

Image Processing Toolbox™ Release Notes



MATLAB®

How to Contact MathWorks



Latest news: www.mathworks.com
Sales and services: www.mathworks.com/sales_and_services
User community: www.mathworks.com/matlabcentral
Technical support: www.mathworks.com/support/contact_us



Phone: 508-647-7000



The MathWorks, Inc.
1 Apple Hill Drive
Natick, MA 01760-2098

Image Processing Toolbox™ Release Notes

© COPYRIGHT 2000–2019 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Big Images: Process images that are too large to fit in memory	1-2
Deep Learning Data Preprocessing: Perform additional image augmentations	1-2
Random Patch Extraction Datastore: Extract patches from 3-D data and transformed datastores, and train in parallel	1-2
Deep Learning: Added example using deep neural networks	1-3
inpaintExemplar Function: Fill damaged regions in images with exemplar-based inpainting	1-3
DICOM Volume: Construct isotropic volume from DICOM images	1-3
ROI Tools: Create crosshair shape and other enhancements	1-3
View 3-D Volumes as Slice Planes	1-4
imlocalbrighten Function: Brighten dark areas of images ...	1-6
reducepoly Function: Reduce density of points in ROIs	1-7
Volume Viewer App: Create new session, export visualization settings, and other enhancements	1-8
imshow Function: Specify interpolation method	1-9

volshow Function: Control lighting in volume rendering	1-9
affineOutputView Function: Control view of warped images	1-9
Image Cropping: Crop 3-D volumes and crop using spatial referencing	1-9
Support for Categorical Data	1-9
C-code generation: Generate code from one additional function using MATLAB Coder	1-10

R2019a

ROI Creation Functions: New cuboid shape added	2-2
ROI Creation Functions: New bringToFront Function	2-3
Enhanced Volume Display: View labeled volumes and specify colormap	2-3
Deep Learning: Added example using deep neural networks	2-5
Measurements of Region Properties: Measure circularity and Feret properties	2-6
NIfTI File Format Enhancements: Read and write neuroscience image volumes in the NIfTI-2 file format	2-6
Camera Response: Estimate real-world illumination as a function of pixel intensity	2-6
inpaintCoherent Function: Fill damaged regions in images with coherence transport based inpainting	2-7
burstinterpolant Function: Generate a high-resolution image from a burst of lower resolution images	2-7

rgb2lightness Function: Convert an RGB image to a lightness image	2-7
Performance improvements: Performance enhancements for image warping	2-7
C-code generation: Generate code from four additional functions using MATLAB Coder	2-7
Functionality Being Removed or Changed	2-8
The imshow function now displays large images passively	2-8
The imfindcircles function uses new filter size for logical images	2-8

R2018b

Random Patch Extraction Datastore: Extract random image patches to split up large images for deep learning workflows	3-2
Deep Learning: Added example using deep neural networks	3-2
New Set of ROI Creation Functions	3-2
Volume Show: Visualize 3-D image volumes using the volshow command	3-4
Image Segmentation: Segment 2-D images and N-D volumes using k-means clustering	3-4
Geometric Transformation Objects: Represent and apply custom 2-D and 3-D geometric transformations	3-5
fspecial3: Create predefined 3-D filters	3-5
imflatfield: Perform flat-field correction	3-5
imnlmfilt: Perform non-local means filtering	3-5

imsplit: Split an N-channel image into individual channels . . .	3-5
piqe: Measure image quality using perception-based image quality evaluator (PIQE)	3-5
tonemapfarbman: Reduce dynamic range of HDR images	3-6
fibermetric: Added 3-D support	3-6
Performance improvements: Performance enhancements for 2-D and 3-D morphology, image warping, and fibermetric . . .	3-6
C-code generation: Generate code from three additional functions using MATLAB Coder	3-6
Functionality Being Removed or Changed	3-7
fibermetric calculates new default structural sensitivity	3-7
Old ROI classes are not recommended	3-7

R2018a

Deep Learning: Added examples using deep neural networks	4-2
Deep Learning Data Preprocessing: Efficiently read and add noise to images for training and prediction	4-2
Image Segmenter: Segment images interactively using new techniques including local graph cut	4-2
Edge-Aware Filtering: Smooth images and reduce noise while preserving edge sharpness with bilateral filtering and anisotropic diffusion filtering	4-3
blendexposure: Perform exposure fusion	4-3
imregmtb: Register images using median threshold bitmaps	4-3

montage: Display multiple images from ImageDatastore object, and specify size of the image thumbnails displayed	4-3
Image Morphology: Perform morphological operations on 3-D volumes, and perform skeletonization on all objects in 2-D image or 3-D binary volume	4-3
Performance: Improved performance in functions including 3-D imwarp, 3-D imfilter, entropyfilt, ordfilt2, medfilt2, and bwmorph	4-4
C-code generation: Generate code from one additional function using MATLAB Coder	4-4
Functionality Being Removed or Changed	4-5
montage function has several behavior changes	4-5
denoisingImageSource object is removed	4-5
denoisingImageSource function will be removed	4-6

R2017b

Deep Learning: Denoise images using deep learning techniques	5-2
3-D Image Processing: Process 3-D volumetric image data with support for several additional functions	5-2
Image Enhancement: Adjust colors with automatic white balancing, and reduce haze in images	5-2
Image Quality Metrics: Measure image quality without a reference image, and model image quality using an eSFR test chart	5-3
NIfTI File Format: Read and write neuroscience image volumes in the NIfTI file format	5-4
Image Segmenter: Segment images using new techniques including texture segmentation	5-4

Image Segmentation: Calculate similarity coefficients	5-5
DICOM Browsing: Explore the contents of DICOM media in a browser and programmatically	5-6
Image Warper: Transform group of images quickly using the images.geotrans.Warper object	5-7

R2017a

Image Registration App: Explore various registration techniques interactively to align images	6-2
Volume Viewer App: View and slice 3-D volumetric data	6-2
3-D Volumetric Data: Process 3-D volumetric image data with support for over a dozen functions	6-3
fibermetric: Enhance tubular or elongated structures in images	6-4
Image Segmenter App: Added support for RGB images and graph cut segmentation	6-4
lazysnapping: Segmentation technique	6-4
N-D histograms: Enhance contrast and adjust histograms of N-D images	6-4
dicomread Supports JPEG Variant	6-4
imfilter can return different results for codegen with certain inputs	6-5
Functions Being Removed or Changed	6-5

Image Browser App: View multiple images and import selected image into apps	7-2
Color Thresholder App: View color data as point cloud for segmentation	7-4
Edge-Aware Filtering: Perform edge-aware filtering based on Laplacian pyramids	7-5
3-D Superpixels: Use simple linear iterative clustering (SLIC) with volumetric images	7-6
3-D Median Filtering: Apply median filter to volumetric image data	7-6
colorcloud: Display color information as a point cloud	7-6
edge: Perform faster Canny edge detection	7-7
imshow now sets colormap of axes	7-7
nitfread Supports JPEG2000 Compression	7-8
imregdemons Supports 3-D Images on GPU	7-9
Contrast Adjustment Tool Now Supports Rsets	7-9
Specify Initial Folder Opened in Open Image Dialog Box	7-9
Functions Being Removed	7-9

superpixels: Use simple linear iterative clustering (SLIC) to group pixels for efficient segmentation of color and grayscale images	8-2
Image Segmenter: Segment images using new techniques, including flood-fill and adaptive thresholding	8-2
Image Batch Processor: Export non-image results using expanded workflows	8-2
imgradient3 and imgradientxyz: Calculate 3-D gradient magnitude, direction, and elevation	8-2
C-code generation: Generate code from 20 additional functions using MATLAB Coder	8-2
imbinarize, otsuthresh, and adaptthresh: Threshold images using global and locally adaptive thresholds	8-4
Structuring Elements: New shapes and a new function	8-4
DICOM function support non-ASCII character sets	8-4
edge: Change to Canny Edge Detection	8-4
Performance improvements	8-4
Color space conversion functions: improved precision and numerical consistency	8-5
Functions Being Removed	8-5

R2015b

C-code generation support for more than 20 functions using MATLAB Coder	9-2
gabor and imgaborfilt: New class and function for designing and applying Gabor filter banks for use in edge and texture analysis	9-2
grayconnected and imboxfilt: Segment regions by intensities and apply spatial domain filters	9-3
Performance: Performance improvements for grayscale morphology and image filtering	9-3
dpxread and dpxinfo: Read Digital Picture Exchange files . . .	9-3
imwarp function supports new SmoothEdges parameter	9-3
DICOM functions support new options	9-3
New example shows use of imaging functions with the Vision HDL Toolbox	9-4

R2015a

C-code generation support for more than 20 functions, including regionprops, watershed, bweuler, bwlabel, bwperim, and multithresh using MATLAB Coder	10-2
App for batch processing sets of images	10-2
Fast geodesic interactive segmentation	10-4
Optimized function for Gaussian filtering	10-4
Fill entire region including border pixels	10-4

Visualize results of boundary tracing	10-4
Examine contents of DICOM files	10-4
Live image capture in Color Thresholder app	10-5
regionprops function can return results in table	10-5
GPU acceleration for imregionalmax, imregionalmin, imgaussfilt, imgaussfilt3, and regionprops functions	10-5
Functions Being Removed	10-6

R2014b

Apps for image segmentation and region analysis	11-2
Image Segmenter app	11-2
Image Region Analyzer app	11-4
C-code generation support for 16 additional functions using MATLAB Coder, including bwtraceboundary, imadjust, imclearborder, and medfilt2	11-5
Nonrigid image registration	11-5
Image warping using displacement fields	11-6
Image segmentation using the Fast Marching Method algorithm	11-6
Image comparison using mean-squared error	11-6
Image filtering based on object properties	11-6
Color space conversion functions	11-6
activecontour function supports parameter to control tendency of contour to expand or contract	11-7

Region-of-Interest (ROI) functions now support deletion from context menu	11-7
dicomwrite function now supports the ability to specify the bitdepth of images written	11-7
GPU acceleration for bwlabel and imregdemons	11-7

R2014a

C-code generation for more than 25 functions, including edge, imfilter, imwarp, imopen, imclose, imerode, and imdilate using MATLAB Coder	12-2
GPU acceleration for an additional nine functions, including bwdist, imfill, imreconstruct, iradon, radon, and stretchlim	12-2
App for color image thresholding	12-2
Image quality metrics, including peak signal to noise (psnr) and structured similarity metric (ssim)	12-3
Guided filtering for image enhancement	12-3
Phase correlation and translation-only image registration functions	12-4
File names of Image Processing Toolbox examples changed	12-4
Location of sample images changed	12-5
regionprops function uses new algorithm to calculate perimeter	12-6

GPU acceleration for more than 20 functions, including bwmorph, edge, imresize, and medfilt2	13-2
Additional 2D geometric transformations: piecewise linear, local weighted mean, and polynomial	13-2
Additional parameter in imregister and imregtform to specify initial transformation	13-2
fitgeotrans function for fitting geometric transformation to control point pairs	13-3
imregister and imregtform Return Different Values	13-3
Functions Being Removed	13-3

Image segmentation using active contours	14-2
Classes and functions for representing and applying 2-D and 3- D geometric transformations	14-2
Classes for defining world coordinate system of an image ..	14-2
Code generation for conndef, imcomplement, imfill, imhmax, imhmin, imreconstruct, imregionalmax, imregionalmin, iptcheckconn, and padarray functions (using MATLAB Coder)	14-2
GPU acceleration for imrotate, imfilter, imdilate, imerode, imopen, imclose, imtophat, imbothat, imshow, padarray, and bwlookup functions (using Parallel Computing Toolbox)	14-3

Unsharp mask filtering	14-3
-------------------------------------	-------------

R2012b

Image gradient computation with <code>imgradient</code> and <code>imgradientxy</code> functions	15-2
Histogram matching with <code>imhistmatch</code> function	15-2
Multilevel thresholding with <code>multithresh</code> and <code>imquantize</code> functions	15-2
3-D image registration with <code>imregister</code> function	15-2
Code generation for <code>bwmorph</code> and <code>bwlookup</code> with MATLAB Coder	15-2
Added function <code>bwlookup</code>	15-3
Writing private metadata when anonymizing DICOM files ..	15-3
Expanded color options with <code>imshowpair</code>	15-3
Performance improvements	15-3
New Example	15-3

R2012a

Intensity-Based Image Registration	16-2
Two New Functions to Visually Compare Images	16-2
Circle Detection Using the Circular Hough Transform	16-2

Performance Improvements	16-2
New and Updated Demos	16-2
Data Type Change to Output Variable for bwdist	16-3
iradon Function Updated	16-3
Functions Being Removed	16-3

R2011b

Parallel Block Processing Now Possible with blockproc	17-2
New bwdistgeodesic Function Computes Geodesic Distance Transform	17-2
New graydist Function Computes Gray-Weighted Distance Transform	17-2
New imapppmatrix Function Computes Linear Combination of Color Channels	17-2
Performance Improvements	17-2
Faster Functions	17-2
hdrread Now Corrects for Small Values	17-2
Change in Behavior for dicomwrite	17-3
Warning and Error ID Changes	17-3
Functions and Function Elements Being Removed	17-4

R2011a

New bwconvhull Function Computes Convex Hull Image . . .	18-2
New dicomwrite Option Writes Multiframe Imagery to Single File	18-2
nitfread Now Reads NITF Files with JPEG-Compressed Images	18-2
Reduced Memory Use for std2	18-2
Reduced Memory Use for watershed	18-2
iccread and iccwrite Now Warn in Cases of Unrecognized PrimaryPlatform Signatures	18-3
Plot Selector Now Includes implay	18-3
Support for Code Generation from MATLAB	18-3
edge Function No Longer Smooths Image Twice	18-3
Functions and Function Elements Being Removed	18-4

R2010b

New corner Function Detects Corners in Image	19-2
Efficient Display and Navigation of Very Large Images of Arbitrary Format in imtool	19-2
Now Possible to Control Padding Behavior when Using the blockproc Function	19-2
Writing to JPEG2000 File Format Supported by blockproc	19-2

Enhancements to the dicomread Function	19-3
The ImageMagnification Field of the nitfinfo Function Now Returns a Numeric Value	19-3
Performance Improvements	19-3
Faster Functions	19-3
Functions and Function Elements Being Removed	19-3

R2010a

New ImageAdapter Class Supports Custom File Formats for blockproc	20-2
The blockproc Function Now Supports Spatially Varying Operations	20-2
Plot Selector Now Generates Plots for imshow and imtool	20-2
makecform Now Supports White Point Adaptation	20-2
IntelIntegrated Performance Primitives Library Support Extended to imdilate, imerode, and medfilt2	20-3
imreconstruct Now Supports int64 and uint64	20-3
Performance Improvements	20-3
Faster Functions	20-3
Multithreaded Functions	20-3
Non-interactive Syntax of improfile Returns Different Output	20-4

New blockproc Function to Process Large Images	21-2
Intel Integrated Performance Primitives Library Upgraded and Support Extended to maci64	21-3
Expanded hough Function Allows Specification of Arbitrary Theta Search Space	21-3
The imfilter Function Now Faster for uint16 and double Inputs	21-4
Improved Speed for Calculating N-D Euclidean Distance Transforms with the bwdist Function	21-4
Modified Behavior for the regionprops ConvexHull Property	21-5
Efficient Display and Navigation of Very Large NITF-File Images in imtool	21-5
Performance Improvements	21-5
Faster Functions	21-5
Multithreaded Functions	21-6

Faster, Less Memory-Intensive Workflow for Labeling Regions and Measuring Their Properties	22-2
Multithreaded Implementation of imfilter Function	22-2
Efficient Display and Navigation of Very Large Images in imtool	22-2
New Dialog Box for Setting Toolbox Preferences	22-2

New imcolormaptool Function That Opens Choose Colormap Tool	22-3
End Point and Branch Point Detection Now Possible	22-3
nitfread Now Allows Image Subregion Selection	22-3
Support for Intel IPP on Mac	22-3
getColor, getLabelVisible, and setLabelVisible Methods Added to imdistline	22-4
Five Functions Moved to MATLAB	22-4
Fan-Beam Functions Updated	22-4

R2008b

Performance Improvements	23-2
New cornermetric Function Detects Corners	23-2
Now Support Absolute Colorimetric Rendering Intent for GrayTRC and MatTRC	23-2
New createMask Method Creates Mask for Any ROI	23-2
Interactive Tools Refresh when Target Image Changes	23-2
The imscrollpanel 'PreserveView' Parameter Now Works for Images of All Sizes	23-3
Distance Tool and Cropping Tool Now Modes in imtool	23-3
In imtool Opening Adjust Contrast Tool No Longer Selects Window/Level Tool	23-3
immovie Command No Longer Shows Preview	23-4

Replace Calls to ipttable Function with MATLAB uitable Function	23-4
imcontour Second Output Argument Changed	23-5
impixelinfo Tool Disappears when Image Changes	23-5
Some Code Moved into Different Directories	23-6
Functions and Demos Being Removed	23-6

R2008a

Create High Dynamic Range (HDR) Images and Write Them to Files	24-2
Measure Properties of Regions in Grayscale Images	24-2
Display Very Large Images by Subsampling	24-2
Enhancements to ROI Tools	24-2
ROI Tools Reimplemented as MATLAB Classes	24-2
ROI Tools Support New wait and resume Methods	24-3
Interactively Add New Vertices to ROI Polygons	24-3
Enhancements to Color Functions	24-3
makecform Supports Converting Between sRGB and CMYK	24-3
iccwrite Creates Smaller ICC Profiles	24-4
cp2tform Function Supports New Transformations	24-4
hough Function Uses Specified RhoResolution Values	24-4
Enhancements to Interactive Tools	24-4
New and Updated Demos	24-5
Enhancements to Other Functions	24-5

R2007b

New Interactive Image Sequence and Video Viewer	25-2
Image Tool Includes Cropping, Enhanced Contrast Adjustment, and Saving of Modified Images	25-2
New Function for Converting Bayer Pattern Encoded Images to RGB	25-2
New Function for Creating a Multiresolution Gaussian Pyramid	25-3
Enhanced ROI Definition Behavior for imcrop, roifill, and roipoly	25-3
New Modular Interactive GUI-building Tools	25-3
New Programmable ROI Tools	25-3
Support for Reading NITF and HDR Images	25-4
Enhanced Performance	25-4
DICOM Dictionary Upgrade	25-4
Changes to Other Functions	25-5

R2007a

Enhancements to imresize Function	26-2
applycform Supports Tetrahedral Interpolation	26-2
Control Point Selection Tool Enhancements	26-2
Enhancements to impoint, imline, and imrect Functions ...	26-3

Enhancements to montage Function	26-3
makecform Uses 'icc' Whitepoint for L*a*b*/sRGB Conversions	26-3
normxcorr2 Might Return Different Results	26-3
watershed Uses New Algorithm	26-3
Changes to Other Functions	26-4

R2006b

Enhancements to DICOM Capabilities	27-2
New Symmetric Option with graycomatrix Function	27-2
Enhancements to ICC Color Capabilities	27-2
Enhancements to the imdistline Function	27-2
setColor Method Accepts Predefined Color Strings	27-3

R2006a

Enhanced ICC Profile Capabilities	28-2
New Pointer Management Functions	28-2
New Constraint Creation Function	28-2
Functions cp2tform, tforminv, imtransform	28-2
IPPL Not Used on 64-Bit Systems	28-3

R14SP3

Support for Two New Medical Image File Formats	29-2
New Point, Rectangle, and Line Functions	29-2
Image Tool Enhancements and Improvements	29-2
New Distance Tool	29-2
Adjust Contrast Tool Enhancements and Improvements	29-2
New Utility Functions for Use with Profile-Based Color Space Conversion Functions	29-2
New Documentation on Processing Image Sequences	29-3
Control Point Selection Tool Now Works on Macintosh Systems	29-3
Obsolete and Deleted Functions	29-3
Image Tool is Not Compilable	29-3

R14SP2

Major Bug Fixes	30-2
Major Revisions to Fan-Beam Functions	30-2
Changes to the DICOM Functions	30-2
Changes to Image Tool and Modular Interactive Tools	30-3
Changes to the imshow Function	30-4
Fixes to Other Functions	30-4
Fixes to Image Processing Toolbox Deployment Issues	30-5
New Directory Needed	30-5

R2019b

Version: 11.0

New Features

Bug Fixes

Big Images: Process images that are too large to fit in memory

Read and process images that are too large to fit in memory by using a `bigimage` object. A `bigimage` supports images with one or multiple resolution levels.

Manage a collection of image blocks that belong to one or more `bigimage` objects by using a `bigimageDatastore`.

Display big image data by using the `bigimageshow` function.

Deep Learning Data Preprocessing: Perform additional image augmentations

You can augment images for network training using more image processing operations.

- Apply randomized color jitter to 2-D RGB images by using the `jitterColorHSV` function.
- Calculate randomized affine transformations, including reflection, rotation, rescaling, and translation, by using the `randomAffine2d` and `randomAffine3d` functions. Apply the transformations to images by using the `imwarp` function.
- Crop 2-D and 3-D images from a random position in the image by using the `randomCropWindow2d` and `randomCropWindow3d` function, respectively.
- Crop 2-D and 3-D images from the image center by using the `centerCropWindow2d` and `centerCropWindow3d` function, respectively.

Random Patch Extraction Datastore: Extract patches from 3-D data and transformed datastores, and train in parallel

You can now use a `randomPatchExtractionDatastore` to extract random patches from 3-D volumetric input. You can also extract patches from `TransformedDatastores` that have an underlying datastore of type `ImageDatastore` or `PixelLabelDatastore`.

`randomPatchExtractionDatastore` now supports parallel training (requires Parallel Computing Toolbox™).

Deep Learning: Added example using deep neural networks

The Deep Learning Classification of Large Multiresolution Images example shows how to train a neural network to classify very large multiresolution images that do not fit in memory by using `bigimage` and `bigimageDatastore`.

inpaintExemplar Function: Fill damaged regions in images with exemplar-based inpainting

The `inpaintExemplar` function performs exemplar-based inpainting for object removal and region filling in 2-D grayscale and RGB images. You can interactively select a region by using ROI Creation Convenience Functions, such as `drawfreehand`. After selecting a region, perform image inpainting of that region by using the `inpaintExemplar` function. For an example, see “Interactive Image Inpainting Using Exemplar Matching”.

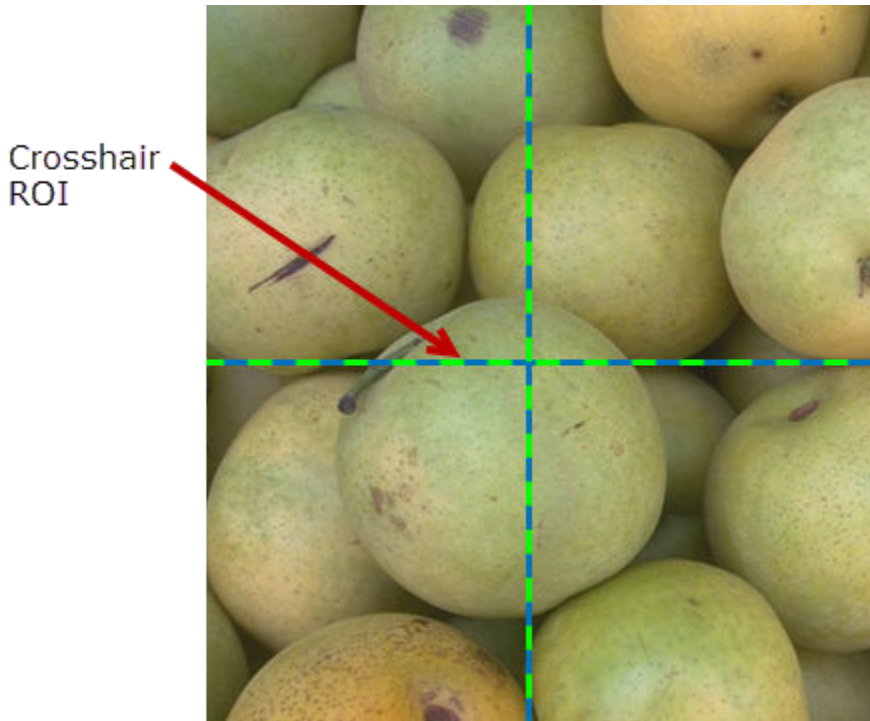
DICOM Volume: Construct isotropic volume from DICOM images

The `dicomreadVolume` function can now construct an isotropic volume from a given set of DICOM images. Use the new name-value pair `'MakeIsotropic', true` in `dicomreadVolume` function to return isotropic volume at the output.

