 4.1 or 4.0.

However, if you are upgrading from Version 3.1 or earlier, then see "Upgrading from an Earlier Release" on page 5-5 of the Image Processing Toolbox 3.2 Release Notes.

Image Processing Toolbox 4.1 Release Notes

New Features 2-2
Reading and Writing Data with JPEG Lossless Compression . 2-2
Reading ICC Profiles Embedded in TIFF Files 2-2
Reading and Writing L*a*b* Color Data 2-2

Major Bug Fixes 2-4
applycform Fixes 2-4
Compiling Spatial Transformation Functions 2-4

New Features

This section introduces the new features and enhancements added in the Image Processing Toolbox 4.1 since Version 4.0 (which was released as a Web-download release after Release 13).

Reading and Writing Data with JPEG Lossless Compression

The toolbox now supports reading and writing data that has been compressed using JPEG lossless compression. With lossless compression, you can recover the original image from its compressed form. Lossless compression, however, achieves lower compression ratios than its counterpart, lossy compression.

Using either the `imread` function or the `dicomread` function, you can read data that has been compressed using JPEG lossless compression.

Using either the `imwrite` or the `dicomwrite` function, you can write data to a JPEG file using lossless compression. For the `imwrite` function, you specify the `Mode` parameter with the `'lossless'` value. For the `dicomwrite` function, you specify the `CompressionMode` parameter with the `'JPEG lossless'` value.

Reading ICC Profiles Embedded in TIFF Files

`iccread` can now read ICC profiles that are embedded in a TIFF file, if the TIFF file contains one. ICC profiles contain information that color management systems need to translate color data between devices.

To determine if a TIFF file contains an ICC profile, use the `imfinfo` function to retrieve information about the file. If the returned data contains the `ICCProfileOffset` field, the file contains an embedded ICC profile.

Reading and Writing L*a*b* Color Data

The `imread` function can now read color data that uses the *L*a*b** color space from TIFF files. The TIFF files can contain *L*a*b** values that are in 8-bit or 16-bit CIELAB encodings or in 8-bit or 16-bit ICCLAB encodings.

If a file contains 8-bit or 16-bit CIELAB data, `imread` automatically converts the data into 8-bit or 16-bit ICCLAB encoding. The 8-bit or 16-bit CIELAB data cannot be represented as a MATLAB array because it contains a combination of signed and unsigned values.

The `imwrite` function can write *L*a*b** data to a file using either the 8-bit or 16-bit CIELAB encoding or the 8-bit or 16-bit ICCLAB encoding. You select the encoding by specifying the value of the `ColorSpace` parameter.

Major Bug Fixes

The Image Processing Toolbox, Version 4.1, includes the following bug fixes

applycform Fixes

The `applycform` function includes two bug fixes.

- The `applycform` function did not apply some profiles correctly when the input color was in the *XYZ* color space. Specifically, profiles containing an 8-bit or 16-bit lookup table containing a non-identity "E" matrix were not processed correctly by `applycform`. For details about the E matrix, see ICC Specification ICC.1:2001-04, sections 6.5.7 and 6.5.8.
- The `applycform` function now handles correctly Matrix/TRC profiles that contain a single gamma correction factor. Previously, the forward and inverse conversions were reversed.

Compiling Spatial Transformation Functions

Applications that call the `imresize`, `imrotate`, `imtransform`, `tformarray`, `tformfwd`, and `tforminv` functions can now be compiled using the MATLAB Compiler.

Image Processing Toolbox 4.0 Release Notes

New Features 2-2
New Image Viewer 2-2
Enhanced Color Space Functions 2-3
New Image Enhancement Methods 2-4
Enhanced DICOM Support 2-4
Fan Beam Projection Transforms 2-4
Boundary Tracing Functions 2-5
Unsigned Integer Lookup Tables 2-5
Optimized Image Arithmetic Functions 2-5
Performance Improvements 2-5

Minor Enhancements 2-6
Changes to Sample Images Included 2-7
Obsolete Functions 2-7

Major Bug Fixes 2-8

New Features

This section introduces the new features and enhancements added in the Image Processing Toolbox 3.2 since Version 3.2 (Release 13).