ROI Tools: Create crosshair shape and other enhancements

The ROI tools include these enhancements.

- Crosshair ROI — Use the `drawcrosshair` function or `Crosshair` object to create a crosshair ROI in an axes.

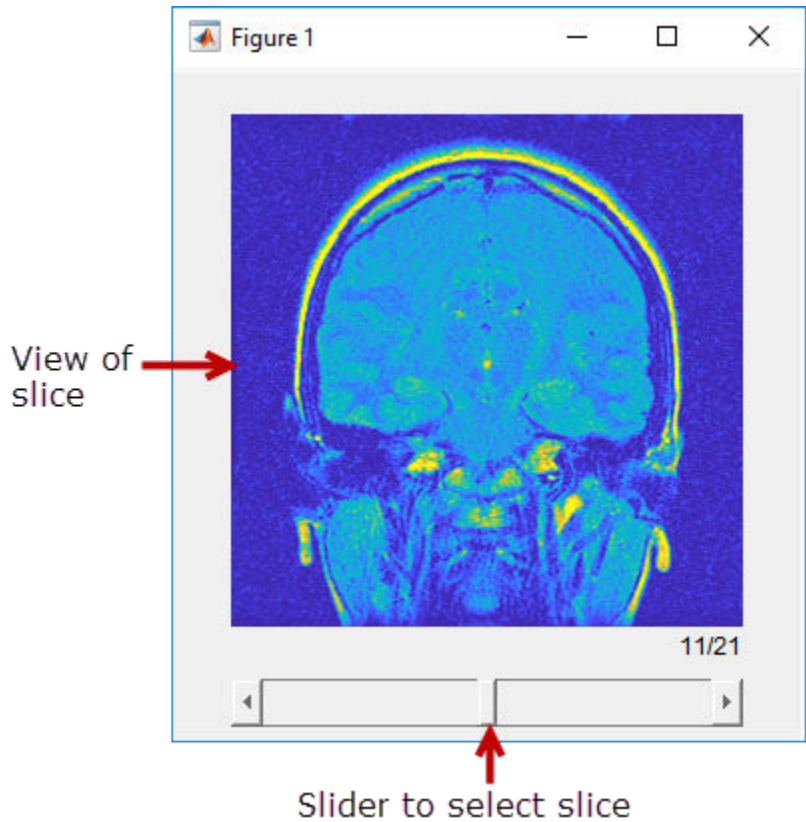


- **wait method** — All ROI classes now support the `wait` method so that the classes can be used in scripts. For example, using the `wait` method, you can enable users of your script to make the initial placement of the ROI, adjust the ROI and accept it, and then use the ROI position in the script to create a mask.
- **reduce method** — The polygon, polyline, freehand, and assisted freehand ROIs now support the `reduce` method. The `reduce` method uses the Douglas-Peucker algorithm to reduce the number of points used to define the ROI. For an example, see “Subsample or Simplify a Freehand ROI”.
- **Support for UIAxes** — You can now create an ROI in a UIAxes. For information about limitations, see “Using ROIs in Apps Created with App Designer”.

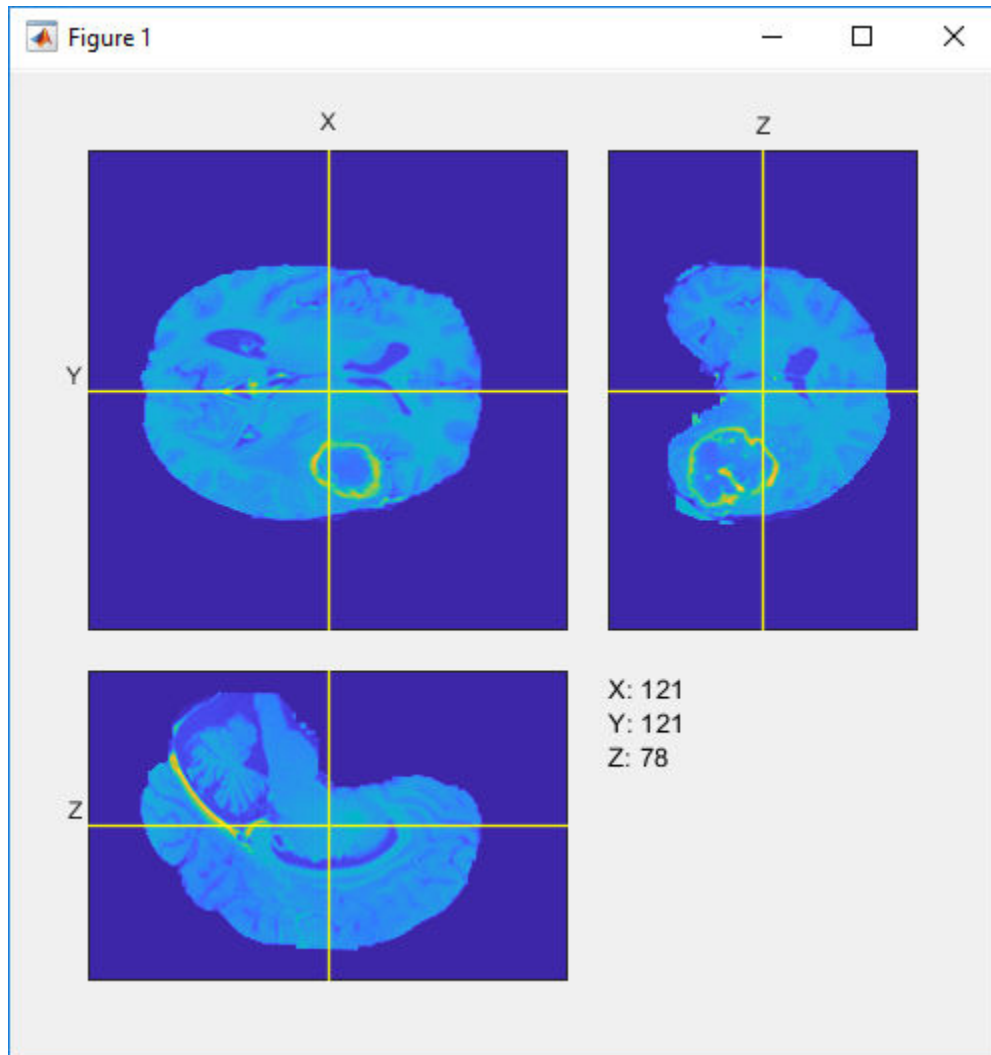
View 3-D Volumes as Slice Planes

Use `sliceViewer` and `orthosliceViewer` to view 3-D grayscale or RGB volumes as slice planes.

sliceViewer displays the center slice plane when it opens. You can navigate through the other slices by moving the slider at the bottom of the figure.

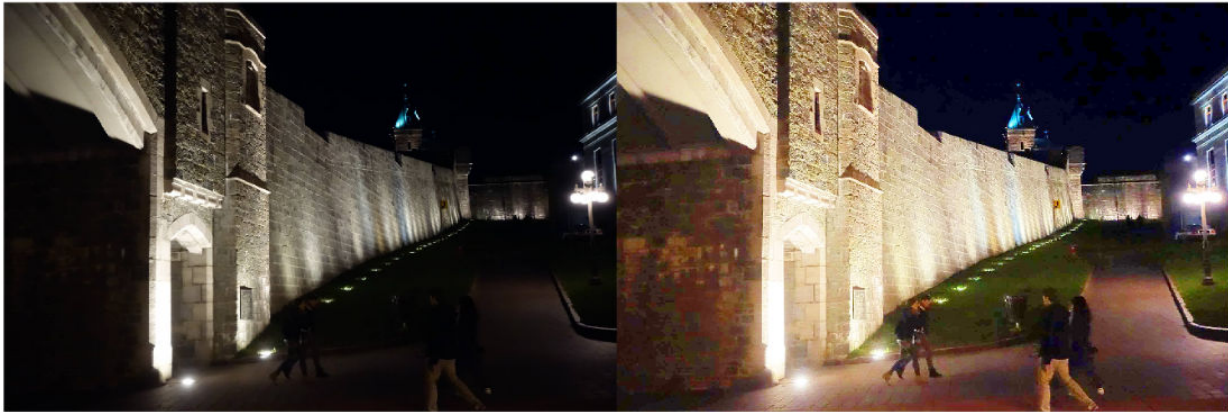


orthosliceViewer displays three views of the 3-D volume: the x-, y-, and z-planes. You can navigate through each view of the 3-D volume by moving the crosshair in the figure.



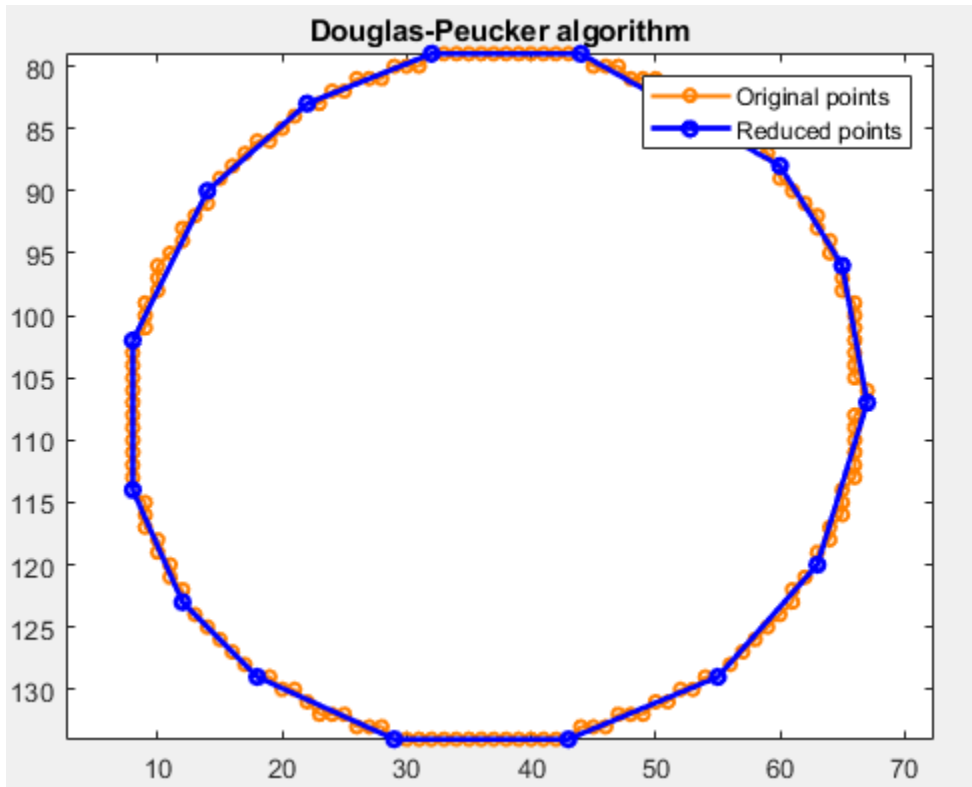
imlocalbrighten Function: Brighten dark areas of images

The `imlocalbrighten` function brightens dark areas of an image.



reducepoly Function: Reduce density of points in ROIs

The `reducepoly` function reduces the density of points in an ROI, by using the Douglas-Peucker algorithm.



Volume Viewer App: Create new session, export visualization settings, and other enhancements

The **Volume Viewer** app includes several new capabilities:

- **New Session** button — Use this button to clear all the data currently in the app.
- **Export** button — Use this button to save the current rendering and camera configurations in the window. This option returns the configuration in a structure. To replicate a view in the Volume Viewer, pass this structure to the `volshow` function.
- Additional alphamaps presets — In the Rendering Editor, use several new preset alphamaps for viewing CT and MRI data, including maximum intensity projection alphamaps.

imshow Function: Specify interpolation method

The `imshow` function now enables you to choose the interpolation method used when scaling an image. You can choose either nearest-neighbor ('nearest') or bilinear ('bilinear') interpolation for the 'Interpolation' Name/Value pair argument.

volshow Function: Control lighting in volume rendering

The `volshow` function now enables you to control lighting in rendering volumes, similar to the lighting control in the Volume Viewer.

affineOutputView Function: Control view of warped images

The `affineOutputView` function creates a spatial referencing object that can be used with the 'OutputView' argument of `imwarp` to control the output limits and grid spacing of a warped image.

Image Cropping: Crop 3-D volumes and crop using spatial referencing

The `imcrop3` function enables you to crop 3-D volumetric images.

The `Rectangle` and `Cuboid` objects store the spatial extents of 2-D and 3-D cropping windows, respectively. The `imcrop` function now accepts a `Rectangle` input argument that specifies the size and position of a 2-D cropping window. Likewise, the `imcrop3` function accepts a `Cuboid` input argument that specifies the size and position of a 3-D cropping window.

Support for Categorical Data

These functions now accept categorical data as input and output categorical data as outputs: `imcrop`, `imresize`, `imresize3`, and `imwarp`.

C-code generation: Generate code from one additional function using MATLAB Coder

This function supports the generation of C code using MATLAB® Coder™. If you choose the generic MATLAB Host Computer target platform, the function generates code that uses a precompiled, platform-specific shared library.

```
imregcorr
```

R2019a

Version: 10.4

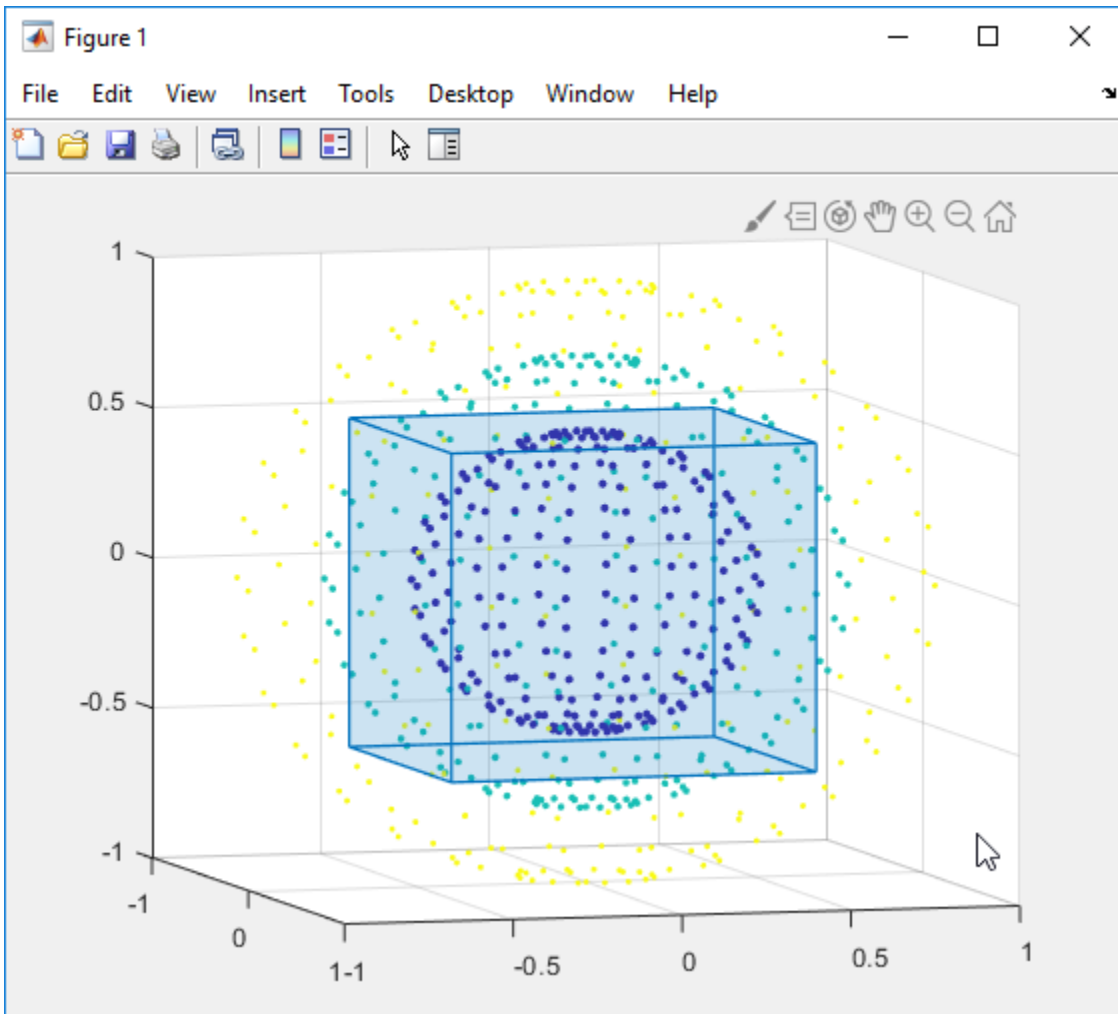
New Features

Bug Fixes

Compatibility Considerations

ROI Creation Functions: New cuboid shape added

You can create cuboidal ROIs for 3-D volumes with the new `drawcuboid` function or the `images.roi.Cuboid` class. As with the other ROIs, cuboidal ROIs support properties, methods, and events that let you customize their appearance and behavior.



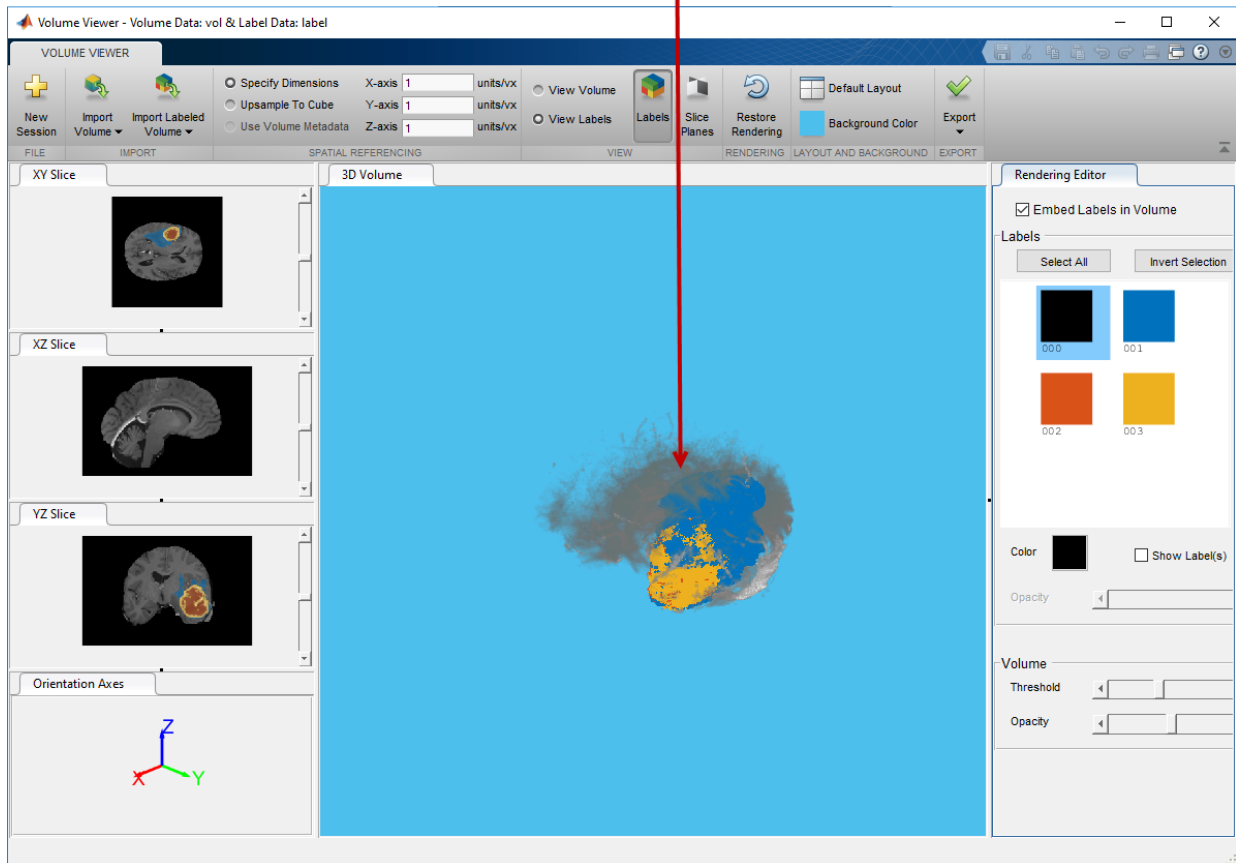
ROI Creation Functions: New bringToFront Function

You can now change the visual stacking order of ROIs using the new bringToFront function.

Enhanced Volume Display: View labeled volumes and specify colormap

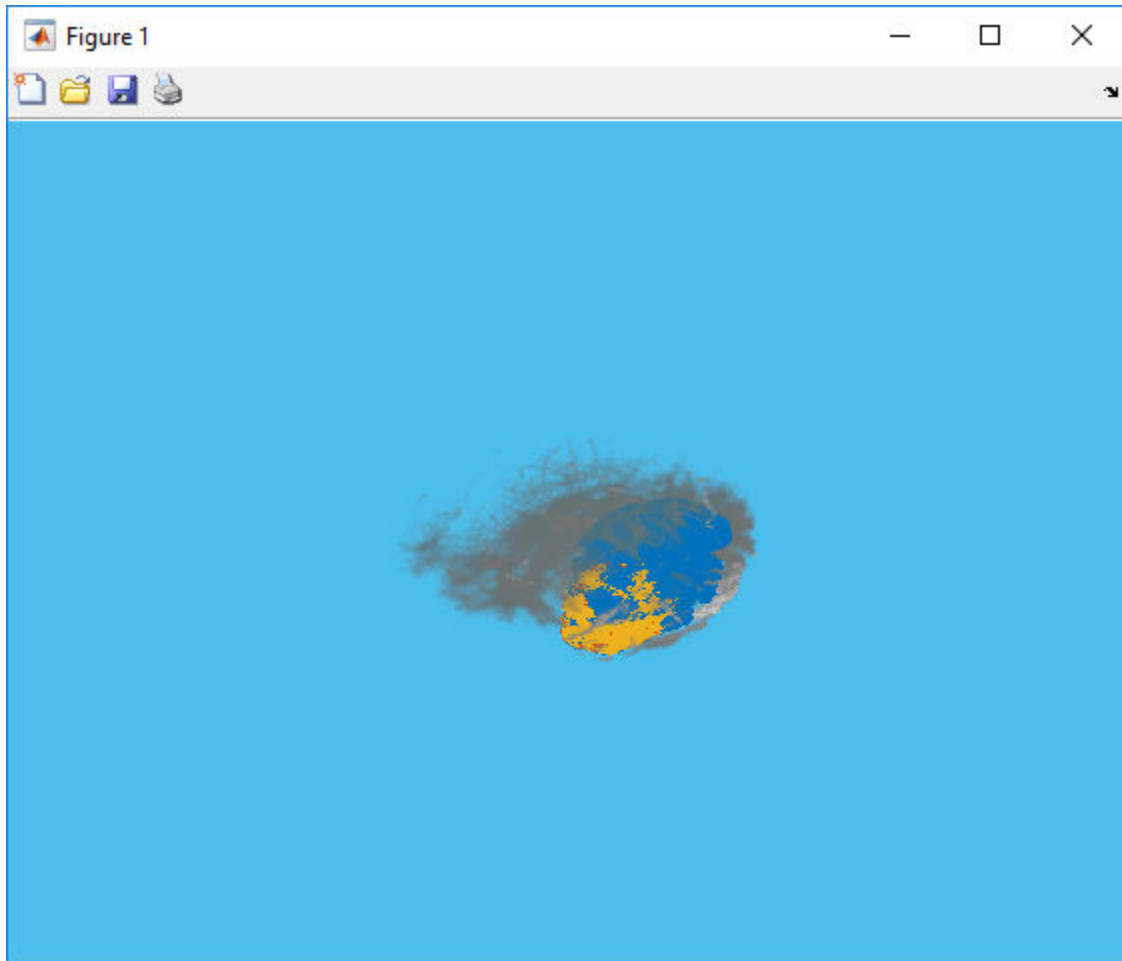
The Volume Viewer app now supports the display of 3-D labeled (segmented) volumes. You can also view a labeled volume overlaid on an intensity volume, to view the labels in the context of the original volume.

View labeled volume and intensity volume together.



The Volume Viewer app now enables you to change the colormap of the displayed intensity volume data. You can select a built-in MATLAB colormap or specify your own colormap.

Instead of using the Volume Viewer app, you can also display 3-D labeled volumes in a figure by using the `labelvolshow` function. As with the Volume Viewer, you can also view a labeled volume overlaid on an intensity volume.



Deep Learning: Added example using deep neural networks

The 3-D Brain Tumor Segmentation using Deep Learning example shows how to train a neural network to perform semantic segmentation of a 3-D volume.

Measurements of Region Properties: Measure circularity and Feret properties

The `regionprops` function now measures the circularity and Feret properties of regions in a binary image. To measure the circularity, include `'Circularity'` when specifying the `properties` input argument. To compute the minimum or maximum Feret properties, include `'MinFeretProperties'` or `'MaxFeretProperties'`, respectively, when specifying the `properties` input argument.

Alternatively, you can measure the minimum or maximum Feret properties of regions in a binary image by using the added `bwferet` function. This table shows the minimum and maximum Feret properties.

Minimum Feret Properties	Maximum Feret Properties
Minimum Feret diameter	Maximum Feret diameter
Angle of minimum Feret diameter	Angle of maximum Feret diameter
Endpoint coordinates of minimum Feret diameter	Endpoint coordinates of maximum Feret diameter

NIFTI File Format Enhancements: Read and write neuroscience image volumes in the NIfTI-2 file format

NIFTI functions `niftiinfo`, `niftiread`, and `niftiwrite` now support NIfTI-2 file formats.

- The metadata returned by the `niftiinfo` function now includes an additional field called `Version`. The value of this field is either `'NIfTI1'` or `'NIfTI2'`, depending on the input file format.
- The `niftiwrite` function now includes an additional name-value pair `Version`. To write a file in NIfTI-2 file format, set `Version` to `'NIfTI2'`.

Camera Response: Estimate real-world illumination as a function of pixel intensity

The `camresponse` function calculates a camera's response function using a series of images captured with different exposure times. The camera response function characterizes the sensitivity of a camera sensor by mapping pixel intensities in an input image to the amount of real-world illumination reaching the sensor.

The `makehdr` function now accepts an optional camera response argument. Providing a camera response argument can improve the accuracy of a high dynamic range image created from a series of low dynamic range images.

inpaintCoherent Function: Fill damaged regions in images with coherence transport based inpainting

The `inpaintCoherent` function performs coherence transport based inpainting for object removal and region filling in 2-D grayscale and RGB images.

burstinterpolant Function: Generate a high-resolution image from a burst of lower resolution images

The `burstinterpolant` function generates a high-resolution image from a set of low-resolution images captured in burst mode. The low-resolution images can be 2-D grayscale or RGB color images.

rgb2lightness Function: Convert an RGB image to a lightness image

The `rgb2lightness` function converts an RGB image to a lightness image. The lightness image corresponds to the lightness component in the CIE 1976 L*a*b* color space.

Performance improvements: Performance enhancements for image warping

The performance of the following function has been improved:

- `imwarp` (2-D and 3-D)

C-code generation: Generate code from four additional functions using MATLAB Coder

The following table lists the Image Processing Toolbox functions that have been enabled for code generation in this release. For all target platforms, these functions generate C code. For a complete list of Image Processing Toolbox functions that support code generation, see [Functions Supporting Code Generation](#).

affine3d	inpaintCoherent	rgb2lightness
----------	-----------------	---------------

In addition, the following function that already supported code generation has additional capabilities. This function supports the generation of C code using MATLAB Coder. Note that if you choose the generic MATLAB Host Computer target platform, then the function generates code that uses a precompiled, platform-specific shared library.

Function	New Capability
regionprops	Supports the circularity property for code generation.

Functionality Being Removed or Changed

The `imshow` function now displays large images passively

Behavior change

The `imshow` function now handles large images passively. The function does not return a warning message while displaying large images at the largest magnification that fits on the screen. You can modify the initial magnification value by changing the preferred percentage value in the Image Processing Toolbox Preferences dialog box. To access the dialog box, click **Preferences** on the **Home** tab in the MATLAB desktop, or call the `iptprefs` function.

The `imfindcircles` function uses new filter size for logical images

Behavior change

Starting in R2019a, the `imfindcircles` function uses a 5-by-5 filter size for smoothing logical images. `imfindcircles` may now return a different answer than in previous releases, when the filter size was 6-by-6. For example, in some instances, the function may return a different number of circles.

R2018b

Version: 10.3

New Features

Bug Fixes

Compatibility Considerations

Random Patch Extraction Datastore: Extract random image patches to split up large images for deep learning workflows

The `randomPatchExtractionDatastore` object extracts randomly-positioned patches from training images and corresponding patches from images representing the desired network output, for training deep neural networks. The `randomPatchExtractionDatastore` object can also extract randomly-positioned patches from ground truth images and corresponding patches from pixel label data, for training semantic segmentation networks.

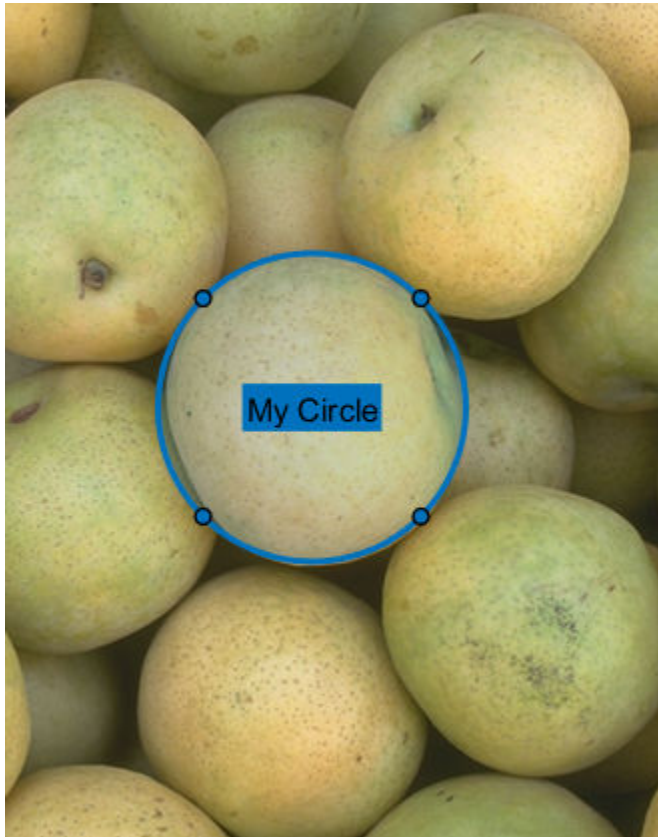
The Single Image Super-Resolution Using Deep Learning example has been updated to use a random patch extraction datastore, instead of a custom mini-batch datastore, as a source of training images.

Deep Learning: Added example using deep neural networks

The Image Processing Operator Approximation Using Deep Learning example shows how to train a neural network to transform an image such that the resulting image resembles the output of a traditional image processing pipeline.

New Set of ROI Creation Functions

The toolbox includes a new set of region-of-interest (ROI) creation functions. Use these functions to create ROIs of many shapes, including circular or polygonal ROIs and ROIs drawn as freehand shapes. A new type of ROI, called assisted freehand, lets you draw a freehand ROI that snaps to edges of existing objects in the image.



The functions create ROI objects. You can change properties of the ROI objects to modify the appearance and behavior of the ROI. For more advanced workflows, such as designing interactive apps, the ROI objects support events and listeners.

ROI Creation Functions	ROI Object	Description of ROI
<code>drawassisted</code>	<code>images.roi.AssistedFreehand</code>	Freehand ROI that snaps to edges of existing objects in the image
<code>drawcircle</code>	<code>images.roi.Circle</code>	Circular ROI
<code>drawellipse</code>	<code>images.roi.Ellipse</code>	Ellipsoid ROI

ROI Creation Functions	ROI Object	Description of ROI
<code>drawfreehand</code>	<code>images.roi.Freehand</code>	Freehand ROI that follows the path of the mouse
<code>drawline</code>	<code>images.roi.Line</code>	Linear ROI that consists of a single line segment
<code>drawpoint</code>	<code>images.roi.Point</code>	Point ROI
<code>drawpolygon</code>	<code>images.roi.Polygon</code>	Polygonal ROI that consists of a closed set of line segments
<code>drawpolyline</code>	<code>images.roi.Polyline</code>	Polyline ROI that consists of an open set of line segments
<code>drawrectangle</code>	<code>images.roi.Rectangle</code>	Rectangular ROI

You can use the ROI objects as an input to four new functions. The `createMask` function creates a mask with pixels inside the specified ROI set to `true` and pixels outside the ROI set to `false`. The `inROI` function queries if points are within the ROI. The `draw` and `beginDrawingFromPoint` functions begin an interactive drawing mode for the ROI object, maintaining the appearance of the displayed ROI.

These new ROI functions are recommended over the existing set of ROI functions (see “Old ROI classes are not recommended” on page 3-7).

Volume Show: Visualize 3-D image volumes using the `volshow` command

The `volshow` function displays a 3-D grayscale image volume in a figure.

Image Segmentation: Segment 2-D images and N-D volumes using k-means clustering

The `imsegkmeans` and `imsegkmeans3` functions segment images or volumes using k-means clustering.

Geometric Transformation Objects: Represent and apply custom 2-D and 3-D geometric transformations

The `geometricTransform2d` and `geometricTransform3d` objects define 2-D and 3-D geometric transformations using point-wise mapping functions. The objects enable you to apply custom inverse and forward geometric transformations to images.

`fspecial3`: Create predefined 3-D filters

The `fspecial3` function creates predefined 3-D filters of various types, including ellipsoidal averaging filters, Laplacian of Gaussian filters, and edge-detecting Prewitt and Sobel filters.

`imflatfield`: Perform flat-field correction

The `imflatfield` function applies flat-field correction to 2-D grayscale and RGB images.

`imnlmfilt`: Perform non-local means filtering

The `imnlmfilt` function performs edge-preserving, non-local means filtering of 2-D grayscale and RGB images.

`imsplit`: Split an N-channel image into individual channels

The `imsplit` function creates a set of n images representing each channel in an n -channel image. For an example, see [Display Separated Color Channels of an RGB Image](#).

`piqe`: Measure image quality using perception-based image quality evaluator (PIQE)

The `piqe` function calculates an opinion-unaware no-reference quality score for natural images using perception-based features.

tonemapfarbman: Reduce dynamic range of HDR images

The `tonemapfarbman` function converts high dynamic range (HDR) images into a format that can be displayed using edge-preserving multi-scale decompositions.

fibermetric: Added 3-D support

The `fibermetric` function now supports 3-D grayscale input volumes.

Processing volumetric input requires a performance improvement, which is achieved by a new default value of the `StructureSensitivity` argument of `fibermetric`. The function `maxhessiannorm` is introduced to help calculate the prior default value of `StructureSensitivity` for 2-D images.

Performance improvements: Performance enhancements for 2-D and 3-D morphology, image warping, and fibermetric

The performance of the following functions has been improved:

- `imerode`
- `imdilate`
- `imopen`
- `imclose`
- `imtophat`
- `imbothat`
- `imdiffusefilt`
- `fibermetric`

C-code generation: Generate code from three additional functions using MATLAB Coder

The following table lists the Image Processing Toolbox functions that have been enabled for code generation in this release. For all target platforms, these functions generate C code. For a complete list of Image Processing Toolbox functions that support code generation, see [List of Supported Functions with Usage Notes](#).

imsplit	fspecial3
---------	-----------

The function listed in the following table previously only generated C code, and can now also generate C code that uses a platform-specific shared library. To use a shared library, choose the generic MATLAB Host Computer option in the MATLAB Coder configuration settings.

bwdist	
--------	--

Functionality Being Removed or Changed

fibermetric calculates new default structural sensitivity

Behavior change

Starting in R2018b, `fibermetric` calculates the default value of the `StructureSensitivity` argument for grayscale image `I` as $0.01 * \text{diff}(\text{getrangefromclass}(I))$. The change in the default value increases computational efficiency and reduces memory usage. These improvements also enable processing 3-D volumetric images.

Previous versions of `fibermetric` defined the default value of `StructureSensitivity` as half of the maximum of the Hessian norm of the image. If you want to reproduce the prior default value for 2-D images, then specify `StructureSensitivity` as $0.5 * \text{maxhessiannorm}(I)$. The `maxhessiannorm` function does not support 3-D input.

Old ROI classes are not recommended

Still runs

Old ROI classes (listed in the first column of the table) are not recommended. Use the new ROI functions and classes instead. The new ROI functions and classes have these advantages.

- More functional capabilities, such as face color transparency.
- Support for events. Use events to respond to changes in your ROI such as moving or being clicked.
- Simplified access to the ROI. The new classes return a single object representing the ROI. In contrast, the old classes return the ROI as a group of lines and patch objects.

There are no plans to remove the old ROI classes.

Functionality	Result	Use Instead	Compatibility Considerations
<code>imellipse</code>	Still runs	<code>drawellipse</code> or <code>drawcircle</code>	If you use the <code>imellipse</code> syntax where you specify the position, replace the position vector with the name-value pairs <code>Center</code> and <code>Semiaxes</code> . To specify the position and size of the circle, use the <code>Center</code> and <code>Radius</code> name-value pairs.
<code>imfreehand</code>	Still runs	<code>drawfreehand</code> or <code>drawassisted</code>	None
<code>imline</code>	Still runs	<code>drawline</code>	If you use the <code>imline</code> syntax where you specify the position, replace the position vector with the <code>Position</code> name-value pair.
<code>impoint</code>	Still runs	<code>drawpoint</code>	If you use the <code>impoint</code> syntax where you specify the position, replace the position vector with the <code>Position</code> name-value pair.

Functionality	Result	Use Instead	Compatibility Considerations
<code>impoly</code>	Still runs	<code>drawpolygon</code> or <code>drawpolyline</code>	If you use the <code>impoly</code> syntax where you specify the position, replace the position vector with the <code>Position</code> name-value pair.
<code>imrect</code>	Still runs	<code>drawrectangle</code>	If you use the <code>imrect</code> syntax where you specify the position, replace the position vector with the <code>Position</code> name-value pair.

R2018a

Version: 10.2

New Features

Bug Fixes

Compatibility Considerations

Deep Learning: Added examples using deep neural networks

The following featured examples show how to solve image processing problems by using deep neural networks.

- The Single Image Super-Resolution Using Deep Learning example shows how to recover a high-resolution image from a low-resolution image by using a Very-Deep Super-Resolution (VDSR) neural network. The example shows how to set up and train a VDSR network. The example implements a custom mini-batch datastore, called a `vdsrImagePatchDatastore`, that creates batches of upsampled low-resolution patches and corresponding residual patches, with support for shuffling during training.
- The JPEG Image Deblocking Using Deep Learning example shows how to remove JPEG compression artifacts from images by using a DnCNN network. The example shows how to set up and train the DnCNN network. The example implements a custom mini-batch datastore, called a `JPEGImagePatchDatastore`, that extracts patches from input distorted images and computes the target residual images.
- The Semantic Segmentation of Multispectral Images Using Deep Learning example shows how to use perform semantic segmentation of an image with data in seven channels: three infrared channels, three RGB color channels, and a mask. The example shows how to set up and train a U-Net network to perform the segmentation. The example implements a custom mini-batch datastore, called a `PixelLabelImagePatchDatastore`, that extracts patches from the multispectral images and the corresponding labels.

Deep Learning Data Preprocessing: Efficiently read and add noise to images for training and prediction

A `denoisingImageDatastore` preprocesses training images by adding Gaussian noise. You can use a `denoisingImageDatastore` for both training and prediction.

Image Segmenter: Segment images interactively using new techniques including local graph cut

The **Image Segmenter** app now includes the local graph cut segmentation technique. For an example of segmenting an image by using the Image Segmenter, see [Image Segmentation Using the Image Segmenter App](#).

Edge-Aware Filtering: Smooth images and reduce noise while preserving edge sharpness with bilateral filtering and anisotropic diffusion filtering

The `imdifffusefilt` function performs anisotropic diffusion filtering of images. The `imbilatfilt` function performs edge-aware bilateral filtering of images by using Gaussian kernels.

blendexposure: Perform exposure fusion

The `blendexposure` function blends images that have different exposures into a single well-exposed image.

imregmtb: Register images using median threshold bitmaps

The `imregmtb` function registers images using the median threshold bitmap technique. This technique is useful for registering images that have camera or scene motion, or that have different exposures.

montage: Display multiple images from ImageDatastore object, and specify size of the image thumbnails displayed

The `montage` function now can display images in an `ImageDatastore`. You can now specify the size of the image thumbnails displayed in the montage. Additionally, `montage` no longer requires *m-by-n-by-1-by-p* inputs for volume inputs. Instead, you can specify an *m-by-n-by-p* array.

Starting in R2018a, there are several changes to the behavior of `montage`. See [Compatibility Considerations](#).

Image Morphology: Perform morphological operations on 3-D volumes, and perform skeletonization on all objects in 2-D image or 3-D binary volume

You can perform morphological operations on 3-D volumes, and perform skeletonization on all objects in a 2-D image or 3-D binary volume.

The `bwmorph3` function performs morphological operations on 3-D volumes.

The `bwskel` function reduces all objects in the 2-D binary image or 3-D binary volume to curved lines, without changing the essential structure of the image.

Performance: Improved performance in functions including 3-D `imwarp`, 3-D `imfilter`, `entropyfilt`, `ordfilt2`, `medfilt2`, and `bwmorph`

The performance of the following functions has been improved:

- `bwmorph`
- `entropyfilt`
- `imfilter` (3-D)
- `imwarp` (3-D)
- `medfilt2`
- `ordfilt2`

C-code generation: Generate code from one additional function using MATLAB Coder

The following table lists the Image Processing Toolbox function that has been enabled for code generation in this release. For all target platforms, this function generates C code. This function can also generate C code that uses a precompiled, platform-specific shared library (`.dll`, `.so`, or `.dylib`). Using a shared library preserves performance optimizations in this function but limits the target to only those platforms that support MATLAB (see system requirements). For a complete list of Image Processing Toolbox functions that support code generation, see [List of Supported Functions with Usage Notes](#).

<code>imbilatfilt</code> ¹	
---------------------------------------	--

¹ If you choose the generic MATLAB Host Computer option in the MATLAB Coder configuration settings, this function generates C code that uses a precompiled, platform-specific shared library.

Functionality Being Removed or Changed

montage function has several behavior changes

Behavior change

Starting in R2018a, there are several changes to the behavior of montage.

Previous Behavior	New Behavior	Compatibility Mode
If you pass a single truecolor (RGB) image to <code>montage</code> , then the function displays a single truecolor image.	<code>montage</code> displays each color channel of an RGB image as three separate grayscale images.	Use the <code>imshow</code> function to display a truecolor image <code>imshow(RGB)</code> .
If one or more of the image files contained an indexed image, then <code>montage</code> used the colormap from the first indexed image file.	<code>montage</code> converts any indexed image into its corresponding RGB version using the internal colormap present in the file.	Explicitly read the colormap of the first indexed image and then use the syntax <code>montage(filenamees, colormap)</code> .
Each constituent thumbnail preserved the full image size.	Each thumbnail is resized to the specified (or default) <code>ThumbnailSize</code> with appropriate zero-padding.	Use the syntax <code>montage(..., 'Thumbnail', [])</code> .
If you specify a set of indexed images, then <code>montage</code> used the colormap of the first image to set the colormap of the axes. This enabled you to change the colormap of all the images displayed after creating the montage by changing the colormap of the axes.	If you specify a set of indexed images, then <code>montage</code> converts each one to an RGB image using its internal colormap or the colormap specified in the command line. Changing the colormap of the axes has no effect.	Explicitly specify the colormap at time of creation, using the syntax <code>montage(filenamees, colormap)</code> .

denoisingImageSource object is removed

In 2017b, you could create a `denoisingImageSource` object for training deep learning networks. Starting in R2018a, the `denoisingImageSource` object has been removed. Use a `denoisingImageDatastore` object instead.

A `denoisingImageDatastore` has additional properties and methods to assist with data preprocessing. Unlike `denoisingImageSource`, which could be used for training only, you can use a `denoisingImageDatastore` for both training and prediction.

To create a `denoisingImageDatastore` object, you can use either the `denoisingImageDatastore` function (recommended) or the `denoisingImageSource` function.

denoisingImageSource function will be removed

Still runs

The `denoisingImageSource` function will be removed in a future release. Create a `denoisingImageDatastore` using the `denoisingImageDatastore` function instead.

To update your code, change instances of the function name `denoisingImageSource` to `denoisingImageDatastore`. You do not need to change the input arguments.

R2017b

Version: 10.1

New Features

Bug Fixes

Deep Learning: Denoise images using deep learning techniques

The toolbox includes the new `denoiseImage` function, which estimates the denoised version of a noisy image using a denoising deep neural network. You can use a pretrained network using the new `denoisingNetwork` function, or you can train your own network starting with layers provided by the new `dnCNNLayers` function. These functions require the Neural Network Toolbox™.

3-D Image Processing: Process 3-D volumetric image data with support for several additional functions

The toolbox includes several new functions that enable working with 3-D volumetric data: `regionprops3`, `edge3`, `bwselect3`, `jaccard`, `dice`, `bfscore`, and `dicomreadVolume`. The toolbox includes one new function that enables working with N-D data: `imadjustn`.

Five existing functions now support 3-D volumetric data: `imbinarize`, `adaptthresh`, `niftiinfo`, `niftiread`, and `niftiwrite`.

Image Enhancement: Adjust colors with automatic white balancing, and reduce haze in images

The toolbox includes several new functions that enhance the appearance of images.

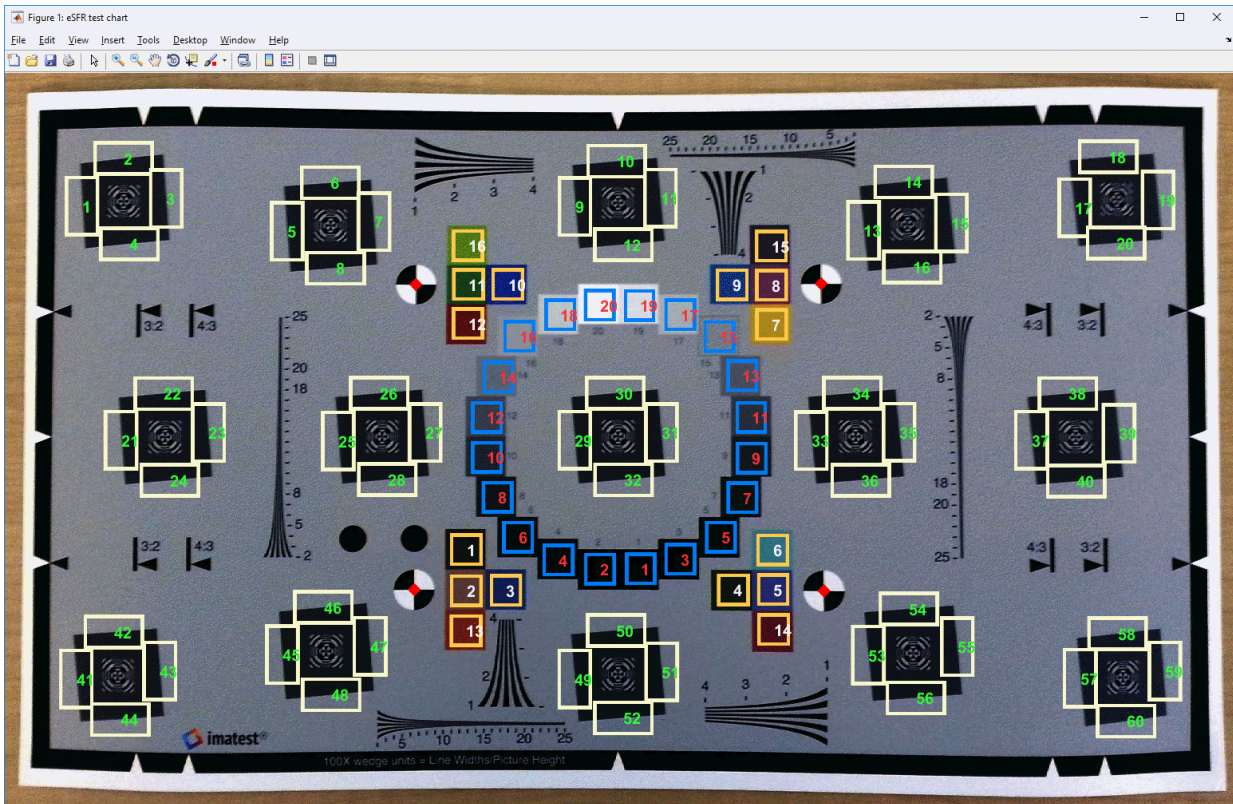
- The new `chromadapt` function adjusts the color balance of an sRGB image according to the scene illuminant. Three new functions, `illumgray`, `illumzca`, and `illumwhite`, estimate scene illumination using the Gray World algorithm, principal component analysis, and the White Patch Retinex algorithm.
- The new `imreducehaze` function reduces atmospheric haze in an RGB or grayscale image.
- The new `lin2rgb` function applies gamma correction to convert linear RGB images to sRGB or Adobe RGB (1998). The new `rgb2lin` function undoes gamma correction to linearize RGB images.