New Image Viewer

The toolbox includes a new tool for displaying images, called the Image Viewer. This tool supports zooming, scrolling, and overview navigation with large images. The Image Viewer automatically displays the pixel value at the mouse location but you can also use a special zoom tool, called the Pixel Region tool, to perform simultaneous color and quantitative inspection of individual pixels. You can also view metadata for the image file or MATLAB variable.

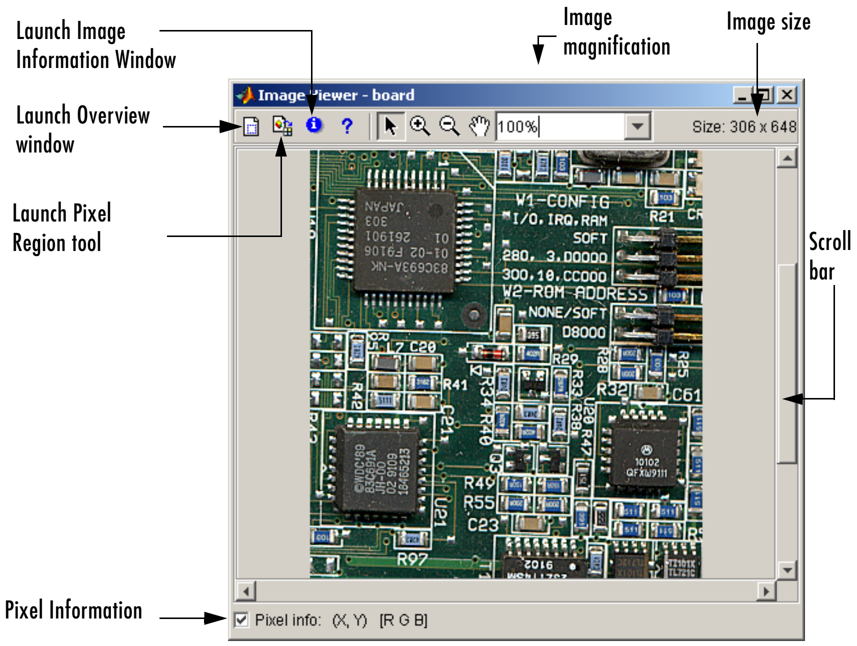
To start the Image Viewer, use the `imview` function.

```
imview('board.tif')
```

The following figure illustrates the Image Viewer and its capabilities.

Note On platforms that don't support JAVA, have an older version of JAVA, and on Macintosh systems, calls to `imview` invoke the `imshow` function. The toolbox issues this warning when `imview` is invoked:

```
 'IMVIEW is not available on this platform.', ...
'Calling IMSHOW instead.');
```



Enhanced Color Space Functions

The toolbox includes a pair of new functions, `makecform` and `applycform`, for converting to and from a family of standard, device-independent color spaces. The functions support conversions between members of the family of color spaces defined by the CIE *Commission Internationale de l'Éclairage* (International Commission on Illumination), including the XYZ, xyY, uvL, $u'v'L$, $L^*a^*b^*$, and L^*ch color spaces. The functions also support conversion to and from the industry standard $sRGB$ color space.

The toolbox also includes new function, `iccread`, for reading in ICC color profiles and using them to transform color data.

In addition, the toolbox also includes functions for converting the class representation of converted color spaces: `lab2uint8`, `lab2uint16`, `lab2double`, `xyz2uint8`, and `xyz2double` functions.

New Image Enhancement Methods

The toolbox includes two new image enhancement functions: `adapthisteq` and `decorrstretch`.

The `adapthisteq` function performs contrast-limited adaptive histogram equalization (CLAHE). This function uses a contrast-enhancement method that works significantly better than regular histogram equalization for most images.

The `decorrstretch` function performs a decorrelation stretch on truecolor images, or images with multiple color or spectral bands. Decorrelation stretch is a technique used to enhance, or stretch, the color differences in an image. This function can be used, for example, to aid visual interpretation when two or more bands are significantly correlated.

Enhanced DICOM Support

The `dicomwrite` function now supports exporting to DICOM files using the MR (magnetic resonance) and CT (computed tomography) modalities.

The `dicominfo` and `dicomread` functions can now read some files that are marginally noncompliant with the DICOM specification. Some commonly-used medical imaging devices produce such files. In addition, these functions can now read some files produced by GE devices that use certain private transfer syntaxes.

The toolbox includes a new function, `dicomuid`, that generates DICOM unique identifiers. This is the same method used by the `dicomwrite` function.

Fan Beam Projection Transforms

The toolbox includes two new functions, `fanbeam` and `ifanbeam`, for computing an alternative mathematical representation of an image using fan beam projections. Using the `ifanbeam` function, you can reconstruct an image from fan beam projection data.

The toolbox also includes functions, `fan2para` and `para2fan`, for converting projection data between fan-beam and parallel-beam geometries. (You use the `radon` function to create parallel beam projection data.)

Boundary Tracing Functions

The toolbox includes a new function, `bwboundaries`, to trace the boundaries of all objects in a binary image. The new `bwtraceboundary` function traces a single boundary from a given starting point.

Unsigned Integer Lookup Tables

The toolbox includes a new function, `uintlut`, that changes element values in a `uint8` or `uint16` array by passing them through a 256-element or 65,536-element lookup table. This low-level utility function has been used to speed up other toolbox functions such as `imadjust`.

Optimized Image Arithmetic Functions

The image arithmetic functions have been optimized in two ways:

- Portable code improvements have been made to speed up the arithmetic functions on all platforms.
- Pentium- and MMX-specific code improvements have been made to provide additional speed improvements on the Windows and Linux platforms. The changes are based on the Intel Performance Primitives Library.

Functions affected by these improvements include the `imabsdiff`, `imadd`, `imcomplement`, `imdivide`, `imlincomb`, and the `immultiply` functions. To determine if the Intel Performance Primitives Library is being used, call the `ippl` function.

Performance Improvements

A variety of existing toolbox functions have been optimized to run faster and use less memory.

- Image type conversion functions
- Certain image enhancement functions: `imadjust` and `imhist`
- Certain color space conversion functions: `rgb2gray`, `rgb2ntsc`, `rgb2ycbcr`, and `ycbcr2rgb`
- Deblurring functions: `deconvblind`, `deconvlucy`, `deconvreg`, and `deconvwnr`

Minor Enhancements

In addition to the major new features, the toolbox includes several additional enhancements.

Function	Enhancement
`cp2tform`	The algorithms have been modified to make them more robust numerically when the input coordinates have a very large offset from the origin.
`cpselect`	The Control Point Selection Tool now displays a single legend window even if multiple instances of the tool are being displayed.
`imadjust`	Supports a simplified, single-input syntax, in which it uses `stretchlim` to compute contrast-stretch parameters automatically.
`immovie`	No longer flickers while a movie is being generated.
`medfilt2`	Class support has been extended to all numeric types and it uses a new algorithm for large window sizes that is significantly faster than the old one.
`ordfilt2`	Class support has been extended to all numeric types and it uses a new algorithm for large rectangular domains that is significantly faster than the old one.
`regionprops`	Uses an improved method for computing the convex hull of a labeled object, resulting in more accurate `ConvexHull`, `Convexity`, and `ConvexImage` measurements.
`roipoly`	Uses a new algorithm that produces more intuitive results and has better performance. Users who have code that requires the same output as the previous version can use the function `roipolyold`.
`tformfwd` and `tforminv`	Supports additional syntaxes that make these functions easier to use for common operations.

Changes to Sample Images Included

The sample images listed below have been removed from the toolbox.

`afmsurf`	`bonemarr`	`enamel`	`ngc4024l`	`rice`	`testpat2`
`alumgrns`	`circles`	`flowers`	`ngc4024m`	`saturn`	`text`
`bacteria`	`circlesm`	`ic`	`ngc4024s`	`shot1`	`tissue1`
`blood1`	`debye1`	`lily`	`pearlite`	`testpat1`	

The following new sample images are included with the toolbox.