Image Quality Metrics: Measure image quality without a reference image, and model image quality using an eSFR test chart

The toolbox includes several new functions and objects that enable objective measures of quality from images.

- The new `brisque` and `niqe` functions calculate a no-reference quality score for images using trained models with natural scene statistics. You can train custom models using the `fitbrisque` and `fitniqe` functions. The models are stored in `brisqueModel` and `niqeModel` objects.
- The new `esfrChart` object automatically identifies the slanted edge, gray patch, and color patch regions of interest (ROIs) on an image of an edge spatial frequency response (eSFR) test chart from Imatest®. You can display the ROIs overlaid on the chart using the `displayChart` function.

You can measure several aspects of the eSFR test chart using the `measureSharpness`, `measureChromaticAberration`, `measureColor`, `measureNoise`, and `measureIlluminant` functions. You can display the measured sharpness using the `plotSFR` function. You can visually compare the measured and expected colors using the `displayColorPatch` and `plotChromaticity` functions.



NIFTI File Format: Read and write neuroscience image volumes in the NifTI file format

The toolbox includes three new functions that enable reading, writing, and reading image volumes from files in the file format defined by the Neuroimaging Informatics Technology Initiative (NIFTI). The functions are: `niftiinfo`, `niftiread`, and `niftiwrite`. This format is commonly used in the neuroimaging community for CT, MR, and PET data.

Image Segmenter: Segment images using new techniques including texture segmentation

The **Image Segmenter** app now includes several new segmentation techniques including texture segmentation using Gabor filtering. If you also have the Statistics and Machine

Learning Toolbox™, the Image Segementer includes an automatic clustering capability based on K-means processing. In addition, you can now load a pre-existing mask image into the Image Segementer to continue refining a mask.

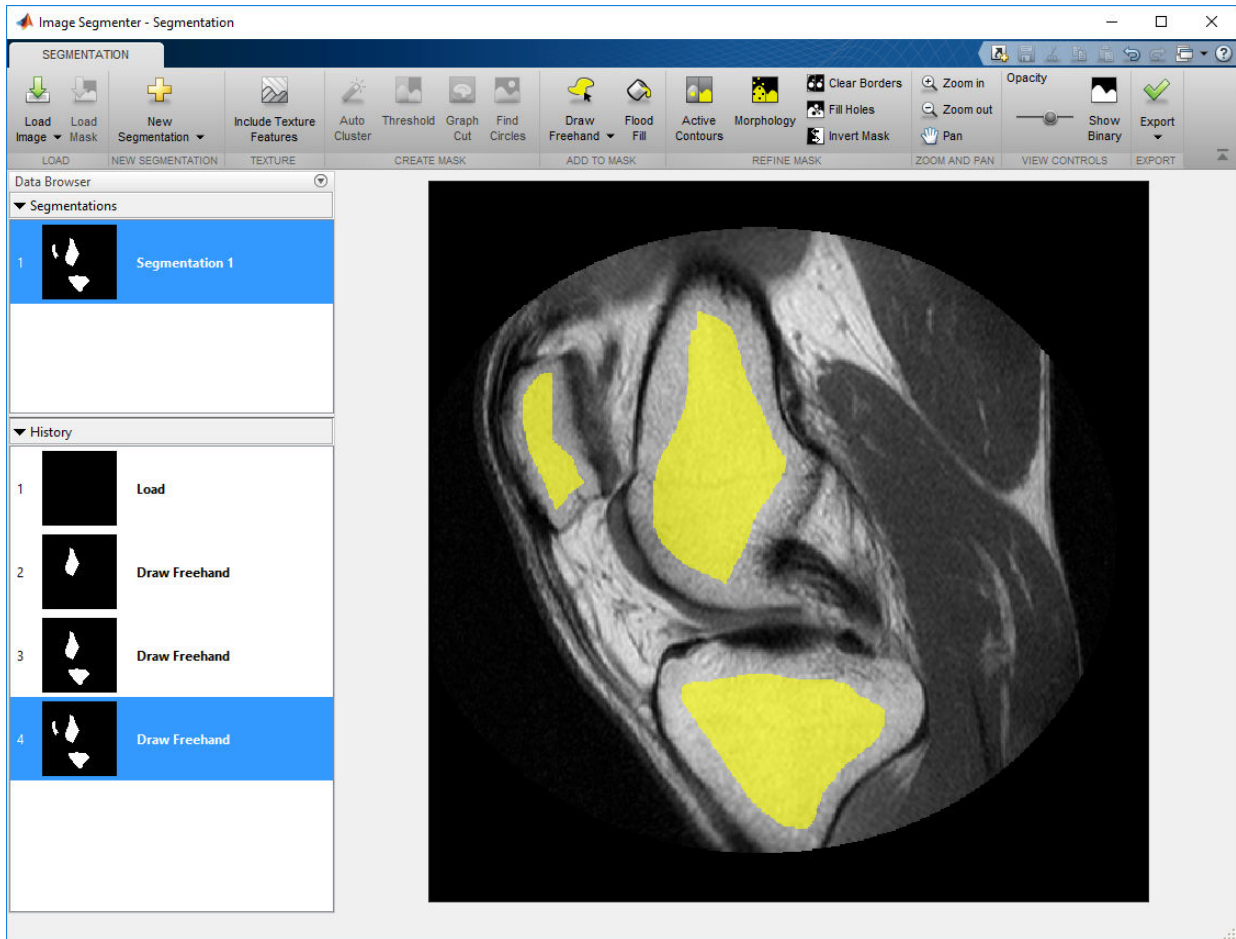


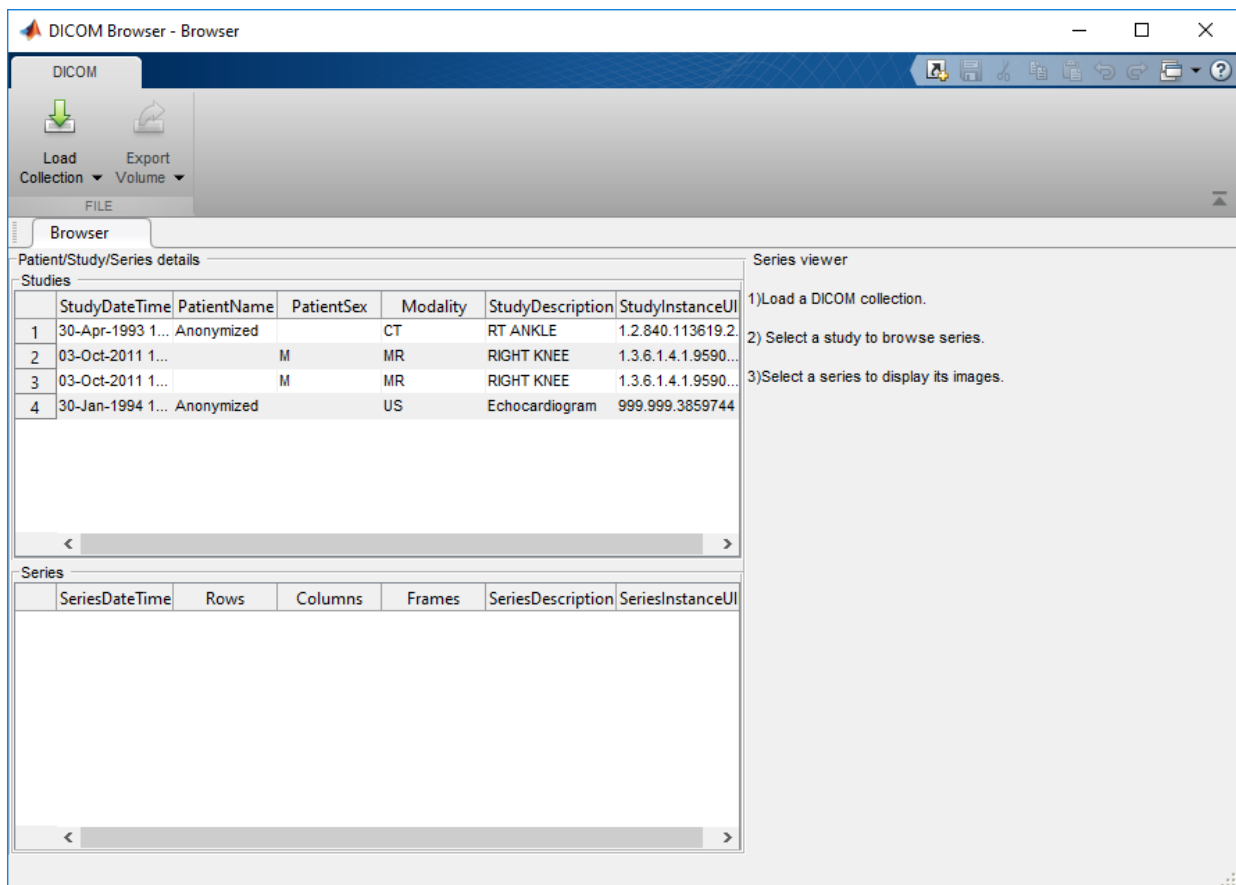
Image Segmentation: Calculate similarity coefficients

The jaccard, dice, and bfscore functions compute similarity coefficients that can be used to evaluate segmentation accuracy.

DICOM Browsing: Explore the contents of DICOM media in a browser and programmatically

The toolbox now includes several new functions and an app that you can use to explore the contents of DICOM files and folders.

The DICOM Browser app (**DICOM Browser**) lets you explore the contents of DICOM media.



In addition, you can read volumetric data from DICOM media using the new `dicomreadVolume` and `dicomCollection` functions. You can also decode DICOM UIDs

and parse the DICOMDIR file using `images.dicom.decodeUID` and `images.dicom.parseDICOMDIR` functions.

Image Warper: Transform group of images quickly using the `images.geotrans.Warper` object

The toolbox includes a new object, called `Warper`, that enables you to apply a geometric transformation to a group of images, all the same size. You identify the images, define the geometric transformation, and then create a `Warper` object. To apply the transformation to the images, use the `warp` function.

R2017a

Version: 10.0

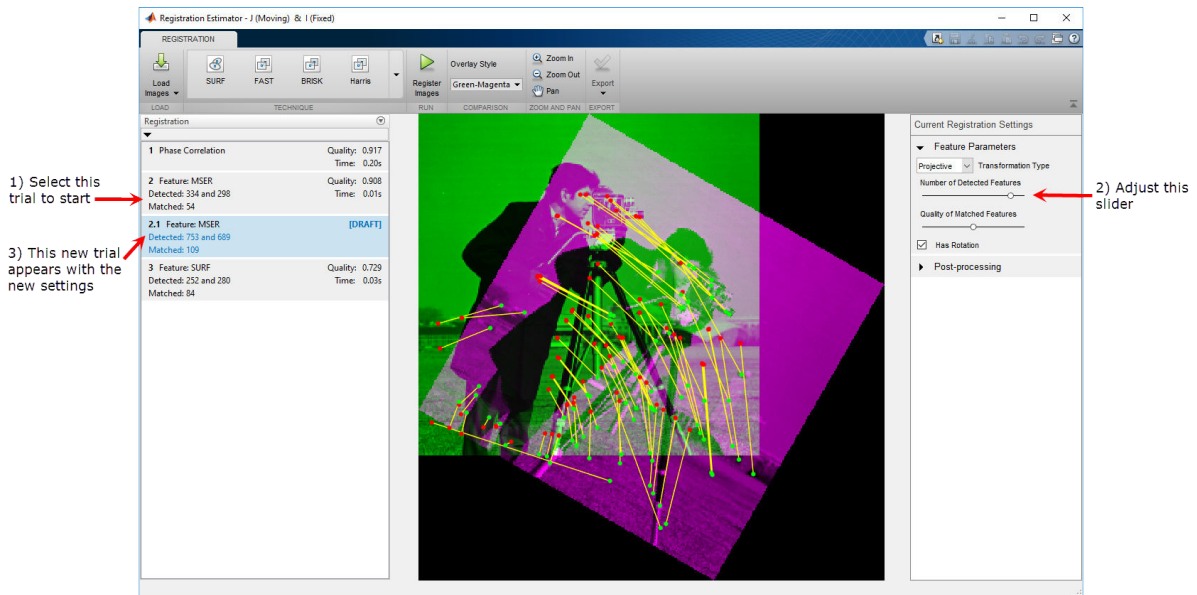
New Features

Bug Fixes

Compatibility Considerations

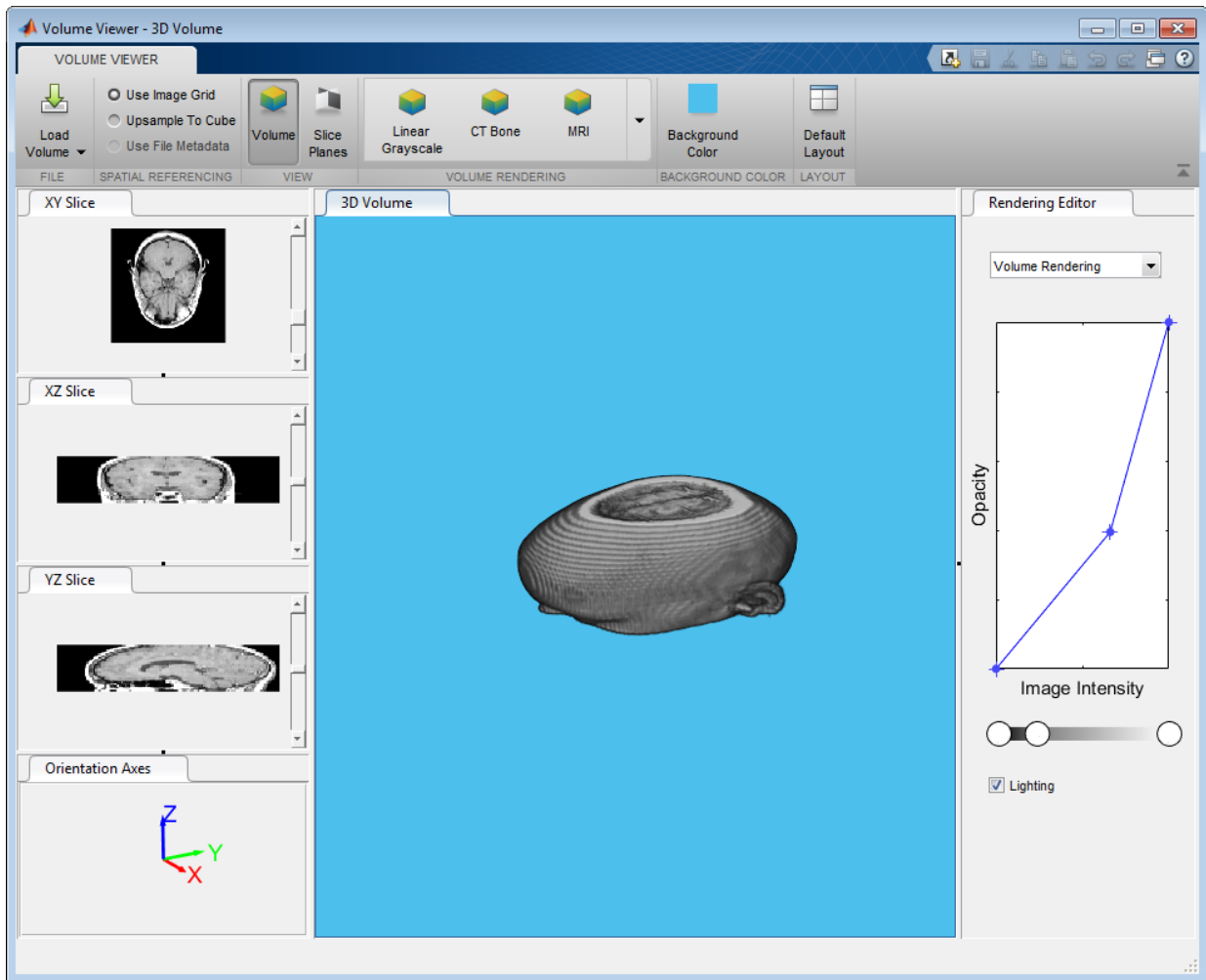
Image Registration App: Explore various registration techniques interactively to align images

The new image registration app, called Registration Estimator, lets you use several different registration techniques to bring two images into alignment. You can use feature-based, intensity-based, or nonrigid approaches. For more information, see Register Images Using the Registration Estimator App.



Volume Viewer App: View and slice 3-D volumetric data

The Volume Viewer app lets you view 3-D volumetric data as a volume or as a set of slices. The Volume Viewer provides a rendering editor that enables you to manipulate the mapping of intensity values to opacity to achieve different views of your 3-D data. The Volume Viewer includes several preset mappings to achieve certain well-known effects, such as emphasizing the visibility of bony structures or revealing inner structures in your data. For more information, see Explore 3-D Volumetric Data with the Volume Viewer App.



3-D Volumetric Data: Process 3-D volumetric image data with support for over a dozen functions

The toolbox now provides more support for processing 3-D volumetric data. This includes several new functions, `imresize3` and `imrotate3`. In addition, existing functions, such as `activecontour`, highlight their support of volumetric data, many with new 3-D

examples. To see an example of segmenting a volume, see Segment Lungs from 3-D Chest Scan and Calculate Lung Volume.

fibermetric: Enhance tubular or elongated structures in images

The toolbox includes the new function, `fibermetric`, that you can use to enhance elongated or tubular structures in an image.

Image Segmenter App: Added support for RGB images and graph cut segmentation

The Image Segmenter app now includes support for RGB images and segmentation using the graph cut technique. Graph cut is a semi-automatic technique of segmenting the foreground from the background in an image. For more information, see Segment Image Using Graph Cut.

lazysnapping: Segmentation technique

The toolbox includes the new function, `lazysnapping`, that you can use to segment an image foreground from the background.

N-D histograms: Enhance contrast and adjust histograms of N-D images

The toolbox includes a new function, `imhistmatchn`, that allows you to match an N-D histogram to a reference histogram. The toolbox adds N-D support to two other functions, `imhist` and `histeq`, that you can use to calculate the histogram and enhance the contrast of N-D images.

dicomread Supports JPEG Variant

The `dicomread` function now supports JPEG compression, process 2 and 4.

imfilter can return different results for codegen with certain inputs

When using `imfilter`, if you specify a large kernel, a kernel that contains large values, or specify an image containing large values, you can see different results between MATLAB and generated code using codegen for floating point data types. This happens because of accumulation errors due to different algorithm implementations.

Functions Being Removed or Changed

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
<code>imclose</code> and functions that perform morphological closing using <code>imclose</code> , such as <code>imbothat</code> .	Still works	Not applicable	<code>imclose</code> now pads the input image border by half the size of the structuring element. Padding the image removes border artifacts when there are foreground pixels near the boundary of the input image.

R2016b

Version: 9.5

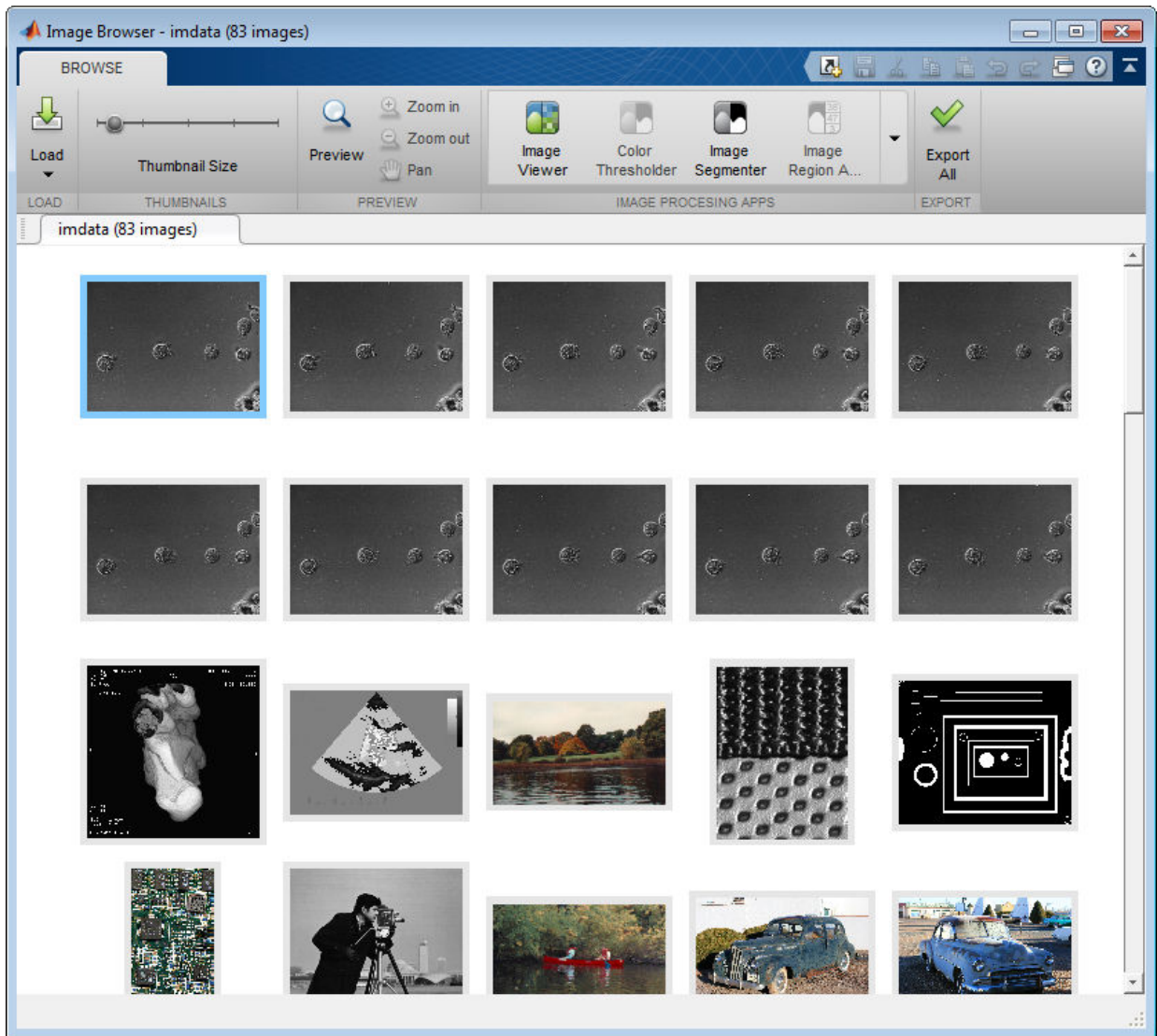
New Features

Bug Fixes

Compatibility Considerations

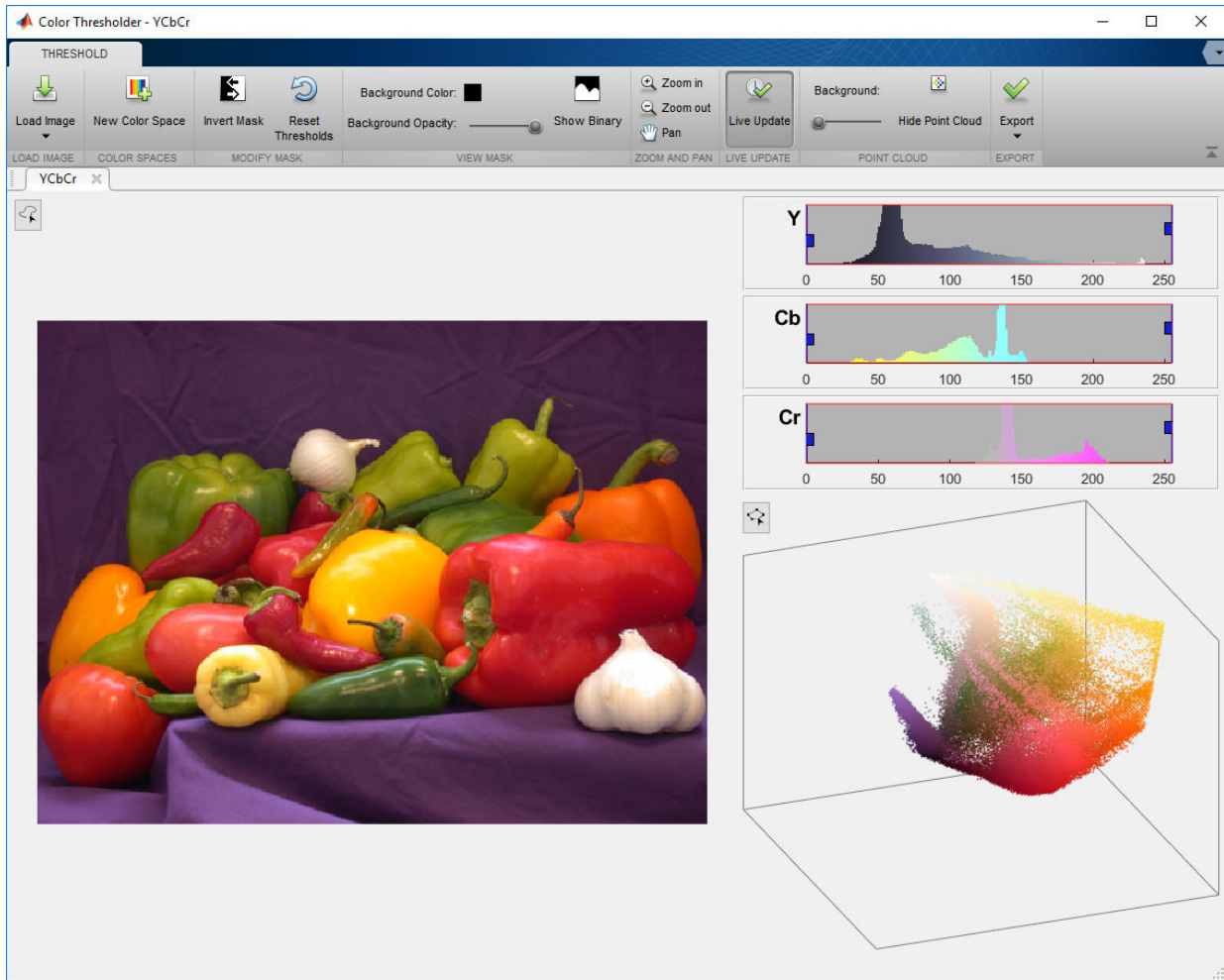
Image Browser App: View multiple images and import selected image into apps

The new Image Browser app lets you view all the images in a folder or an image datastore and lets you import a selected image into an app. For an example, see [View Thumbnails of Images in Folder or Datastore](#).