`blobs.png`	`peppers.png`	`testpat1.png`
`circles.png`	`rice.png`	`text.png`
`coins.png`	`saturn.png`	`tissue.png`
`glass.png`	`solarspectra.fts`	

New ICC Profiles

The toolbox includes sample ICC profiles that can be used with the color space conversion functions.

File	Description
`lab8.icm`	8-bit $L*a*b*$ profile
`monitor.icm`	Typical monitor profile
`sRGB.icm`	$sRGB$ profile
`swopcmyk.icm`	CMYK input profile

Obsolete Functions

Because of fundamental ambiguities in their behavior and definition, the `isrgb`, `isind`, `isbw`, and `isgray` functions have been obsoleted and will issue a warning when called.

Major Bug Fixes

The Image Processing Toolbox, Version 4.0, includes the following bug fixes.

Function	Fix
`bwlabel`	No longer produces a segmentation violation when called with an empty matrix.
`cpselect`	• The Control Point Selection Tool no longer causes MATLAB to hang when invoked from within a script or GUI with the `input` or `uiwait` functions, which wait for users to complete the selection. • Point prediction in the Control-Point Selection Tool no longer fails with the error message “Attempt to reference field on non-structure array 'pickPair'.”
`dicomread`	Now reads DICOM files that contain 1-bit overlay data.
`imabsdiff`, `imadd`, `imdivide`, `imlincomb`, `imsubtract`	The image arithmetic functions no longer error when called with logical inputs
`imfill`	Now fills hole pixels on the outer edge of an image that are not connected to the background because a non-default connectivity was specified.
`imresize`	The anti-aliasing filtering, applied by `imresize` when shrinking an image, no longer causes a narrow strip of dark pixels to appear around the edge of the image.
`label2rgb`	Now uses a new default colormap. The zero-color in the previous default colormap was the same as the last color in the colormap, with the result that the object labeled with the highest number could not be distinguished from the background.

Function	Fix
`radon`	An off-by-one error in the calculation of the center pixel location for images with even dimensions has been fixed.
`rgb2ind`	No longer errors when passed an input image that contain only a single color.
`strel`	The syntax `strel('ball', r, h, 0)` returns an empty `strel` object instead of a degenerate ball structuring element with radius 0.

If you are upgrading from a release earlier than Release 13, then you should also see “Major Bug Fixes” on page 3-4 of the Image Processing Toolbox 3.2 Release Notes.

Image Processing Toolbox 3.2 Release Notes

New Features 3-2

Major Bug Fixes 3-3

Upgrading from an Earlier Release 3-5

New Features

This section introduces the new features and enhancements added in the Image Processing Toolbox 3.2 since Version 3.1 (Release 12.1).

The new features introduced in the Image Processing Toolbox, Version 3.2, include:

- More error checking of input images, specifically input classes, attributes and option string processing, with clearer error messages
- Support for writing DICOM files
- Changes to how binary images are represented. (Support for the new MATLAB `logical` data type.)
- Enhancements to several existing functions

If you are upgrading from a release earlier than Release 12.1, then you should also see “New Features” on page 6-2.

Writing DICOM Files

The Image Processing Toolbox now supports writing files in Digital Imaging and Communications in Medicine (DICOM) format, using the `dicomwrite` function. Previous releases of the toolbox supported reading DICOM files with the `dicomread` function and reading metadata from a DICOM file using the `dicominfo` function.

Representing Binary Images

In previous releases, toolbox functions that returned binary images returned them as `uint8` logical arrays. The toolbox used the presense of the logical flag to signify that the data range in the file was [0,1].

With this release, the toolbox returns binary images as logical arrays, using the new MATLAB `logical` data type. For more information about the new `logical` class, see the MATLAB 6.5 Release Notes.

Changes to Existing Functions

The Image Processing Toolbox, Version 3.2, includes changes to these existing functions.

Function	Description of Change
`circshift`	Moved into MATLAB
`freqz2`	Checks for insignificant real part in addition to insignificant imaginary part
`getnhood`	Returns a logical array
`gray2ind`	More efficient memory usage
`imfill`	New syntax for grayscale images does not require `'holes'` argument. This option is selected automatically.
`imlincomb`	Accepts more than two images as input and you can specify the output class
`immovie`	Flicker during movie creation eliminated
`imtransform`	Linear and bicubic interpolation are faster
`ordfilt2`	Uses a different algorithm for binary images that improves processing speed for these images
`roifilt2`	More efficient. Operation is performed only on the region of interest, not the entire image.

Major Bug Fixes

The Image Processing Toolbox 3.2 includes several bug fixes made since Version 3.1. You can see a list of the particularly important Version 3.2 bug fixes.

If you are viewing these Release Notes in PDF form, please refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site and use the link provided.

If you are upgrading from a release earlier than Release 12.1, then you should also see “Major Bug Fixes” on page 6-11.

Upgrading from an Earlier Release

This section describes several upgrade issues involved in moving from the Image Processing Toolbox Version 3.1 to Version 3.2.

Change to Data Type of Output Binary Images

All the Image Processing Toolbox functions that return a binary image now return a binary image of class `logical`. In previous releases, these functions returned binary images of a numeric class with the logical flag set. The Image Processing Toolbox used the existence of the logical flag to identify a binary image.

If your application checks the data type of the binary images returned by Image Processing Toolbox functions, you will need to change your code.

Note The `logical` class is not one of the numeric classes in MATLAB.

Change to Interpretation of Input Images

Image Processing Toolbox functions that accept different types of images, such as grayscale and binary, no longer attempt to determine if an input image of a numeric class is intended to be a binary image.

In previous releases, toolbox functions that acccepted different types of images checked the contents of an image to determine how to interpret it. For example, if an image was of class `double` and contained only 0s and 1s, the toolbox function would interpret it as a binary image. With Version 3.2, the toolbox only interprets images of class `logical` as binary images.

In the Image Processing Toolbox, the names of functions that accept both grayscale and binary images typically start with the characters “`im`”, such as `imdilate`.

Converting Binary Images to an Integer Data Type

With this release, if you convert a binary image to a numeric type, the image ceases to be a binary image.

In previous releases, the Image Processing Toolbox conversion functions `im2uint8` and `im2double` preserved the binary attribute of the converted image. For example, if you converted a binary image of class `double`, which had the logical flag set, the output image returned by the `im2uint8` function would also be a logical image of class `uint8`, with the logical flag set.

For example, create a simple logical array

```
bw = logical([1 0; 0 1])
bw =

     1     0
     0     1
whos
Name           Size                    Bytes  Class

bw             2x2                         4  logical array
```

When you convert this array to a `uint8` data type, notice that it is no longer of class `logical`.

```
new_image = im2uint8(bw)

new_image =

   255     0
     0   255

whos
Name           Size                    Bytes  Class

bw               2x2                       4  logical array
new_image        2x2                       4  uint8 array
```

Image Processing Toolbox 3.1 Release Notes

New Features 4-2
Morphology 4-3
Spatial Transformations 4-4
Image Registration 4-5
Integer Image Arithmetic 4-6
Integer Image Filtering 4-7
Deconvolution/Deblurring 4-7
Support for DICOM Files 4-8
Miscellaneous New Functions 4-8
New Demos 4-9

Major Bug Fixes 4-11
Version 3.1 Bug Fixes 4-11

New Features

This section describes the new features and enhancements of the Image Processing Toolbox, Version 3.0, and the Version 3.1 update, both introduced since the Image Processing Toolbox 2.2.2 (Release 12.0).

Note The Image Processing Toolbox 3.0 was made available in Web-downloadable form after Release 12.0. The Image Processing Toolbox 3.1 was part of Release 12.1.

The new features introduced in the Image Processing Toolbox, Version 3.0, include:

- Many new morphology functions
- New spatial transformation functions
- New image registration functions, with a new graphical user interface
- New integer image arithmetic functions
- New integer image filtering function
- New image deblurring (deconvolution) functions
- Support for DICOM files
- Miscellaneous new functions
- New image processing demos

The new features introduced in the Image Processing Toolbox, Version 3.1, include:

- New deblurring function, `deconvblind`, that implements the blind deconvolution algorithm
- New utility function, `label2rgb`, that converts a label matrix into an RGB color image

For information about Image Processing Toolbox features that are incorporated from Version 2.2.2, see "New Features" on page 7-2.