Color Thresholder App: View color data as point cloud for segmentation

The Color Thresholder app now lets you view image color data as a point cloud in four different color spaces: RGB, HSV, YCbCr, L*a*b*. You can use these point clouds to see which color space provides the clearest separation of color that makes it easy to select particular colors for segmentation. You use the point clouds in conjunction with the existing controls offered for each color space. For an example, see [Image Segmentation Using Point Clouds in the Color Thresholder App](#).



Edge-Aware Filtering: Perform edge-aware filtering based on Laplacian pyramids

The toolbox includes several new functions that provide edge-aware filtering based on Laplacian pyramids. These functions, `locallapfilt`, `localcontrast`, and `localtonemap`, produce high-quality results without introducing halos.



Before and After Images of Local Laplacian Filtering

3-D Superpixels: Use simple linear iterative clustering (SLIC) with volumetric images

The toolbox includes a new function that enables you to perform a simple linear iterative clustering (SLIC) 3-D superpixel segmentation of 3-D volumetric images. Use the `superpixels3` function to obtain the segmentation of the image, using the SLIC superpixel algorithm.

3-D Median Filtering: Apply median filter to volumetric image data

The toolbox includes a new function for 3-D median filtering: `medfilt3`.

colorcloud: Display color information as a point cloud

The toolbox includes the new function, `colorcloud`, that you can use to display color information in an image as a point cloud.

edge: Perform faster Canny edge detection

The `edge` function supports a new parameter, `approx_canny`, that lets you perform Canny edge detection with better performance than the `canny` option, although the `canny` option might provide more precise filtering.

imshow now sets colormap of axes

The `imshow` function now sets the `colormap` property of individual axes, rather than setting the `colormap` of the figure object. This enables graphics objects with different colormaps to be displayed in the same figure. When displaying multiple images in one figure window using `subplot`, each image maintains its own colormap. For an example, see [Display Multiple Images in the Same Figure](#).

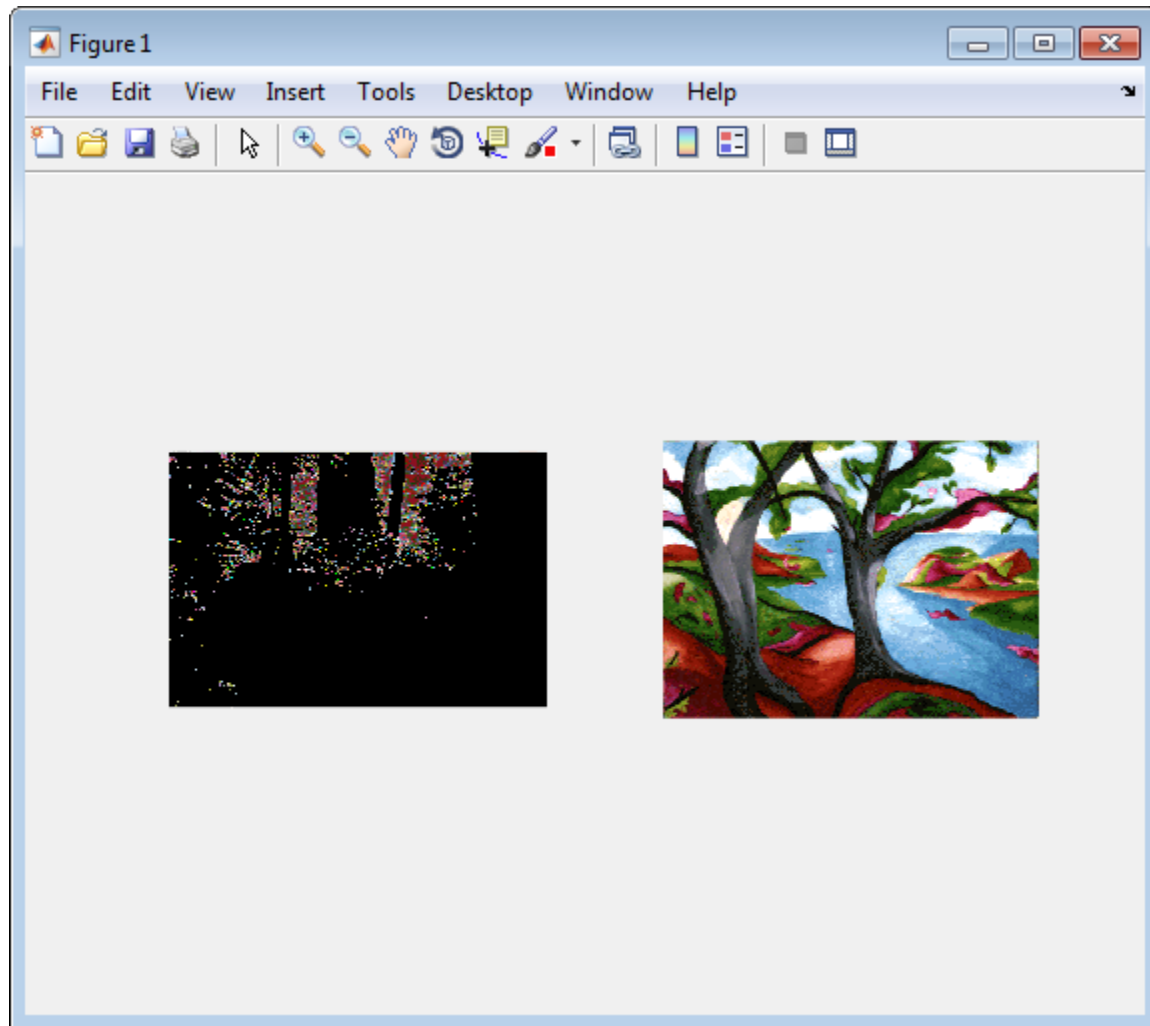
Compatibility Considerations

Certain legacy code may be impacted by the new behavior of `imshow`, including:

- Code that depends on all axes in the figure window having the same colormap as the most recently displayed graphics object.
- Code that gets the colormap of the figure window after using `imshow`.
- Code that sets the colormap of the figure window after using `imshow`.

You can reproduce the prior behavior of `imshow` in subplots by providing each axes in the figure window with the same colormap, determined by the most recent graphics object displayed. For example, you can use this syntax to display two images.

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



Note how the first image displayed, X1, appears dark after applying the colormap of the second image.

nitfread Supports JPEG2000 Compression

The `nitfread` function now supports NITF files that use JPEG2000 compression.

imregdemons Supports 3-D Images on GPU

The `imregdemons` function now supports 3-D images when run on a GPU.

Contrast Adjustment Tool Now Supports Rsets

The Contrast Adjustment tool that is part of the Image Viewer app now supports reduced resolution files (Rsets).

Specify Initial Folder Opened in Open Image Dialog Box

The `imgetfile` function now supports the new 'InitialPath' option that you can use to specify the folder displayed when the Open Image dialog box opens.

Functions Being Removed

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
subimage	Still works	Use <code>imshow</code> instead.	Replace instances of <code>subimage</code> with the <code>imshow</code> function.

R2016a

Version: 9.4

New Features

Bug Fixes

Compatibility Considerations

superpixels: Use simple linear iterative clustering (SLIC) to group pixels for efficient segmentation of color and grayscale images

The toolbox includes several new functions that enable you to perform a simple linear iterative clustering (SLIC) superpixel segmentation of grayscale and color images, and then visualize the segmentation. Use the `superpixels` function to obtain the segmentation of the image, using the SLIC superpixel algorithm. Then, visualize the segmentation using the `boundarymask` function. You can view the segmentation boundaries overlaid on the original image using the `imoverlay` function. You can also obtain lists of the pixels in each segmented region using the `label2idx` function.

Image Segmenter: Segment images using new techniques, including flood-fill and adaptive thresholding

The Image Segmenter app now includes more tools to refine your segmentation, including flood-fill and adaptive thresholding. For an example, see [Image Segmentation Using the Image Segmenter App](#).

Image Batch Processor: Export non-image results using expanded workflows

The Image Batch Processor app now supports more complex workflows. The app can now return other types of information besides a processed image. For an example, see [Batch Processing Using the Image Batch Processor App](#).

imgradient3 and imgradientxyz: Calculate 3-D gradient magnitude, direction, and elevation

The toolbox includes two new functions for computing 3-D image gradients: `imgradient3` and `imgradientxyz`. `imgradient3` computes the gradient magnitude, direction, and elevation. `imgradientxyz` computes the X, Y, and Z directional gradients.

C-code generation: Generate code from 20 additional functions using MATLAB Coder

The following table lists the Image Processing Toolbox functions that have been enabled for code generation in this release. For all target platforms, these functions generate C

code. Some of these functions can also generate C code that uses a precompiled, platform-specific shared library (.dll, .so, or .dylib). Using a shared library preserves performance optimizations in these functions but limits the target to only those platforms that support MATLAB (see system requirements). For a complete list of Image Processing Toolbox functions that support code generation, see List of Supported Functions with Usage Notes.

adaptthresh ¹	imbinarize ¹	imgradientxyz	offsetstrel
boundarymask	imboxfilt ¹	imoverlay	otsuthresh ¹
bwboundaries ¹	imfindcircles ¹	impyramid	rgb2lab
bwconncomp	imgaussfilt ¹	lab2rgb	superpixels
demosaic	imgradient3	label2idx	

¹ If you choose the generic MATLAB Host Computer option in the MATLAB Coder configuration settings, this function generates C code that uses a precompiled, platform-specific shared library.

The function listed in the following table only generates C code that uses a platform-specific shared library. When using this function, choose the generic MATLAB Host Computer option in the MATLAB Coder configuration settings.

imread	
--------	--

The functions listed in the following table previously only generated C code, and can now also generate C code that uses a platform-specific shared library. To use a shared library, choose the generic MATLAB Host Computer option in the MATLAB Coder configuration settings.

rgb2gray	rgb2hsv
----------	---------

In addition, the following function that already supported code generation has additional capabilities.

Function	New Capability
regionprops	Supports the connected components structure for code generation.

imbinarize, otsuthresh, and adapttthresh: Threshold images using global and locally adaptive thresholds

The toolbox includes the new function, `imbinarize`, that converts grayscale images to binary images using global threshold or a locally adaptive threshold. The toolbox includes two new functions, `otsuthresh` and `adapttthresh`, that provide a way to determine the threshold needed to convert a grayscale image into a binary image. This function replaces the `im2bw` function.

Structuring Elements: New shapes and a new function

The `strel` function, which you use to create structuring elements used in morphological operations, supports several new 3-D shapes: 'sphere', 'cube', and 'cuboid'.

In addition, the toolbox now includes a new function, named `offsetstrel`, that you use to create the nonflat structuring elements, 'ball' and 'arbitrary', formerly created using the `strel` function. You now use the `strel` function to create flat structuring elements, such as 'disk' and 'line'.

DICOM function support non-ASCII character sets

The DICOM functions `dicominfo` and `dicomwrite` now support non-ASCII character sets. DICOM files can contain non-ASCII data, such as Kanji, Katakana, and Hiragana characters, in personal name fields and other fields.

edge: Change to Canny Edge Detection

The implementation of the Canny method of edge detection in the `edge` function has been modified to make the gradient computation more accurate.

Compatibility Considerations

Because of this change, the `edge` function returns more accurate results; however, the results returned are different than what they were in previous releases.

Performance improvements

The performance of the following functions has improved:

- `rgb2lab`
- `rgb2xyz`

Color space conversion functions: improved precision and numerical consistency

The following color space conversion functions have improved precision and numerical accuracy:

- `lab2rgb`
- `xyz2rgb`
- `rgb2lab`
- `rgb2xyz`

Compatibility Considerations

Because of the improved precision and numerical consistency of the `lab2rgb`, `xyz2rgb`, `rgb2lab`, `rgb2xyz` functions, the results they return are different than what they returned in previous releases.

Functions Being Removed

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
corner	Still works	Use <code>detectHarrisFeatures</code> or <code>detectMinEigenFeatures</code> in Computer Vision System Toolbox™ instead.	Replace instances of <code>corner</code> with the appropriate replacement function.

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
cornermetric	Still works	Use <code>detectHarrisFeatures</code> or <code>detectMinEigenFeatures</code> and the <code>cornerPoints</code> class in Computer Vision System Toolbox instead.	Replace instances of <code>cornermetric</code> with the appropriate replacement function.
im2bw	Still works	Use <code>imbinarize</code> instead.	Replace instances of <code>im2bw</code> with <code>imbinarize</code> .

R2015b

Version: 9.3

New Features

Bug Fixes

Compatibility Considerations

C-code generation support for more than 20 functions using MATLAB Coder

The following table lists the Image Processing Toolbox functions that have been enabled for code generation in this release. For all target platforms, these functions generate C code. Some of these functions can also generate C code that uses a precompiled, platform-specific shared library (.dll, .so, or .dylib). Using a shared library preserves performance optimizations in these functions but limits the target to only those platforms that support MATLAB (see system requirements). For a complete list of Image Processing Toolbox that support code generation, see List of Supported Functions with Usage Notes.

bwareaopen	houghpeaks	immse	integralBoxFilter
grayconnected ¹	imabsdiff	imresize	psnr
hough	imcrop	imrotate ¹	
houghlines	imgaborfilt	imtranslate ¹	

¹ If you choose the generic MATLAB Host Computer option in the MATLAB Coder configuration settings, this function generates C code that uses a precompiled, platform-specific shared library.

This release also includes functions, listed in the following table, that previously generated C code that uses a platform-specific shared library but now also generate C code that does not require a shared library. For these functions, if you choose the generic MATLAB Host Computer option in the MATLAB Coder configuration settings, these functions generate C code that uses a precompiled, platform-specific shared library.

histeq	im2uint16	imlincomb	rgb2ycbcr
im2int16	imadjust	intlut	ycbcr2rgb

gabor and imgaborfilt: New class and function for designing and applying Gabor filter banks for use in edge and texture analysis

The toolbox includes two new functions, `imgaborfilt` and `gabor`, that provide an easy-to-use interface for Gabor filtering.

grayconnected and imboxfilt: Segment regions by intensities and apply spatial domain filters

The toolbox includes a new function, `grayconnected`, that provides a way to segment regions of similar intensity in a grayscale image. In addition, the toolbox includes several new functions that create an integral image, `integralImage` and `integralImage3`, and that can use the integral image for filtering, `imboxfilt`, `imboxfilt3`, `integralBoxFilter`, and `integralBoxFilter3`.

Performance: Performance improvements for grayscale morphology and image filtering

The performance of several morphological processing functions and image filtering functions has been improved.

dpxread and dpxinfo: Read Digital Picture Exchange files

The toolbox includes two new functions, `dpxread` and `dpxinfo`, that provide an easy-to-use interface for reading DPX files.

imwarp function supports new SmoothEdges parameter

The `imwarp` function supports a new parameter, `SmoothEdges`, that controls whether `imwarp` pads the input image to create a smoother edge in the output image.

Compatibility Considerations

The default value of `SmoothEdges` is `false`, which produces different results than in previous releases. To obtain the same edge behavior as in previous releases, set `SmoothEdges` to `true`.

DICOM functions support new options

The `dicominfo`, `dicomread`, `dicomdisp`, and `addicomanon` functions supports a new parameter, `UseVRHeuristic`. When set to `true` (the default), this parameter instructs the parser to use a heuristic to help read certain noncompliant files which switch value representation (VR) modes incorrectly. The functions issue a warning if they use the heuristic.

The `dicominfo` function supports a new parameter, `UseDictionaryVR`, which specifies whether the data types of returned metadata should conform to the data dictionary, regardless of what information is present in the file.

New example shows use of imaging functions with the Vision HDL Toolbox

There is a new example, “Generate HDL Code for Image Sharpening,” that showcases use of Image Processing Toolbox functions with the Vision HDL Toolbox™.

R2015a

Version: 9.2

New Features

Bug Fixes

Compatibility Considerations

C-code generation support for more than 20 functions, including regionprops, watershed, bweuler, bwlabel, bwperim, and multithresh using MATLAB Coder

The following table lists the Image Processing Toolbox functions that have been enabled for code generation in this release. For all target platforms, these functions generate C code. If you choose the generic MATLAB Host Computer option in the MATLAB Coder configuration settings, many of these functions also generate C code that uses a precompiled, platform-specific shared library (.dll, .so, or .dylib). Using a shared library preserves performance optimizations in these functions but limits the target to only those platforms that support MATLAB (see system requirements). For a complete list of Image Processing Toolbox that support code generation, see List of Supported Functions with Usage Notes.

bweuler ¹	bwperim ¹	watershed ¹
bwlabel	regionprops ¹	

¹ If you choose the generic MATLAB Host Computer option in the MATLAB Coder configuration settings, these functions generate C code that uses a precompiled, platform-specific shared library.

This release also includes functions, listed in the following table, that previously generated C code that uses a platform-specific shared library but now also generate C code that does not require a shared library. You choose which by specifying the platform in MATLAB Coder. If you choose the generic MATLAB Host Computer option in the MATLAB Coder configuration settings, these functions generate C code that uses a precompiled, platform-specific shared library.

bwselect	imclearborder	imfill	imhmin	imregionalm in	multithresh
edge	imextended max	imhist	imreconstru ct	imwarp	ordfilt2
im2uint8	imextended min	imhmax	imregionalm ax	medfilt2	stretchlim

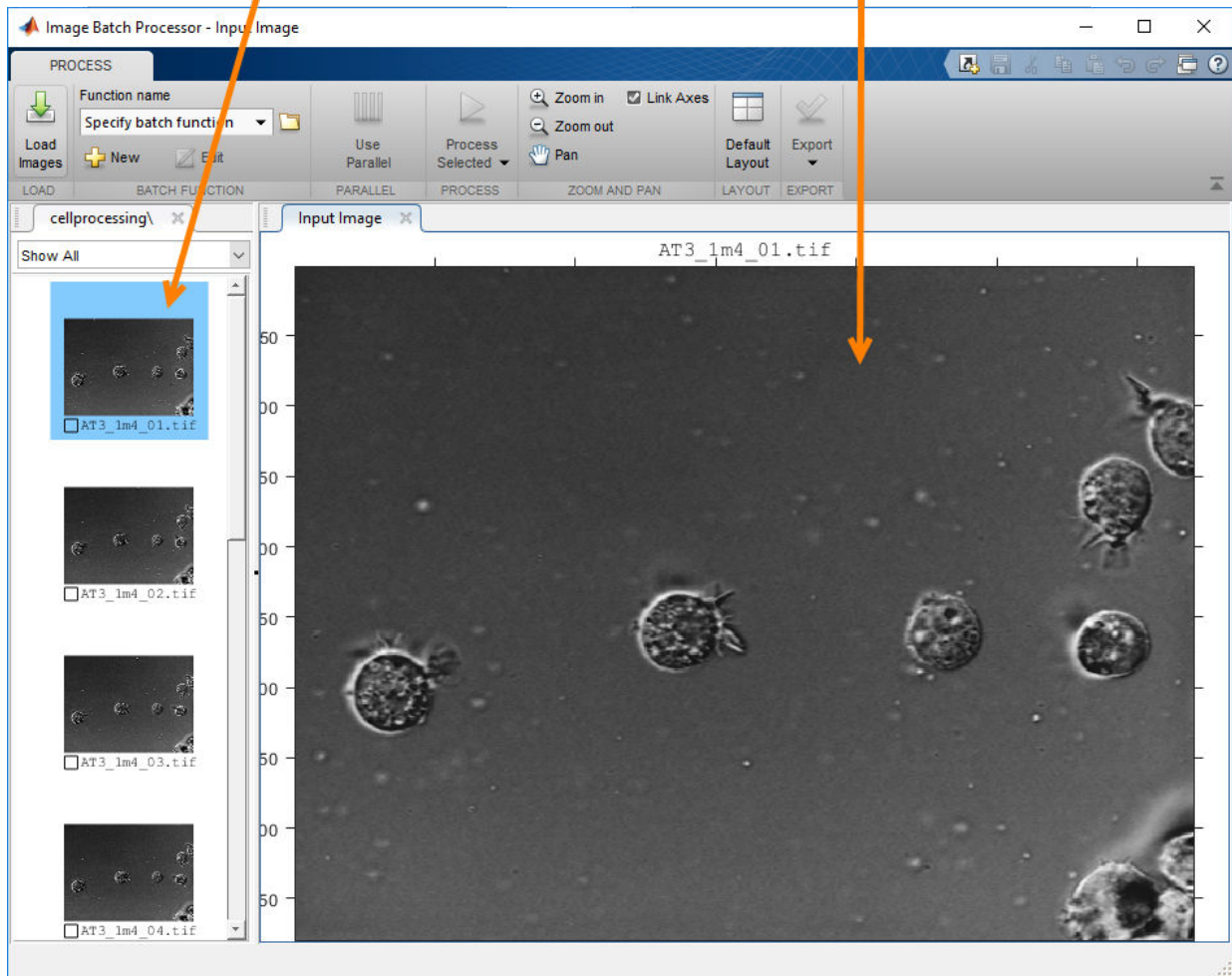
App for batch processing sets of images

The toolbox includes new app that facilitates batch processing called the Batch Processing app, shown below. The Batch Processing app enables you to perform an image

processing operation on a selection of files or the entire contents of a folder. Using the app, you specify the folder and the function that you want executed. For an example, see [Batch Processing Using the Image Batch Processor App](#).

View thumbnails of images in folder.

View selected image in larger resolution.



Fast geodesic interactive segmentation

The toolbox includes a new function, `imseggeodesic`, that provides adaptive, geodesic distance-based binary or trinary segmentation for color images. With this function, you can specify a few pixels, called scribbles, belonging to the different regions (for example, background and foreground in binary segmentation) in the image, and from them the whole image is automatically segmented.