Morphology

Version 3.0 adds a broad suite of new mathematical morphology tools open up broad new classes of applications in segmentation and image enhancement.

The existing dilation and erosion operators have been extended to work with grayscale images. New functions range from additional basic operators (opening, closing, tophat) to advanced tools useful for segmentation (distance transforms, reconstruction-based operators, and the watershed transform). The functions use advanced techniques for high performance, including automatic-structuring element decomposition, 32-bit binary image packing, and queue-based algorithms.

Function	Description
`bwareaopen`	Binary area open (remove small objects)
`bwdist`	Distance transform
`bwhitmiss`	Binary hit-miss operation
`bwlabeln`	Label-connected components in N-D binary image
`bwpack`	Pack binary image
`bwulterode`	Ultimate erosion
`bwunpack`	Unpack binary image
`conndef`	Default connectivity array
`imbothat`	Perform bottom-hat filtering
`imclearborder`	Suppress light structures connected to image border
`imclose`	Close image
`imdilate`	Dilate image
`imerode`	Erode image
`imextendedmax`	Extended-maxima transform
`imextendedmin`	Extended-minima transform

Function	Description
`imfill`	Fill image regions and holes
`imhmax`	H-maxima transform
`imhmin`	H-minima transform
`imimposemin`	Impose minima
`imopen`	Open image
`imreconstruct`	Morphological reconstruction
`imregionalmax`	Regional maxima
`imregionalmin`	Regional minima
`imtophat`	Tophat filtering
`strel`	Create morphological structuring element
`strel/getheight`	Get structuring element height
`strel/getnhood`	Get structuring element neighborhood
`strel/getsequence`	Get sequence of decomposed structuring elements
`strel/isflat`	Return true for flat structuring element
`strel/reflect`	Reflect structuring element about its center
`strel/translate`	Translate structuring element
`watershed`	Find image watershed regions

Spatial Transformations

Version 3.0 adds functions for applying a variety of spatial transformations to images and to points. This is a core computational capability. Supported transform types include affine, projective, and user-defined custom transformations. Multidimensional transformations are supported, where you can control which dimensions are the transform dimensions. For example, you can apply a two-dimensional transform to an RGB image, and each color plane is automatically transformed the same way. You can even control the type of interpolation independently along each dimension, and specify interpolants that you define.

Function	Description
`checkerboard`	Create checkerboard image
`findbounds`	Find output bounds for geometric transformation
`fliptform`	Flip the input and output roles of a `TFORM` struct
`imtransform`	Apply geometric transformation to image
`makeresampler`	Create resampler structure
`maketform`	Create geometric transformation structure (`TFORM`)
`tformarray`	Geometric transformation of a multidimensional array
`tformfwd`	Apply inverse geometric transformation
`tforminv`	Apply forward geometric transformation

Image Registration

Version 3.0 adds several functions useful for registering (aligning) two images. This is critical in remote sensing and medical imaging, for example. There are functions for inferring various spatial transformations from control-point pairs, for the subpixel adjustment of control-point pair locations, and for normalized cross-correlation. There is also a graphical user interface (GUI) for selecting control-point pairs in a pair of images.

Function	Description
`cp2tform`	Infer spatial transformation from control-point pairs
`cpcorr`	Tune control-point locations using cross-correlation
`cpselect`	Control-point selection tool (graphical user interface)

Function	Description
`cpstruct2pairs`	Convert `CPSTRUCT` to valid pairs of control points
`normxcorr2`	Normalized two-dimensional cross-correlation

Integer Image Arithmetic

The Image Processing Toolbox 3.1 includes new functions for performing arithmetic on image arrays without converting them to double-precision. In addition to the basic operations (add, subtract, multiply, and divide), there are several key functions (absolute difference, linear combination, and complementation) that cannot readily be implemented in terms of the basic operations.