Optimized function for Gaussian filtering

The toolbox includes two new functions, `imgaussfilt` and `imgaussfilt3`, that provide an easy-to-use interface for Gaussian filtering for both 2-D and 3-D data. With these functions, you can switch between filtering in the spatial or frequency domains. In addition, these functions support anisotropic Gaussian filters, enabling you to apply Gaussian smoothing with different standard deviations along each data dimension. Use these functions instead of using `fspecial` and `imfilter` to create a Gaussian kernel and then applying it.

Fill entire region including border pixels

The toolbox includes a new function, `regionfill`, that provides a way to fill a specified region in an image using inward interpolation so that the region blends in with the surrounding background. `regionfill` replaces the `roifill` function. `regionfill` fill in regions in an image including the border pixels. `roifill` left the pixels on the border of the region unprocessed.

Visualize results of boundary tracing

The toolbox includes a new function, `visboundaries`, that provides a way to plot region boundaries on set of axes. You can use this function to visualize the results returned by the `bwboundaries` function.

Examine contents of DICOM files

The toolbox includes a new function, `dicomdisp`, that displays the metadata of a DICOM file at the command prompt. `dicomdisp` can be useful when debugging issues with DICOM files.

Live image capture in Color Thresholder app

You can now do color thresholding on an image acquired from a Webcam using the Color Thresholder app. The new Image Capture tab allows you to bring a live image from USB Webcams into the app. Previously, you had to save your images to disk and manually add them into the app.

The image capture functionality in the Color Thresholder app allows you to:

- Capture live images from USB Webcams
- Save an acquired image to a variable
- Integrate between image acquisition and color thresholding
- Control camera properties, such as brightness and contrast

Use the `colorThresholder` function to open the app. Then select **Load Image > Acquire Image From Camera** to open the Image Capture tab. Select your device, set any properties, and preview the image. You can then capture an image.

regionprops function can return results in table

The `regionprops` function can now return results in a table. To use this capability, specify the string 'table' as the first input argument.

GPU acceleration for imregionalmax, imregionalmin, imgaussfilt, imgaussfilt3, and regionprops functions

This release adds GPU acceleration for several toolbox functions: `imgaussfilt`, `imgaussfilt3`, `imregionalmax`, `imregionalmin`, and `regionprops`. GPU acceleration for these functions requires Parallel Computing Toolbox software and meeting the GPU computing requirements detailed here.

Functions Being Removed

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
roifill	Still works but issues a warning	regionfill	Replace instances of roifill with regionfill.

R2014b

Version: 9.1

New Features

Bug Fixes

Apps for image segmentation and region analysis

The toolbox includes new apps:

- Image Segmenter app
- Image Region Analyzer app

Image Segmenter app

The Image Segmenter app enables you to segment images using the active contours (snakes) algorithm. In this app, you first initialize the segmentation by specifying a rough segmentation or initial condition. When you click **Evolve**, the app evolves the initial segmentation, performing the number of iterations you specify, creating a binary mask image. For an example, see Image Segmentation Using the Image Segmenter App.

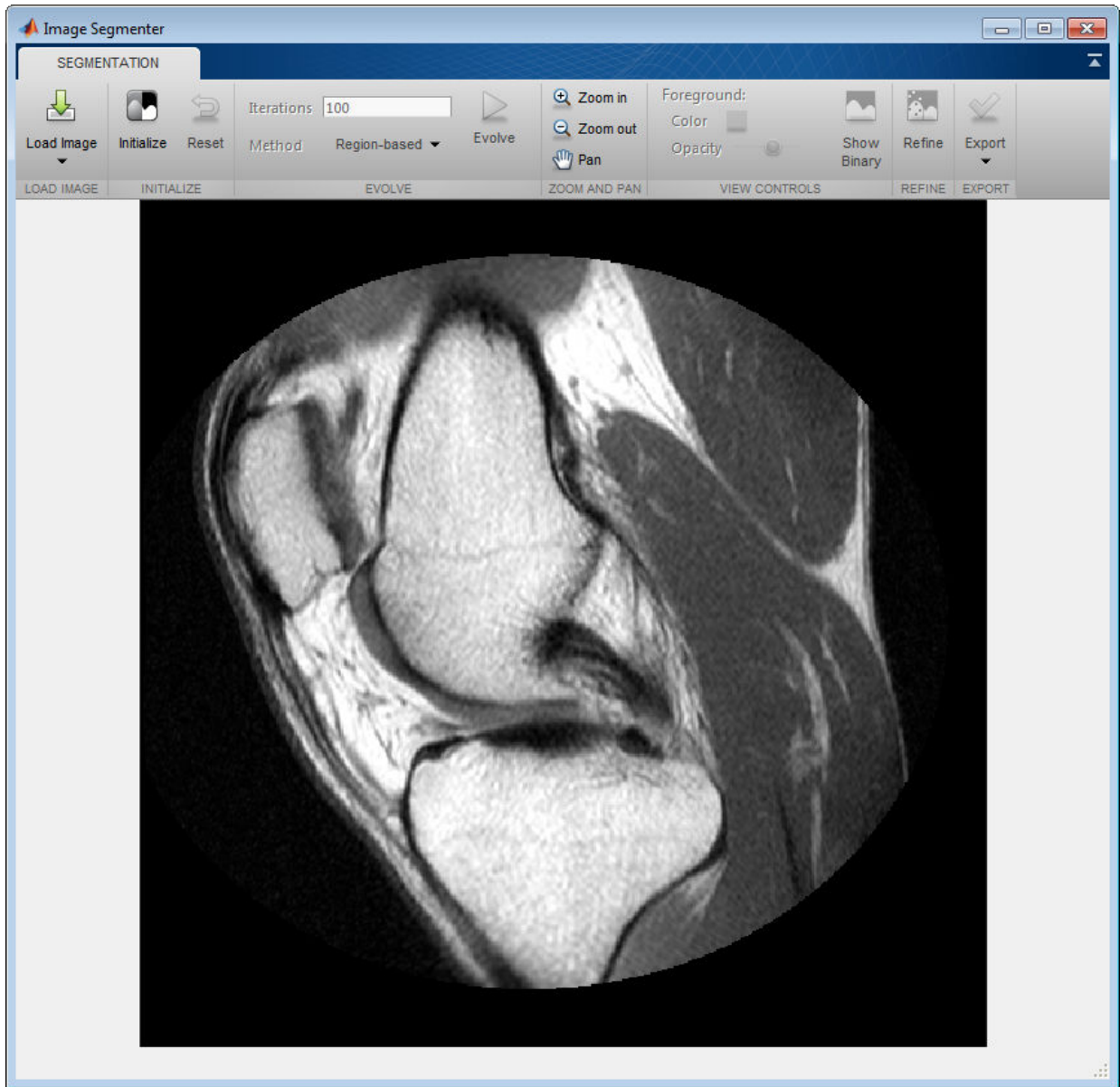
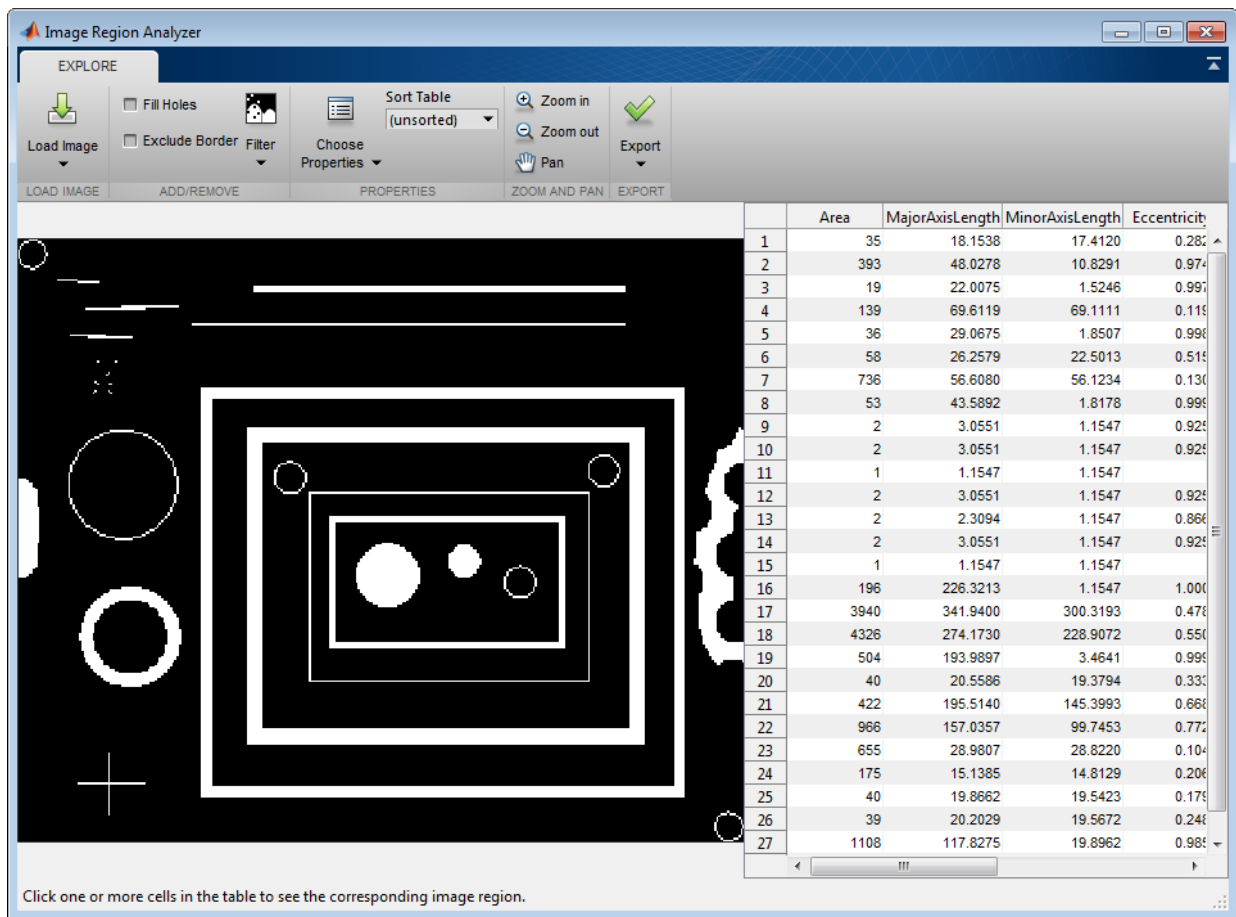


Image Region Analyzer app

The Image Region Analyzer app enables you to explore binary images and filter images based on the properties of regions in the image. For example, you can filter an image to remove all objects smaller than a particular size. The following figure shows the Image Region Analyzer app. When you select the value of a property in the table, the app highlights the corresponding region in the image. For examples, see Calculate Region Properties Using Image Region Analyzer and Filter Images on Region Properties Using Image Region Analyzer App.



C-code generation support for 16 additional functions using MATLAB Coder, including bwtraceboundary, imadjust, imclearborder, and medfilt2

This release includes 16 additional toolbox functions that support the generation of C code using MATLAB Coder. Some Image Processing Toolbox functions generate C code that depends on a platform-specific shared library (.dll, .so, or .dylib). Using a shared library preserves performance optimizations in these functions but limits the target to only those platforms that support MATLAB (see system requirements).

The following table lists the Image Processing Toolbox functions that have been enabled for code generation in this release. The table identifies functions that can also generate code that uses a shared library. For a complete list of Image Processing Toolbox that support code generation, see List of Supported Functions with Usage Notes.

<code>bwdist</code> ¹	<code>imadjust</code> ¹	<code>intlut</code> ¹	<code>ordfilt2</code> ¹	<code>ycbcr2rgb</code> ¹
<code>bwtraceboundary</code>	<code>imclearborder</code> ¹	<code>iptcheckmap</code>	<code>rgb2gray</code> ¹	
<code>fitgeotrans</code>	<code>imlincomb</code> ¹	<code>medfilt2</code> ¹	<code>rgb2ycbcr</code> ¹	
<code>histeq</code> ¹	<code>imquantize</code>	<code>multithresh</code> ¹	<code>stretchlim</code> ¹	

¹ Generated code uses a precompiled, platform-specific shared library.

This release also includes nine functions, listed in the following table, that generate C code or generate C code that uses a platform-specific shared library, depending on which platform you specify in MATLAB Coder. If you choose the generic MATLAB Host Computer option in the MATLAB Coder configuration settings, these functions generate C code that uses a precompiled, platform-specific shared library.

<code>bwlookup</code>	<code>imclose</code>	<code>imfilter</code>
<code>bwmorph</code>	<code>imdilate</code>	<code>imopen</code>
<code>imbothat</code>	<code>imerode</code>	<code>imtophat</code>

Nonrigid image registration

The toolbox includes a new function, `imregdemons`, that provides a way to perform nonrigid image registration. The toolbox already supports several types of rigid image

registration that work at a global level, applying the same mathematical transformation to every pixel in an image. Now the toolbox supports a function that works at a local level, capable of applying different transformations to every image pixel. The `imregdemons` function returns the warped image and a displacement field that describes how each pixel is transformed.

Image warping using displacement fields

The `imwarp` function now accepts the displacement field returned by the `imregdemons` as input. The `imregdemons` function returns a warped image and a displacement field that describes how each pixel is transformed. If you want more detailed control of a nonrigid registration, you can pass this displacement field to `imwarp` to perform the transformation. For example, by using `imwarp`, you can specify the interpolation method used.

Image segmentation using the Fast Marching Method algorithm

The toolbox includes a new function, `imsegfmm`, that segments an image using the Fast Marching Method (FMM) algorithm. To segment an image, you must first create a weight image using either the `gradientweight` or `graydiffweight` functions. These functions create an image in which every pixel is a value. You then pass this weight image to `imsegfmm`, specifying where the algorithm should start, the seed location.

Image comparison using mean-squared error

The toolbox includes a new function, `immse`, that calculates the mean-squared error.

Image filtering based on object properties

The toolbox includes two new functions, `bwareafilt` and `bwpropfilt` that filter images based on the values of image properties. For example, using `bwareafilt`, you can remove objects from an image if their area is less than a specified number of pixels.

Color space conversion functions

The toolbox includes several new functions to convert between the RGB, XYZ, and L^*a^*b color spaces.

rgb2lab	rgb2xyz	lab2xyz
lab2rgb	xyz2rgb	xyz2lab

Use these new conversion functions instead of using `makecform` with these conversion types: `lab2srgb`, `srgb2lab`, `srgb2xyz`, `xyz2srgb`, `lab2xyz`, and `xyz2lab`.

activecontour function supports parameter to control tendency of contour to expand or contract

The `activecontour` function supports a new parameter, `ContractionBias`, that influences whether the contour grows outward or shrinks inward during segmentation.

Region-of-Interest (ROI) functions now support deletion from context menu

The `imellipse`, `imfreehand`, `imline`, `impoint`, `impoly`, and `imrect` functions, that you use to define regions of interest in images, now support a deletion option from their context menus.

dicomwrite function now supports the ability to specify the bitdepth of images written

The `dicomwrite` function now supports the `UseMetadataBitDepths` parameter which you can use to specify the bitdepth of the image written to the DICOM file.

GPU acceleration for bwlabel and imregdemons

This release adds GPU acceleration for two toolbox functions: `bwlabel` and `imregdemons`. GPU acceleration for these functions requires Parallel Computing Toolbox software and meeting the GPU computing requirements detailed here.

R2014a

Version: 9.0

New Features

Bug Fixes

Compatibility Considerations

C-code generation for more than 25 functions, including edge, imfilter, imwarp, imopen, imclose, imerode, and imdilate using MATLAB Coder

You can generate standalone C code for a group of toolbox functions, listed below. Generating code requires MATLAB Coder.

affine2d	im2double	imclose	imhist	mean2
bwpack	im2int16	imdilate	imopen	projective2d
bwselect	im2single	imerode	imref2d	strel
bwunpack	im2uint16	imextendedmax	imref3d	
edge	im2uint8	imextendedmin	imtophat	
getrangefromclass	imbothat	imfilter	imwarp	

GPU acceleration for an additional nine functions, including bwdist, imfill, imreconstruct, iradon, radon, and stretchlim

This release adds GPU acceleration for an additional nine toolbox functions, listed below. GPU acceleration for these functions requires Parallel Computing Toolbox software and meeting the GPU computing requirements detailed here.

bwdist	imreconstruct	normxcorr2
imcomplement	iradon	radon
imfill	mean2	stretchlim

App for color image thresholding

The toolbox includes a new Color Thresholder app that enables you to segment color images by manipulating their color components. The app presents the image using several standard color spaces. You choose which color space representation gives the best contrast between foreground and background and then use the component histograms to segment the image.

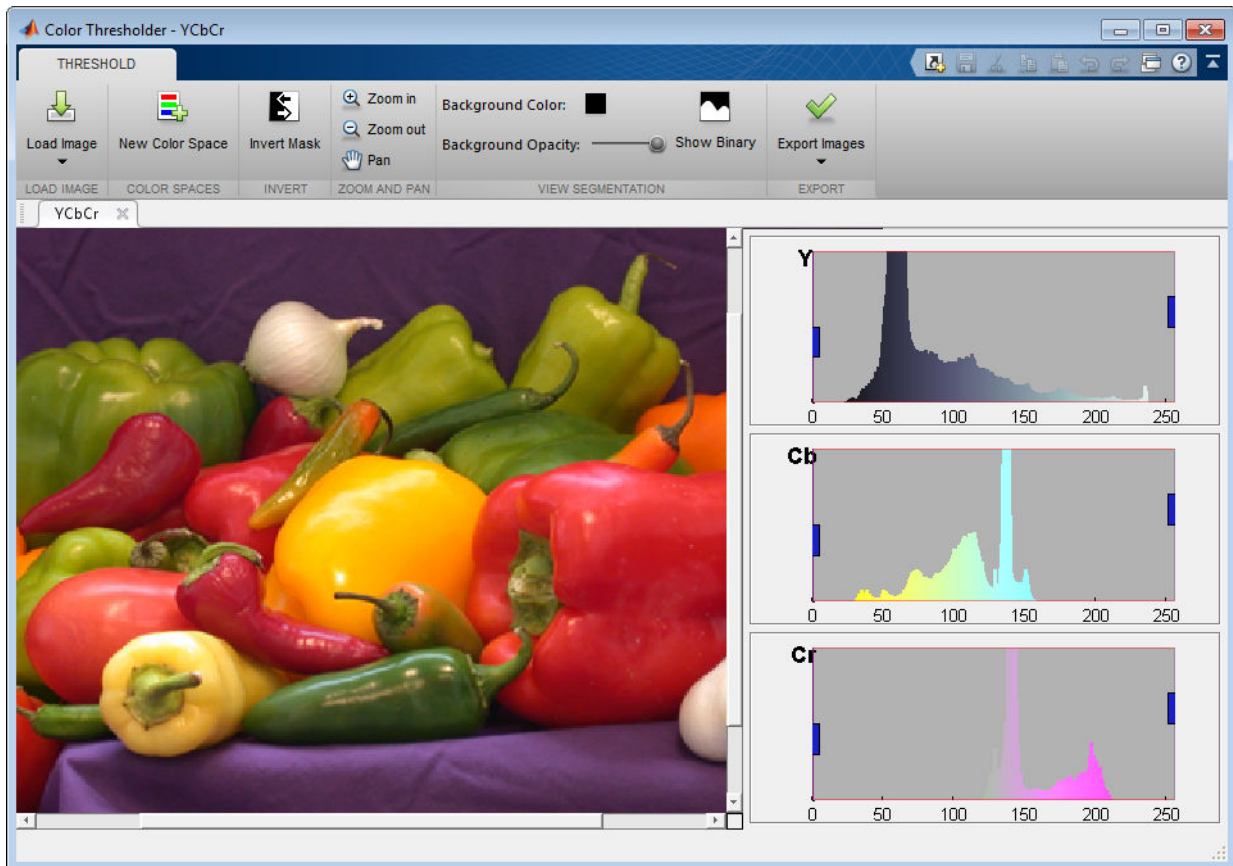


Image quality metrics, including peak signal to noise (psnr) and structured similarity metric (ssim)

The toolbox includes two new functions, `ssim` and `psnr`, that compute image quality metrics.

Guided filtering for image enhancement

The toolbox includes a new function, `imguidedfilter`, that provides an edge-preserving nonlinear filter for use with images.

Phase correlation and translation-only image registration functions

The toolbox includes a new function, `imregcorr`, that applies an FFT-based transform to register two images with regard to translation, rotation, and scale. The toolbox includes a new function, `imtranslate`, that applies a translation transformation to an input image and returns the transformed image. Using `imtranslate`, you can provide spatial reference information, specify the method used for interpolation, and control other aspects of the translation.

File names of Image Processing Toolbox examples changed

The file names of Image Processing Toolbox examples have changed. The following table lists the example files in alphabetical order by their former names.

Old Example File Name	New Example File Name
<code>ipexaerial.m</code>	<code>RegisterAerialPhotoExample.m</code>
<code>ipexangle.m</code>	<code>MeasureAngleExample.m</code>
<code>ipexautorotate.m</code>	<code>RotationFeatureMatchingExample.m</code>
<code>ipexbatch.m</code>	<code>BatchProcessImageExample.m</code>
<code>ipexblind.m</code>	<code>BlindImageDeblurringExample.m</code>
<code>ipexblockprocedge.m</code>	<code>BlockProcessLargeImageExample.m</code>
<code>ipexblockprocstats.m</code>	<code>BlockProcessStatisticsExample.m</code>
<code>ipexcell.m</code>	<code>CellSegmentationExample.m</code>
<code>ipexcheckerboard.m</code>	<code>GalleryTransformedImagesExample.m</code>
<code>ipexcircles.m</code>	<code>DetectCirclesExample.m</code>
<code>ipexconformal.m</code>	<code>ConformalMappingImageExample.m</code>
<code>ipexcontrast.m</code>	<code>ContrastEnhancementExample.m</code>
<code>ipexfabric.m</code>	<code>LabColorSegmentationExample.m</code>
<code>ipexhistology.m</code>	<code>KMeansSegmentationExample.m</code>
<code>ipexknee.m</code>	<code>RegisterMultimodalImagesExample.m</code>
<code>ipexlanstretch.m</code>	<code>MultispectralImageEnhancementExample.m</code>

Old Example File Name	New Example File Name
ipexlucy.m	LucyRichardsonImageDeblurringExample.m
ipexmri.m	MRISliceExample.m
ipexndvi.m	MultispectralVegetationSegmentationExample.m
ipexnormxcorr2.m	RegisterNormalizedCrossCorrelationExample.m
ipexpendulum.m	PendulumLengthExample.m
ipexprops.m	MeasureGrayscaleRegionsExample.m
ipexradius.m	MeasureRadiusExample.m
ipexreconstruct.m	ReconstructImageExample.m
ipexregularized.m	RegularizedImageDeblurringExample.m
ipexrice.m	NonuniformIlluminationExample.m
ipexrotate.m	RotationFitgeotransExample.m
ipexroundness.m	RoundObjectsExample.m
ipexshear.m	PadShearImageExample.m
ipexsnow.m	SnowflakesGranulometryExample.m
ipextexturefilter.m	TextureSegmentationExample.m
ipextraffic.m	TrafficSegmentationExample.m
ipexwatershed.m	WatershedSegmentationExample.m
ipexwiener.m	WienerImageDeblurringExample.m

Location of sample images changed

The location of Image Processing Toolbox sample images has changed. These images were stored in the `imdemos` folder and are now stored in the `imdata` folder.

regionprops function uses new algorithm to calculate perimeter

The `regionprops` function uses a new algorithm to calculate a perimeter, when used with the 'Perimeter' option. Because of this change, `regionprops` returns different results for the perimeter calculations than it did in earlier releases.

Compatibility Considerations

While the new algorithm used with `regionprops` returns more accurate perimeter calculation, you can get the same return value as previous releases by specifying the 'perimeterold' option.

R2013b

Version: 8.3

New Features

Bug Fixes

Compatibility Considerations

GPU acceleration for more than 20 functions, including `bwmorph`, `edge`, `imresize`, and `medfilt2`

This release introduces GPU acceleration for a group of toolbox functions, listed below. GPU acceleration for these functions requires Parallel Computing Toolbox software and meeting the GPU computing requirements detailed here.

<code>bwmorph</code>	<code>im2single</code>	<code>imgradientxy</code>	<code>medfilt2</code>
<code>corr2</code>	<code>im2uint8</code>	<code>imhist</code>	<code>rgb2gray</code>
<code>edge</code>	<code>im2uint16</code>	<code>imlincomb</code>	<code>rgb2ycbcr</code>
<code>histeq</code>	<code>imabsdiff</code>	<code>imnoise</code>	<code>std2</code>
<code>im2double</code>	<code>imadjust</code>	<code>imresize</code>	<code>stdfilt</code>
<code>im2int16</code>	<code>imgradient</code>	<code>mat2gray</code>	<code>ycbcr2rgb</code>

Additional 2D geometric transformations: `piecewise linear`, `local weighted mean`, and `polynomial`

The toolbox includes several new classes for representing 2D geometric transformations, listed below.

<code>images.geotrans.PiecewiseLinearTransformation2d</code>	2D piecewise linear geometric transformation
<code>images.geotrans.PolynomialTransformation2d</code>	2D polynomial geometric transformation
<code>images.geotrans.LocalWeightedMeanTransformation2d</code>	2D local weighted mean geometric transformation

Additional parameter in `imregister` and `imregtform` to specify initial transformation

The `imregister` and `imregtform` functions both support a new parameter, `InitialTransformation`, that enables you to specify the starting point for the transformation. Specifying an initial geometric transformation as a starting point, the functions can return better results.

fitgeotrans function for fitting geometric transformation to control point pairs

The toolbox includes a new function, `fitgeotrans`, that takes pairs of control points and uses them to infer a geometric transformation. The function returns a geometric transformation object that can be used with `imwarp`.

imregister and imregtform Return Different Values

Some changes to the implementation of the `imregister` and `imregtform` functions might cause these functions to return different results than they did before R2013b.

Compatibility Considerations

You might need to change parameters to the `imregister` and `imregtform` functions to achieve similar results to what you had previously.

Functions Being Removed

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
<code>cp2tform</code>	Still works but issues a warning	<code>fitgeotrans</code>	Replace instances of <code>cp2tform</code> with <code>fitgeotrans</code> .
<code>imtransform</code>	Still works but issues a warning	<code>imwarp</code>	Replace instances of <code>imtransform</code> with <code>imwarp</code> .
<code>maketform</code>	Still works but issues a warning	<code>fitgeotrans</code> , <code>affine2d</code> , <code>affine3d</code> , or <code>projective2d</code>	Replace instances of <code>maketform</code> with one of the recommended functions.

R2013a

Version: 8.2

New Features

Bug Fixes

Image segmentation using active contours

The toolbox includes a new function, `activecontour`, for segmenting an image using the active contours (snakes) algorithm.

Classes and functions for representing and applying 2-D and 3-D geometric transformations

The toolbox includes several new classes for representing 2-D and 3-D geometric transformations, listed below.

<code>affine2d</code>	2-D affine geometric transformation
<code>affine3d</code>	3-D affine geometric transformation
<code>projective2d</code>	2-D projective geometric transformation
<code>imregtform</code>	Estimate the geometric transformation that aligns two images
<code>imwarp</code>	Apply geometric transformation to image

Classes for defining world coordinate system of an image

The toolbox includes several new classes for representing world coordinate systems associated with an image, listed below.

<code>imref2d</code>	Reference 2-D image to world coordinates
<code>imref3d</code>	Reference 3-D image to world coordinates

Code generation for `conndef`, `imcomplement`, `imfill`, `imhmax`, `imhmin`, `imreconstruct`, `imregionalmax`, `imregionalmin`, `iptcheckconn`, and `padarray` functions (using MATLAB Coder)

You can generate standalone C code for a group of toolbox functions, listed below. Generating code requires MATLAB Coder.

<code>conndef</code>	<code>imcomplement</code>
<code>imfill</code>	<code>imhmax</code>

<code>imhmin</code>	<code>imreconstruct</code>
<code>imregionalmax</code>	<code>imregionalmin</code>
<code>iptcheckconn</code>	<code>padarray</code>

GPU acceleration for `imrotate`, `imfilter`, `imdilate`, `imerode`, `imopen`, `imclose`, `imtophat`, `imbothat`, `imshow`, `padarray`, and `bwlookup` functions (using Parallel Computing Toolbox)

This release introduces GPU acceleration for a group of toolbox functions, listed below. GPU acceleration for these functions requires Parallel Computing Toolbox software and meeting the GPU computing requirements detailed here.

<code>bwlookup</code>	<code>imbothat</code>
<code>imclose</code>	<code>imdilate</code>
<code>imerode</code>	<code>imfilter</code>
<code>imopen</code>	<code>imrotate</code>
<code>imshow</code>	<code>imtophat</code>
<code>padarray</code>	<code>std2</code>

Unsharp mask filtering

The toolbox includes a new function, `imsharpen`, that does unsharp mask filtering. Use this new function instead of using the 'unsharp' option with the `fspecial` function.

R2012b

Version: 8.1

New Features

Bug Fixes

Compatibility Considerations

Image gradient computation with `imgradient` and `imgradientxy` functions

The toolbox includes two new functions for computing image gradients: `imgradient` and `imgradientxy`. `imgradient` computes the gradient magnitude and direction. `imgradientxy` computes the X and Y directional gradients

Histogram matching with `imhistmatch` function

The new function `imhistmatch` adjusts the histogram of an image to match the N-bin histogram of a reference image.

Multilevel thresholding with `multithresh` and `imquantize` functions

The new `imquantize` and `multithresh` functions enable multilevel grayscale thresholding and labeling. `imquantize` labels an image using a fixed range of grayscale levels. `multithresh` computes threshold levels using Otsu's method. The levels computed by `multithresh` can be used as input to `imquantize`.

3-D image registration with `imregister` function

In addition to 2-D images, `imregister` can now align 3-D images using intensity-based registration of the images' voxel data.

Code generation for `bwmorph` and `bwlookup` with MATLAB Coder

Standalone C code can be generated for `bwmorph` and `bwlookup`. The generated C code meets the strict memory and data type requirements of embedded target environments. To generate this code you need a MATLAB Coder license.

Added function `bwlookup`

Compatibility Considerations

Function `applylut` is not recommended; use `bwlookup` instead.

Writing private metadata when anonymizing DICOM files

The `dicomanon` function now supports writing private metadata fields using the `'WritePrivate'` parameter

Expanded color options with `imshowpair`

`imshowpair` now supports additional color options for displaying image differences and stereo imagery when using the new `'ColorChannels'` parameter. Using the `'red-cyan'` value with this parameter is particularly useful for viewing stereo anaglyphs.

Performance improvements

The performance of the following functions has improved by taking advantage of hardware optimizations and multicore capabilities:

- `adapthisteq`
- `histeq`
- `imrotate`
- `intlut`

The `dicomanon`, `dicominfo`, `dicomlookup`, `dicomread`, `dicomwrite` functions have optimized implementations.

New Example

Detect and measure circular objects in an image (`ipexcircles`).

R2012a

Version: 8.0

New Features

Bug Fixes

Compatibility Considerations

Intensity-Based Image Registration

The new `imregister` function lets you automatically align two images using intensity values, even when the images were created by two different devices (multimodal). With intensity-based registration, you do not need to specify control points.

You use the new `imregconfig` function to create the optimizer and the metric that `imregister` uses to specify the desired registration parameters.

Two New Functions to Visually Compare Images

The toolbox includes two new functions for visually comparing images: `imshowpair` and `imfuse`.

`imshowpair` creates a composite of two images and displays them in a figure.

`imfuse` creates a composite of two images and returns a third image that is a numeric matrix containing a fused version of the original images.

Circle Detection Using the Circular Hough Transform

The new `imfindcircles` function uses the Hough transform to find circular elements in grayscale, RGB, or binary images. To view the circles that have been detected, overlaid on the original image, use the `viscircles` function.

Performance Improvements

The performance of the `imlincomb` function has improved by taking advantage of multicore capabilities.

New and Updated Demos

The toolbox includes these new and updated demos.

- Registering Multimodal MRI Images (`ipexknee`)
- Finding the Rotation and Scale of a Distorted Image (`ipexrotate`)
- Measuring the Radius of a Roll of Tape (`ipexradius`) - Updated to use the new `imfindcircles` function

Data Type Change to Output Variable for `bwdist`

The `bwdist` function returns two output variables: the Euclidean distance transform and the closest-pixel map. The data type of the second output variable, the closest-pixel map, has changed.

Compatibility Considerations

Before R2012a, the data type of the second output variable returned by the `bwdist` function was `single`. Now the data type of the second output variable returned by `bwdist` is dynamically chosen.

`iradon` Function Updated

A small imprecision in the way the `dc` level was computed in `iradon` has been corrected.

Compatibility Considerations

Output variables `I` and `H` will be slightly different (about 1% to 2%) than in previous versions.

Functions Being Removed

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
<code>iptcheckinput</code>	Still works but issues a warning	<code>validateattributes</code>	Replace all existing instances of <code>iptcheckinput</code> with <code>validateattributes</code> .
<code>iptcheckstrs</code>	Still works but issues a warning	<code>validatestring</code>	Replace all existing instances of <code>iptcheckstrs</code> with <code>validatestring</code> .

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
<code>iptchecknargin</code>	Still works but issues a warning	<code>narginchk</code>	Replace all existing instances of <code>iptchecknargin</code> with <code>narginchk</code> .

R2011b

Version: 7.3

New Features

Bug Fixes

Compatibility Considerations

Parallel Block Processing Now Possible with `blockproc`

If you have Parallel Computing Toolbox, you can now use a new option in `blockproc` to improve performance of block processing tasks. Set the `'UseParallel'` argument to `true` to use this option.

New `bwdistgeodesic` Function Computes Geodesic Distance Transform

Use `bwdistgeodesic` to compute the geodesic distance transform for a binary image.

New `graydist` Function Computes Gray-Weighted Distance Transform

Use `graydist` to compute the gray-weighted distance transform of a grayscale image.

New `imapplymatrix` Function Computes Linear Combination of Color Channels

The new `imapplymatrix` function applies a weighted sum to the color planes of an image for use in color space conversions.

Performance Improvements

Faster Functions

- `imhist` for large images
- `rgb2gray`

`hdrread` Now Corrects for Small Values

The `hdrread` function now returns correct small values. Specifically, the special case value of `(0,0,0,0)` now maps to `(0,0,0)`.

Compatibility Considerations

Before R2011b, the value (0,0,0,0) mapped to (0.5740E-41, 0.5740E-41, 0.5740E-41).

Change in Behavior for `dicomwrite`

If you use the `dicomwrite` function with the `'CreateMode'` option set to `'Create'` and pass in a data structure that contains the `'InversionTime'` tag, you will always receive `'InversionTime'` in your output.

Compatibility Considerations

Before R2011b, if your input structure contained the `'InversionTime'` field, the DICOM file may or may not contain the `'InversionTime'` field. The inclusion of `'InversionTime'` depended on other parameters.

Warning and Error ID Changes

Many warning and error IDs have changed from their previous versions. These warnings or errors typically appear during a function call.

Compatibility Considerations

If using warning or error IDs, you might need to change the strings you use. For example, if you turned off a warning for a certain ID, the warning might now appear under a different ID. If you use a `try/catch` statement in your code, replace the old identifier with the new identifier. There is no definitive list of the differences, or of the IDs that changed.

Functions and Function Elements Being Removed

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
edge function — K-direction syntax	Errors	<code>BW = edge(..., direction)</code>	The syntax <code>BW = edge(...,K)</code> has been removed. Use the <code>BW = edge(...,direction)</code> syntax instead.
edge function — Marr-Hildreth syntax	Errors	<code>edge(I, 'marr-hildreth', ...)</code>	The syntax <code>edge(I, 'marr-hildreth', ...)</code> has been removed. Use the <code>edge(I, 'log', ...)</code> syntax instead.
<code>imfeature</code>	Errors	<code>regionprops</code>	<code>imfeature</code> has been removed. Use <code>regionprops</code> instead.
<code>immovie</code> — <code>immovie(D,size)</code>	Errors	<code>immovie(X,map)</code>	<code>immovie(D,size)</code> is an obsolete syntax and is no longer supported. Use <code>immovie(X,map)</code> instead.
<code>imrotate</code> function — <code>[R,G,B]</code> output syntax	Errors	<code>RGB2 = imrotate(RGB1)</code>	The syntax <code>[R,G,B] = imrotate(RGB)</code> has been removed. Use the <code>RGB2 = imrotate(RGB1)</code> syntax instead.

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
imrotate — no output argument syntax	Starting in R2011b, no output argument syntax returns result in ans.	Call imshow to display the output of imrotate	Replacement is to call imshow to display the output of imrotate, like this: B = imrotate(A,30); imshow(B)
imshow function — imshow(..., DISPLAY_OPTION)	Errors	imshow(..., 'InitialMagnification',100) or imshow(..., 'InitialMagnification','fit')	<ul style="list-style-type: none"> The syntax imshow(..., 'truesize') has been removed. Use the imshow(..., 'InitialMagnification',100) syntax instead. The syntax imshow(..., 'notruesize') has been removed. Use the imshow(..., 'InitialMagnification','fit') syntax instead.
imshow function — imshow(x,y,...) syntax	Errors	imshow(..., 'XData', x, 'YData', y)	The syntax imshow(x,y,...) has been removed. Use the imshow(..., 'XData', x, 'YData', y) syntax instead.

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
<code>imshow</code> function — <code>imshow(I,N)</code> syntax	Errors	Not applicable	The syntax <code>imshow(I,N)</code> has been removed. Your grayscale image will be displayed using 256 shades of gray.
<code>imview</code>	Errors	<code>imshow</code>	<code>imview</code> has been removed. Use <code>imshow</code> instead.
<code>iptsetpref</code> function — 'ImshowTruesize' preference	Errors	'ImshowInitialMagnification' preference	Replace all existing instances of 'ImshowTruesize' with 'ImshowInitialMagnification'.
<code>iptsetpref</code> function — 'ImviewInitialMagnification' preference	Errors	'ImtoolInitialMagnification' preference	Replace all existing instances of 'ImviewInitialMagnification' with 'ImtoolInitialMagnification'.
Any calls to <code>iptsetpref</code> from within a <code>startup.m</code> file	It will do nothing.	Any calls to <code>iptsetpref</code> in any session are persistent.	You can safely remove calls to <code>iptsetpref</code> from <code>startup.m</code> files.
<code>isbw</code>	Errors	Not applicable	<code>isbw</code> has been removed. No replacement.
<code>isgray</code>	Errors	Not applicable	<code>isgray</code> has been removed. No replacement.

Functionality	What Happens When You Use This Functionality?	Use This Instead	Compatibility Considerations
<code>isind</code>	Errors	Not applicable	<code>isind</code> has been removed. No replacement.
<code>isrgb</code>	Errors	Not applicable	<code>isrgb</code> has been removed. No replacement.
<code>medfilt2</code> function — <code>medfilt2(A, [M N], [Mb Nb], ...)</code> syntax	Errors	Not applicable	No replacement.
<code>montage</code> — <code>montage(D, [M N P])</code> syntax	Errors	Not applicable	Syntax has been removed. No replacement.
<code>uintlut</code>	Errors	<code>intlut</code>	<code>uintlut</code> has been removed. Use <code>intlut</code> instead.
<code>wiener2</code> function — <code>wiener2(I, [m n], [mblock nblock], ...)</code> syntax	Errors	<code>wiener2(I, [m n])</code> or <code>wiener2(I, [m n], noise)</code>	The syntax <code>wiener2(I, [m n], [mblock nblock])</code> has been removed. Use the <code>wiener2(I, [m n])</code> syntax instead. The syntax <code>wiener2(I, [m n], [mblock nblock], noise)</code> has been removed. Use the <code>wiener2(I, [m n], noise)</code> syntax instead.

R2011a

Version: 7.2

New Features

Bug Fixes

Compatibility Considerations

New bwconvhull Function Computes Convex Hull Image

Use `bwconvhull` to compute the convex hull image from a binary image.

New dicomwrite Option Writes Multiframe Imagery to Single File

Set the new 'MultiframeSingleFile' option of the `dicomwrite` function to `true` to write multiframe imagery to one file, regardless of how many frames the input image contains.

nitfread Now Reads NITF Files with JPEG-Compressed Images

If you have NITF files containing JPEG-compressed images, you can now read them using `nitfread`.

Reduced Memory Use for std2

The `std2` function has improved performance for large 1-, 8-, and 16-bit integers.

Compatibility Considerations

Compared to releases before R2011a, the `std2` function now returns slightly different results for some images. To receive the same results as previously, use this code:

```
% For input image im
if ~isa(im,'double')
    im = double(im);
end
std_old = std(im(:));
```

Reduced Memory Use for watershed

The `watershed` function is now more memory efficient than it was in releases before R2011a.

Compatibility Considerations

The watershed regions in the label matrix returned by `watershed` have different indices than they did before R2011a.

Also, the label matrix returned by `watershed` was class `double` in previous releases, and is now an unsigned integer class.

If you want to return a label matrix of class `double`, as you did before, use the `double` function to convert it:

```
L = watershed(A);  
L = double(L);
```

iccread and iccwrite Now Warn in Cases of Unrecognized PrimaryPlatform Signatures

If `iccread` or `iccwrite` encounter an unrecognized `PrimaryPlatform` signature in the profile header, they will warn. In releases before R2011a, these functions would error instead of warn in cases with unrecognized `PrimaryPlatform` signatures.

Plot Selector Now Includes implay

The `implay` function has been added to the list of functions available in the Plot Selector. You can now display data in `implay` directly from the Plot Selector workspace tool. For details about the Plot Selector, see [Enhanced Plot Selector Simplifies Data Display](#).

Support for Code Generation from MATLAB

You can now generate standalone C code for two Image Processing Toolbox functions: `label2rgb` and `fspecial`. The generated C code meets the strict memory and data type requirements of embedded target environments. To generate this code you need a MATLAB Coder license. See the [Code Generation for Image Processing Toolbox Functions](#) chapter in the User's Guide for details, including limitations.

edge Function No Longer Smooths Image Twice

In previous releases, the implementation of the Canny filter, called with the `canny` method of the `edge` function, smoothed the image twice while constructing the gradient

image. The function smoothed the image once using a Gaussian filter and then used a first derivative of a Gaussian filter to extract a smoothed version of the image gradient. Smoothing an image and then differentiating it is the same as convolving the image with a derivative of the smoothing kernel, so this implementation had the effect of smoothing the image twice. In addition, the original implementation of the Canny filter included an extra morphological thinning step that is not in the published algorithm.

Compatibility Considerations

The `edge` function no longer smooths an image twice. If you are setting the value of `sigma` and want similar results to the previous implementation, increase `sigma` by a factor of $\sqrt{2}$.

To achieve the same results produced by the previous implementation, use this syntax:

```
BW = edge(I, 'canny_old', ...)
```

Functions and Function Elements Being Removed

Function or Function Element	What Happens When You Use This Function or Element?	Use This Instead	Compatibility Considerations
<code>ipttable</code>	Errors	<code>uitable</code>	Replace all existing instances of <code>ipttable</code> with <code>uitable</code> .

R2010b

Version: 7.1

New Features

Bug Fixes

Compatibility Considerations

New corner Function Detects Corners in Image

The new `corner` function detects corners in a grayscale or binary image. Corners are a feature you can use to find the correspondence between images.

Compatibility Considerations

In R2008b and later releases, you could find corners by computing a `cornermetric` matrix with the `cornermetric` function and then finding peak values. Now, you can simplify your workflow by using the `corner` function.

Efficient Display and Navigation of Very Large Images of Arbitrary Format in `imshow`

The `rsetwrite` function allows you to create a multiresolution image pyramid (R-Set) from a large image file. In previous releases, the `rsetwrite` function accepted TIFF or NITF image files. Now, in addition to accepting these image files, `rsetwrite` accepts `ImageAdapter` objects. `ImageAdapter` objects allow you to work with images of arbitrary file format. See *Working with Data in Unsupported Formats* for guidelines on how to create an `ImageAdapter` object.

Now Possible to Control Padding Behavior when Using the `blockproc` Function

With the new `'PadMethod'` option, you can now control padding behavior when using the `blockproc` function. In previous releases, the `blockproc` function only supported zero padding along the boundary of the image. Now, the function supports padding the image with a scalar pad value, repeated border elements of A, or the mirror reflection of A.

Writing to JPEG2000 File Format Supported by `blockproc`

The `blockproc` function now provides more flexibility in format choices. The function supports writing to JPEG2000 file formats with `*.jp2`, `*j2c`, or `*.j2k` extensions.

Enhancements to the `dicomread` Function

The `dicomread` function has been enhanced in two ways: the function now reads multiframe imagery faster, and it reads files containing JPEG-2000 encoded imagery.

The `ImageMagnification` Field of the `nitfinfo` Function Now Returns a Numeric Value

The `ImageMagnification` field of the `nitfinfo` function has been updated. Previously, if you used the function to return a structure with file-level metadata, the `ImageMagnification` field of the structure contained an incorrect value. (The incorrect value was either an empty image magnification value or the text value for the field.) Now, the `ImageMagnification` field returns the value for the image magnification.

Performance Improvements

Faster Functions

- `blockproc`
- `imresize`
- `iradon`

Functions and Function Elements Being Removed

Function or Function Element Name	What Happens When You Use This Function or Element?	Use This Instead	Compatibility Considerations
<code>blkproc</code>	Still runs	<code>blockproc</code>	None

R2010a

Version: 7.0

New Features

Bug Fixes

Compatibility Considerations

New ImageAdapter Class Supports Custom File Formats for blockproc

The `blockproc` function, introduced in R2009b, supported file-based block processing for arbitrarily large images. In R2009b, you could use `blockproc` to read or write TIFF images or to read JPEG2000 images. Now, with the addition of the new `ImageAdapter` class, you can design your own class to use `blockproc` with images of arbitrary file format.

The blockproc Function Now Supports Spatially Varying Operations

Additional fields have been added to the `blockproc` “block struct” that contain spatial information. These new fields facilitate operations that depend on location.

Plot Selector Now Generates Plots for imshow and imtool

The Plot Selector workspace tool creates graphs of workspace variables. The `imshow` and `imtool` functions have been added to the list of possible plotting functions available in the Plot Selector. For more information about the Plot Selector, see [Enhanced Plot Selector Simplifies Data Display](#).

makecform Now Supports White Point Adaptation

`makecform` uses the white point specified by the International Color Consortium (ICC) as the default for the `srgb2lab` and `lab2srgb` transform types. You can now adapt to a white point other than `whitepoint('ICC')`, the default value, by using a new syntax to specify the adapted white point:

```
C = makecform(type, 'AdaptedWhitePoint', WP)
```

You can also create a linear chromatic-adaptation transform:

```
C = makecform('adapt', 'WhiteStart', WPS, 'WhiteEnd', WPE, ...  
    'AdaptModel', modelname)
```

This transform allows you to adapt XYZ color values from one white point to another.

Intel Integrated Performance Primitives Library Support Extended to imdilate, imerode, and medfilt2

The functions `imdilate` and `imerode` are now hardware optimized for ones (3) neighborhoods for `single`, `uint8`, and `uint16` input images.

The `medfilt2` function is now hardware optimized for integer data types (`uint8`, `uint16`, and `int16`) and the `single` data type with kernel size 3 x 3.

imreconstruct Now Supports int64 and uint64

The `imreconstruct` function now supports data types `int64` and `uint64`.

Performance Improvements

Faster Functions

- `edge`
- `imdilate`
- `imerode`
- `imfilter`
- `imresize`
- `iradon`
- `medfilt2`

Multithreaded Functions

- `bwmorph`
- `edge`
- `imabsdiff`
- `imadd`
- `imclose`
- `imdivide`
- `immultiply`
- `imopen`

- `iradon`
- `medfilt2`

Non-interactive Syntax of `improfile` Returns Different Output

One of the non-interactive syntaxes of `improfile` now returns different output. The output for the syntax

```
C = improfile(I,xi,yi,N)
```

has changed. In the syntax above, `N` specifies the number of points for which to compute intensity values and `xi` and `yi` specify the spatial coordinates of the endpoints of the line segments.

For a given line defined by `xi` and `yi`, `improfile` now returns a profile sampled at both endpoints and all sampling points in between at roughly unit interval spacing. If the distance between `xi` and `yi` is `N` pixels, the profile is evaluated at `N+1` points.

Compatibility Considerations

In previous releases, if you supplied the `xi` and `yi` end points as (1,1) and (10,1), the profile would be evaluated at nine points, the nine unit-length intervals between 1 and 10 in the continuous `x-y` plane. These nine points would be the two end points, plus seven points in between.