Function	Description
`imabsdiff`	Absolute difference of two images
`imadd`	Add two images, or add constant to image
`imcomplement`	Complement image
`imdivide`	Divide two images, or divide image by constant
`imlincomb`	Linear combination of images
`immultiply`	Multiply two images, or multiply image by constant
`imsubtract`	Subtract two images, or subtract constant from image

Integer Image Filtering

Version 3.0 added a function for performing filtering on image arrays without converting them to double precision, a significant memory savings in a common operation. You can specify several different boundary padding options. You can also perform higher dimensional filtering.

Function	Description
`imfilter`	Filter 2-D and N-D images

Deconvolution/Deblurring

Version 3.0 added support for several fundamental algorithms for the deconvolution (deblurring) of images. All of the functions support multidimensional problems.

Function	Description
`deconvblind`	Deblur image using blind deconvolution algorithm [New with Version 3.1]
`deconvlucy`	Deblur image using Lucy-Richardson algorithm
`deconvreg`	Regularized deconvolution
`deconvwnr`	Wiener deconvolution
`edgetaper`	Taper image edges according to PSF
`fspecial`	Existing function; added `'disk'` and `'motion'` options
`otf2psf`	Convert optical transfer function to point-spread function
`psf2otf`	Convert point-spread function to optical transfer function

Support for DICOM Files

Version 3.0 adds functions for reading image data and metadata from DICOM files. DICOM is an important file and network interchange standard in the area of medical imaging.

Function	Description
`dicomread`	Read image data from DICOM file
`dicominfo`	Read metadata from DICOM file

Miscellaneous New Functions

Version 3.1 included several new utility functions or previously undocumented utility functions. Most of these were created to support functions in the key feature categories, such as deconvolution.

Function	Description
`circshift`	Shift array circularly Note: This function was moved into MATLAB in release 3.2 of the Image Processing Toolbox.
`graythresh`	Compute global image threshold using Otsu's method (image enhancement)
`im2mis`	Convert image to Java MemoryImageSource Note: This function was renamed to `im2java` and moved into MATLAB in release 3.2 of the Image Processing Toolbox.
`imnoise`	Added support for new noise types: `'poisson'` and `'localvar'`

Function	**Description**
`label2rgb`	Convert label matrix to RGB image [New for Version 3.1]
`padarray`	Pad array
`regionprops`	Renamed from existing function `imfeature`; extended to N-D
`stretchlim`	Find limits to contrast stretch an image

New Demos

The Image Processing Toolbox 3.1 includes the 15 new extended example demos, presented in HTML form.

Demo Name	**Brief Description**
`ipexconformal`	Explore a Conformal Mapping: illustrates how to use spatial- and image-transformation functions to perform a conformal mapping.
`ipexdeconvblind`	Deblurring Images Using the Lucy-Richardson algorithm: illustrates use of the `deconvlucy` function. [New with Version 3.1]
`ipexdeconvlucy`	Deblurring Images Using the Lucy-Richardson algorithm: illustrates use of the `deconvlucy` function.
`ipexdeconvreg`	Deblurring Images Using a Regularized Filter: illustrates use of the `deconvreg` function.
`ipexdeconvwnr`	Deblurring Images Using the Wiener Filter: illustrates use of the `deconvwnr` function.
`ipexgranulometry`	Finding the Granulometry of Stars in an Image: illustrates how to use morphology functions to perform granulometry.

`ipexmri`	Extracting Slices from a 3-Dimensional MRI Data Set: illustrates how to use the image transformation functions to interpolate and reslice a three-dimensional MRI data set, providing a convenient way to view a volume of data.
`ipexnormxcorr2`	Registering an Image Using Normalized Cross-correlation: illustrates how to use translation to align two images.
`ipexregaerial`	Registering an Aerial Photo to an Orthophoto: illustrates how to use the Control Point Selection Tool to align two images.
`ipexrotate`	Finding the Rotation and Scale of a Distorted Image: illustrates how to use the `cp2tform` function to get the rotation angle and scale factor of a distorted image.
`ipexsegcell`	Detecting a Cell Using Image Segmentation: illustrates how to use dilation and erosion to perform edge detection.
`ipexsegmicro`	Detecting Microstructures Using Image Segmentation: illustrates how to use morphological opening and closing to extract large objects from an image.
`ipexsegwatershed`	Detecting Touching Objects Using Watershed Segmentation: illustrates use of morphology functions to perform marker-control watershed segmentation.
`ipexshear`	Padding and Shearing an Image Simultaneously: illustrates how to use the padding options of the image transformation functions.
`ipextform`	Creating a Gallery of Transformed Images: illustrates how to use the `imtransform` function to perform many types of image transformations.

Major Bug Fixes

This section describes major bug fixes included in the Image Processing Toolbox, Version 3.0 and Version 3.1.

For information about additional bug fixes that are incorporated from Version 2.2.2, see "Major Bug Fixes" on page 7-3.

Version 3.1 Bug Fixes

- `fspecial` — Fixed incorrect normalization for the Gaussian filter option.
- `improfile` — Fixed an occasional indexing problem caused by round-off error.
- `rgb2ind` — Fixed a problem that caused `rgb2ind` to produce bad results for very large images.
- Functions that operate on binary input images now treat `NaN`s in a consistent manner. When an input array that is expected to be a binary image contains `NaN` values, the `NaN` value is always treated as 1.

Image Processing Toolbox 2.2.2 Release Notes

New Features 5-2
New Demo 5-2
Support For Function Handles 5-2
Documentation Enhanced 5-2

Major Bug Fixes 5-3
imshow Fixes 5-3
bwlabel Segmentation Violation Eliminated 5-3
dilate And erode Return Correct Answers 5-3
freqz2 Fixes 5-3
fspecial Function's 'LoG' Option 5-3
Improved Display for imcrop, improfile, and roipoly 5-4

New Features

The focus of the Image Processing Toolbox 2.2.2 is on bug fixes (see “Major Bug Fixes” below).

A number of important new features will be released in the Image Processing Toolbox 3.0, which will be made available in a Web downloadable version after Release 12.0 is released.

New Demo

The Image Processing Toolbox 2.2.2 includes the new `landsatdemo` function, which is a demo that illustrates how to construct color composite images from multispectral Landsat data.

Support For Function Handles

The following functions have been updated to support function handles, a new MATLAB 6.0 language feature:

- `blkproc`
- `colfilt`
- `nlfilter`
- `qtdecomp`
- `roifilt2`

The MATLAB language has a new data type called the function handle. The function handle captures all the information about a function that MATLAB needs to evaluate it. You can pass a function handle in an argument list to other functions.

Documentation Enhanced

The online *Image Processing Toolbox User’s Guide* was enhanced for Release 12 by adding a “Getting Started” section, and by adding glossaries of relevant terms at the beginning of several chapters.

Major Bug Fixes

The Image Processing Toolbox 2.2.2 includes several important bug fixes that were made in the Image Processing Toolbox 2.2.1 (Release 11.1). This section describes the bugs and how they have been fixed.

imshow Fixes

You can now display the same image twice using `imshow`, without the previous problem of having the images appear to move slightly the second time.

Also, you can now use the syntax `imshow(I,[])` when all the elements of `I` are the same. Now `imshow` displays I using an intermediate shade of gray. Previously, `imshow` would generate an error for this case. (This fix was introduced in the Image Processing Toolbox 2.2.1 (Release 11.1).)

bwlabel Segmentation Violation Eliminated

You can now pass a matrix to `bwlabel` that contains values other than 0 or 1. `bwlabel` treats any nonzero element as an object element. Previously, `bwlabel` would cause a segmentation violation for this case. (This fix was introduced in the Image Processing Toolbox 2.2.1 (Release 11.1).)

dilate And erode Return Correct Answers

The `dilate` and `erode` functions now return the correct answer in all cases. In prior versions of the Image Processing Toolbox, in some cases these functions returned the incorrect answer if you specified the frequency-domain option with a structuring element that contained more than 255 elements.

freqz2 Fixes

The `freqz2` function now returns correct values for the frequency scaling. Also, freqz2 no longer uses an excessive amount of memory.

fspecial Function's 'LoG' Option

The `Log` option of the `fspecial` function now returns correctly scaled values.

Improved Display for imcrop, improfile, and roipoly

The animated lines that the `imcrop`, `improfile`, and `roipoly` functions display on top of images are now displayed clearly.