R2009b

Version: 6.4

New Features

Bug Fixes

Compatibility Considerations

New blockproc Function to Process Large Images

The new `blockproc` function supports file-based block processing for arbitrarily large TIFF images. The new function supports in-memory operations as well as file-to-file processing of images which are too large to load completely into memory.

Compatibility Considerations

In previous releases, you could use the `blkproc` function for in-memory block-processing of images. The `blkproc` function will be removed in a future release. Replace all instances of `blkproc` with `blockproc`.

When updating your code from `blkproc` to `blockproc`, it is important to note that the user-defined function, `fun`, has a new signature. It now takes a structure, the "block struct," as input instead of simply a matrix of image data.

The example below demonstrate how to update your code from `blkproc` to `blockproc`.

```
% BLKPROC code
I = imread('cameraman.tif');
fun = @dct2;
J = blkproc(I,[8 8],fun);

% BLOCKPROC equivalent (using an anonymous function)
fun = @(block_struct) dct2(block_struct.data);
J = blockproc(I,[8 8],fun);
```

Here's another example showing how to update your code from `blkproc` to `blockproc`.

```
% BLKPROC code
I = imread('concordorthophoto.png');
h = fspecial('gaussian',[11 11],2.5);
fun = @(x) imfilter(x,h,'conv','same');
J = blkproc(I,[500 500],[5 5],fun);

% BLOCKPROC equivalent (using an anonymous function)
fun = @(block_struct) imfilter(block_struct.data,h,'conv','same');
J = blockproc(I,[500 500],fun,'BorderSize',[5 5]);
```

Intel Integrated Performance Primitives Library Upgraded and Support Extended to maci64

The Intel® Integrated Performance Primitives (Intel IPP) Library has been upgraded from Version 5.3.1 to Version 6.0 Update 1. Intel IPP Library support has been extended to 64-bit Intel-based Mac computers.

Expanded hough Function Allows Specification of Arbitrary Theta Search Space

The hough function now yields faster results for narrower *theta* ranges due to the addition of a parameter/value pair for specifying *theta* values.

Compatibility Considerations

In previous releases, the 'ThetaResolution' parameter controlled the *theta* values for the hough function. Now 'ThetaResolution' is being replaced by the new 'Theta' parameter.

Function Elements Being Removed

Function and Syntax	What Happens When You Use the Function or Element?	Use This Instead	Compatibility Considerations
<code>hough(BW, 'ThetaResolution', val)</code>	Still runs	<code>hough(BW, 'Theta', -90:val:(90-val))</code>	Input parameter no longer recommended. Use new 'Theta' parameter.

Note that with the introduction of the 'Theta' parameter, not all abbreviated forms of 'ThetaResolution' will still work. In previous releases, if you entered the following syntax:

```
hough(BW, 'T', val)
```

'T' stood for 'ThetaResolution'. Now if you enter this same syntax, 'T' stands for the new 'Theta' parameter.

If you have old code that uses the 'ThetaResolution' parameter, please see the definition below:

Parameter	Description
'ThetaResolution'	Real scalar value between 0 and 90, exclusive, that specifies the spacing (in degrees) of the Hough transform bins along the <i>theta</i> axis. Default: 1.

For 'ThetaResolution', $n_{\theta} = 2 * \text{ceil}(90 / \text{ThetaResolution})$. *theta* angle values are in the range [-90, 90) degrees. If $90 / \text{ThetaResolution}$ is not an integer, the actual angle spacing is $90 / \text{ceil}(90 / \text{ThetaResolution})$.

The imfilter Function Now Faster for uint16 and double Inputs

The `imfilter` function now runs faster with `uint16` and `double` inputs than in previous releases. This performance enhancement is due to the use of the Intel IPP Library with inputs of these types.

Compatibility Considerations

Using the Intel IPP library for `uint16` images, poses no compatibility issues. The same is not true, however, for the `double` data type.

If an input image contains NaN values and a filtering kernel contains zero values, the `imfilter` function now gives different results when the Intel IPP library is enabled versus when it is disabled. If you want to preserve the behavior of previous releases, use `iptsetpref('UseIPPL', false)` to disable the Intel IPP library.

Improved Speed for Calculating N-D Euclidean Distance Transforms with the bwdist Function

A new algorithm improves the speed and reduces the memory footprint for the `bwdist` function.

Compatibility Considerations

In previous releases, the `bwdist` function used different algorithms for computing the Euclidean distance transform and the associated label matrix. If you need the same results produced by the previous implementation, use the function `bwdist_old`.

Modified Behavior for the `regionprops ConvexHull` Property

The `'ConvexHull'` property of `regionprops` depends on the MATLAB `convhull` function. Due to changes in `convhull`, the results returned by `'ConvexHull'` will now be slightly different than in previous releases.

Compatibility Considerations

The order of the vertices returned by the `'ConvexHull'` property of `regionprops` may differ from that returned in releases before R2009b. Also, the returned hull may contain additional collinear points that were omitted in previous releases.

Efficient Display and Navigation of Very Large NITF-File Images in `imtool`

The `rsetwrite` function allows you to create multi-resolution image pyramids (R-Sets) that you can open in `imtool`. In previous releases, `rsetwrite` worked only with TIFF files. Now it accepts NITF files, as well, as long as they are Version 2.0 or greater, contain an uncompressed image, have integer data (no floating point data), and have three or fewer image bands. Finally, if a NITF file has more than one band of data, the data must be unsigned.

Performance Improvements

The performance of several existing toolbox functions has been improved in this release. In some cases, other toolbox functions call these functions and therefore will also benefit from these speed improvements.

Faster Functions

- `bwdist`
- `imcomplement`

- `imdilate`
- `imerode`
- `imfilter`
- `improfile`
- `imrotate`

Multithreaded Functions

- `applylut`
- `bwpack`
- `bwunpack`
- `imdilate`
- `imerode`
- `imreconstruct`

R2009a

Version: 6.3

New Features

Bug Fixes

Compatibility Considerations

Faster, Less Memory-Intensive Workflow for Labeling Regions and Measuring Their Properties

The `bwconncomp` function computes connected components for binary images. It uses significantly less memory and is sometimes faster than `bwlabel` and `bwlabeln`.

To extract features from a binary image using `regionprops` with default connectivity, just pass `BW` directly into `regionprops` (i.e., `regionprops(BW)`). To compute a label matrix having more memory-efficient data type (e.g., `uint8` versus `double`), use the `labelmatrix` function on the output of `bwconncomp`.

Multithreaded Implementation of `imfilter` Function

The `imfilter` function is now multithreaded.

Efficient Display and Navigation of Very Large Images in `imtool`

The new `rsetwrite` function allows you to create a multi-resolution image pyramid (R-Set) from a large TIFF image file. In previous releases, large images would not open in `imtool`, or they did open, but navigation was slow. You can now open your R-Set with `imtool` and explore it as you would a standard image.

New Dialog Box for Setting Toolbox Preferences

A new preferences dialog box allows customization of Image Processing Toolbox preferences. You can access the dialog box via the **File** menu in the MATLAB desktop, the **File** menu in the Image Tool (`imtool`), or directly from the command line by typing `iptprefs`.

A new preference has been added that allows you to specify whether the Overview tool opens automatically when you launch the Image Tool.

Compatibility Considerations

In previous releases, the Overview tool opened automatically with `imtool`. The new default behavior is for the Overview tool to no longer open automatically. If you would like to revert to the previous behavior you can set this preference via the Image Processing Preferences dialog box (`iptprefs`).

In previous releases, if you changed the preferences with the `iptsetpref` command, these changes would revert to the default setting when you finished a MATLAB session. Now, if you change preferences, these changes will remain intact from one MATLAB session to the next.

New `imcolormaptool` Function That Opens Choose Colormap Tool

The new function `imcolormaptool` opens the Choose Colormap tool. The Choose Colormap tool allows you to interactively change the colormap of a displayed image. You can also access the tool from the **Tools** menu of the Image Tool, as in previous releases.

End Point and Branch Point Detection Now Possible

`bwmorph` now detects end points and branch points in binary images.

`nitfread` Now Allows Image Subregion Selection

The `nitfread` function now includes a `PixelRegion` parameter that returns a subimage as specified by row and column vectors.

Compatibility Considerations

From R2007a to R2008b, the `nitfread` function returned `uint8` data for images with 1-bit data. Now `nitfread` returns `logical` data for images with 1-bit data. If you want this function to behave as it did in the past, enter the following:

```
imdata = uint8(nitfread(filename));
```

Support for Intel IPP on Mac

In previous releases, the Image Processing Toolbox leveraged the Intel Integrated Performance Primitives (Intel IPP) Library on 32- and 64-bit Linux® and Windows® platforms. Now Intel IPP-use has been extended to the Mac.

getColor, getLabelVisible, and setLabelVisible Methods Added to imdistline

`imdistline` now includes a `getColor` method that returns the color used to draw a specific ROI object. Also, the new `getLabelVisible` and `setLabelVisible` methods make it possible to control the visibility of the Distance tool text label.

Five Functions Moved to MATLAB

The following five functions moved from the Image Processing Toolbox to MATLAB: `cmpermute`, `cmunique`, `dither`, `imapprox`, and `rgb2ind`. The behavior of some of the functions has changed slightly, as described in the compatibility considerations listed below.

Compatibility Considerations

- Functions `dither` and `imapprox`, when called without output arguments, no longer display their output as an image via a call to `imshow`. Now, if you want to display the resulting image, assign the output to one or more variables and call `imshow`. For example, try the following:

```
[Y,newmap] = imapprox(X,map,n)
imshow(Y,newmap)
```

- Function `rgb2ind` errors when called with the syntax `rgb2ind(RGB)`. You must specify the number of colors, tolerance, or colormap. For example, you can use the syntax `rgb2ind(RGB, 128)`, where 128 represents the number of colors.
- Function `imapprox` errors when called with the syntax `imapprox(x, map)`. As with `rgb2ind`, you must specify additional parameters.

Fan-Beam Functions Updated

Compatibility Considerations

Due to a bug fix, the fan-beam functions (`fanbeam`, `ifanbeam`, `fan2para`, `para2fan`) now return different answers than in previous releases.

R2008b

Version: 6.2

New Features

Bug Fixes

Compatibility Considerations

Performance Improvements

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

- Binary erosion and dilation (`imdilate`, `imerode`, `bwhitmiss`, and `rangefilt`)
- `graycomatrix`
- Image arithmetic and filtering now leverage the IPP Library on 32- and 64-bit Windows and Linux platforms.

New `cornermetric` Function Detects Corners

New `cornermetric` function detects corners.

Now Support Absolute Colorimetric Rendering Intent for `GrayTRC` and `MatTRC`

New additions to the `makecform` syntax include rendering intents for the Matrix/Tone Reproduction Curve (`MatTRC`) model and the single-channel Tone Reproduction Curve (`GrayTRC`) model.

New `createMask` Method Creates Mask for Any ROI

Use the new `createMask` method of the `imroi` base class to return a mask, or binary image, that is the same size as the input image with 1s inside the ROI object and 0s outside. The new method is available in the following classes: `impoint`, `imline`, `imrect`, `imellipse`, `impoly`, and `imfreehand`.

Interactive Tools Refresh when Target Image Changes

The following modular interactive tools now update automatically if you modify the target image: Adjust Contrast, Pixel Region, Pixel Information, Overview, Display Range, and Image Information.

The imscrollpanel 'PreserveView' Parameter Now Works for Images of All Sizes

The `replaceImage` function in the `imscrollpanel` API has been modified. You can now use the 'PreserveView' parameter even in cases where your replacement image is not the same size as your original image. The new image will appear with the center of view in the same relative position as in the original image.

Compatibility Considerations

In previous releases, the default for different size images was for the new image to appear centered and at 100% magnification.

Distance Tool and Cropping Tool Now Modes in imtool

The Distance tool and Cropping tool have been modified. Now to use the Distance tool, you click one end of the distance to be measured, drag, and release to complete the measurement. With the new version of the Cropping tool, you may click and drag to define the cropping region as many times as you want. If you define one region and then decide to crop a different region instead, simply click and drag the mouse again to define the new region.

Compatibility Considerations

In previous releases, the Distance tool appeared as a horizontal bar of set length. You could drag the ends of the tool to change size and orientation.

In previous releases, if you defined one cropping region, you could move this box or change the size, but you couldn't start with a new box unless you canceled the tool, clicked on the "Crop Image" toolbar button, and then defined the new region.

In imtool Opening Adjust Contrast Tool No Longer Selects Window/Level Tool

When you open the Adjust Contrast tool, the Window/Level tool is no longer turned on automatically. To operate this feature, simply select the Window/Level tool icon from the Image Tool toolbar. (To identify the icon, note that if you move the cursor over the

Window/Level tool icon, the words “Adjust contrast/brightness via mouse motion” appear.) Or, you can select “Tools” from the Image Tool menu and then click on “Window/Level.”

Compatibility Considerations

In previous releases, when you opened the Adjust Contrast tool, the Window/Level tool automatically turned on at the same time. Note that the Window/Level tool still turns on when you call `imcontrast` from the command line.

immovie Command No Longer Shows Preview

`immovie` no longer opens a figure window to display the movie as it is being created. You can display and explore the output of `immovie` using `implay`.

Compatibility Considerations

If you want to use `movie` to visualize the output but don't know how to set up the figure appropriately, call `imshow` on one of the movie frames first before calling `movie`.

Replace Calls to ipttable Function with MATLAB uitable Function

The `ipttable` function is being deprecated and will be removed in a future release.

Compatibility Considerations

If you used the `ipttable` function to display tabular data, you should replace use of `ipttable` with the MATLAB function `uitable`.

If you used the cell array syntax of `ipttable`, make the following changes to your code to achieve a similar effect using `uitable`. For more information about using `uitable`, see the `uitable` function reference page.

R2008a Code	R2008b Code
<pre>table = ipttable(parent, cell_array_data) ;</pre>	<pre>table = uitable(parent, 'Data', cell_array_data);</pre>

If you used the struct syntax of `ipttable`, make the following changes to your code to achieve a similar effect with `uitable`.

R2008a Code	R2008b Code
<pre>table = ipttable(parent, struct_data);</pre>	<pre>field_names = fieldnames(struct_data); values = struct2cell(struct_data); for idx = 1:numel(values) val = values{idx}; if ~ischar(val) size(val,1) > 1 values{idx} = evalc('disp(values{idx})'); end end table = uitable(parent, 'Data', [field_names values]);</pre>

Code written in previous releases that depends on `ipttable` will begin to warn and eventually error in later releases.

imcontour Second Output Argument Changed

The second output argument of `imcontour` is now a handle to an `hggroup` object instead of an array of handles to patch objects.

Compatibility Considerations

If you need to access handles of individual patch objects, use the following code to work around the change.

```
[c, handleToHGGroup] = imcontour(..);  
arrayOfHandlesToPatchObjects = get(handleToHGGroup, 'Child');
```

impixelinfo Tool Disappears when Image Changes

If you use `imshow` to display an image, open the `impixelinfo` tool, and use `imshow` to open a new image, the `impixelinfo` tool will disappear along with the first image.

Compatibility Considerations

In previous versions, if you entered the following code:

```
imshow pout.tif
impixelinfo
imshow peppers.png
```

the `impixelinfo` tool would update to reflect changes to the image. Now you must call the `impixelinfo` tool again after opening the second image.

Some Code Moved into Different Directories

- Colorspace functionality moved into the new `toolbox/images/colorspaces` directory.
- Medical file formats moved into the `toolbox/images/iptformats` directory with other file formats, and the `toolbox/images/medformats` directory was removed.

Functions and Demos Being Removed

Function or Demo Name	What Happens When You Use Function or Demo?	Use This Instead	Compatibility Considerations
<code>pixval</code>	Errors	Use <code>impixelinfo</code> for pixel reporting and <code>imdistline</code> for measuring distance.	Replace all existing instances of <code>pixval</code> with <code>impixelinfo</code> or <code>imdistline</code> .
<code>dctdemo</code>	Errors	NA	None
<code>edgedemo</code>	Errors	NA	None
<code>firdemo</code>	Errors	NA	None
<code>landsatdemo</code>	Errors	NA	None
<code>nrfiltdemo</code>	Errors	NA	None
<code>qtdemo</code>	Errors	NA	None
<code>roidemo</code>	Errors	NA	None

R2008a

Version: 6.1

New Features

Bug Fixes

Compatibility Considerations

Create High Dynamic Range (HDR) Images and Write Them to Files

Create a high dynamic range image from a group of low dynamic range images using the new `makehdr` function. The low dynamic range images must be spatially registered. You can write the HDR image to a file using the `hdrwrite` function. These functions complement the `hdrread` and `tonemap` functions introduced in R2007b.

Measure Properties of Regions in Grayscale Images

The `regionprops` function now accepts grayscale images as an input parameter, returning measurements based on the values of pixels in specified regions. Using `regionprops`, you can obtain measurements of regions in the image such as the maximum, minimum, and mean intensities in the region, and the weighted centroid.

Display Very Large Images by Subsampling

You can now display very large images from TIFF files by using the `imshow` function's new 'Reduce' parameter. When you specify this parameter, `imshow` displays a subsampled version of the image. `imshow` determines the subsampling factor by considering the size of the image and the reduction required to fit the image on your screen. The 'Reduce' parameter makes it possible to view very large images in their entirety that could not previously be displayed. Note, however, that the image subsampling that is performed reduces that amount of image data displayed.

Enhancements to ROI Tools

The toolbox includes several functions that enable the definition of regions of interest of various shapes: `impoint`, `imline`, `impoly`, `imrect`, and `imfreehand`. These ROI tools have several enhancements:

ROI Tools Reimplemented as MATLAB Classes

The ROI tools have been reimplemented as MATLAB classes. This change does not affect how the ROI tools function; they function identically to their previous implementation. The documentation uses the MATLAB functional syntax descriptions rather than the dot notation. That is, the documentation shows how to call the class methods specifying a handle to the object as the first argument, `method(h, ...)`. Note, however, that you can still use the dot notation when calling the methods, `obj.method(...)`. In addition, the

`iptgetapi` function now returns an object of the new class which means that code similar to the following will continue to work:

```
api = iptgetapi(h)
api.method()
```

Compatibility Considerations

The class of the data returned by the ROI tools is now a handle to an ROI class, such as `imline` or `impoly`. In addition, several undocumented methods supported by the ROI tools have been removed: `getContextMenu`, `setContextMenu`, `getDrawAPI`, `addCallback`, and `removeCallback`.

ROI Tools Support New wait and resume Methods

The ROI tools now support `wait` and `resume` methods so that they can be used in scripts. By using the `wait` method, you can enable users of your script to make the initial placement of the ROI, adjust the ROI and accept it, and then use the position in the script. For example, using the `wait` method with an ROI tool, you could write a script that creates a mask.

The `resume` method is a programmatic way to return control to the command line. When called after `wait`, `resume` causes `wait` to return the accepted position of the ROI.

Interactively Add New Vertices to ROI Polygons

You can now add vertices interactively to polygonal ROIs that you define using the `impoly` function. To create the new vertex, position the pointer over an edge of the polygon and press the A key. The pointer changes shape. Click the mouse to add a new vertex. The `roifill` and `roipoly` functions, which use `impoly` to implement ROIs, also support this new capability.

Enhancements to Color Functions

The following color functions have been enhanced.

makecform Supports Converting Between sRGB and CMYK

The `makecform` function now supports two new color space conversion types for converting between sRGB and CMYK: `'srgb2cmyk'` and `'cmyk2srgb'`.

iccwrite Creates Smaller ICC Profiles

The `iccwrite` function now uses certain optimizations to reduce the size of the International Color Consortium (ICC) color profiles that it creates. `iccwrite` uses aliasing to avoid writing tag data multiple times when it is included in more than one profile table.

cp2tform Function Supports New Transformations

The `cp2tform` function supports two new transformation types: `'similarity'` and `'nonreflective similarity'`.

Compatibility Considerations

The `'linear conformal'` transformation type supported by the `cp2tform` function has been renamed to `'nonreflective similarity'`.

hough Function Uses Specified RhoResolution Values

The `hough` function now uses the value you specify for the `'RhoResolution'` parameter. In previous releases, the function did not use the value specified.

Compatibility Considerations

The Hough matrix, `H`, and the Rho outputs returned by the `hough` function have different results than those obtained from the same function in previous releases.

Enhancements to Interactive Tools

The following modular interactive tools have been enhanced.

- Adjust Contrast tool (`imcontrast`) — The **Adjust Data** button in the Adjust Contrast tool is disabled until you make a change to image contrast.
- Pixel Region tool (`impixelregion`) — To improve the visibility of the image pixels being examined, the Pixel Region tool stops including grid lines in the display at low magnifications.

New and Updated Demos

The toolbox includes the following new and changed demos.

- *Batch Processing Image Files in Parallel* is an existing demo that has been updated, and simplified, through use of the `parfor` function.
- *Detecting Cars in a Video of Traffic* is a new demo that shows how to use the toolbox to visualize and analyze videos or image sequences.
- *Measuring Regions in Grayscale Images* is a new demo that shows how to use the `regionprops` function with grayscale images.

Enhancements to Other Functions

This release includes changes to the following functions.

Function	Description of Enhancement
<code>imageinfo</code>	Accepts files of several additional file formats as an input argument, including NITF, Interfile, and Analyze file formats.
<code>imshow</code>	Supports a new <code>colormap</code> parameter for specifying a colormap for grayscale images.
<code>imtool</code>	Supports a new <code>colormap</code> parameter for specifying a colormap for grayscale images.
<code>trueSize</code>	Preserves the border preference setting of the figure when adjusting the image display size.

R2007b

Version: 6.0

New Features

Bug Fixes

Compatibility Considerations


New Interactive Image Sequence and Video Viewer

The toolbox now supports a new interactive image sequence viewer, called the Movie Player (`implay`). Using the Movie Player you can:

- Play a MATLAB movie, AVI file, or multidimensional array.
- Step through a movie or sequence of images, frame-by-frame, or jump to the beginning or end of the sequence.
- Examine a frame using the Pixel Region tool or export the frame to the Image Tool.

Image Tool Includes Cropping, Enhanced Contrast Adjustment, and Saving of Modified Images

The Image Tool (`imtool`) supports several enhancements:

- You can now modify the image data after performing a contrast adjustment operation. Previously, contrast adjustment only affected the display of the image, not the actual image data. To modify image data, click **Adjust Data** in the Adjust Contrast tool.
- You can now interactively crop an image displayed in the Image Tool using the Crop Image button  in the toolbar (or select **Crop Image** from the Tools menu).
- You can now save the image displayed in the image tool in any of several common image file formats. Select **Save As** from the Image Tool File menu.

New Function for Converting Bayer Pattern Encoded Images to RGB

The toolbox now supports a function, `demosaic`, that can convert a Bayer pattern encoded image into an RGB image.

A Bayer filter mosaic, or color filter array, refers to the arrangement of color filters that let each sensor in a single-sensor digital camera record only red, green, or blue data. The patterns emphasize the number of green sensors to mimic the human eye's greater sensitivity to green light. The `demosaic` function uses interpolation to convert the two-dimensional Bayer-encoded image into a truecolor image, in the RGB color space.

New Function for Creating a Multiresolution Gaussian Pyramid

The toolbox now supports a function, `impyramid`, that you can use to create a multiresolution Gaussian pyramid. If you specify the 'reduce' parameter, `impyramid` returns a low-pass filtered version of the image, half the size of the original image. If you specify the 'expand' parameter, `impyramid` returns a filtered image twice the size of the original image. `impyramid` uses convolution with a Gaussian filter kernel to produce the images.

Enhanced ROI Definition Behavior for `imcrop`, `roifill`, and `roipoly`

The `imcrop`, `roifill`, and `roipoly` functions now let you define an ROI and then adjust the size and position of the ROI interactively using the mouse. In previous releases, these functions supported the interactive definition of ROIs, but only gave you one chance at the definition. Now, when you are satisfied with the size and shape of the ROI, double-click to perform the cropping, filling, or mask creation operation.

Compatibility Considerations

In previous releases, when defining a polygonal ROI using `roipoly`, pressing **Backspace** deleted the most recent vertex you had defined in the polygon. With this release, pressing **Backspace** deletes the entire polygon. To delete an individual vertex, move the pointer over the vertex, right-click to view the vertex context menu, and then choose **Delete Vertex**.

New Modular Interactive GUI-building Tools

The set of modular interactive tools now includes functions to display a file chooser dialog box and write data to a file (`imsave`). The toolbox also includes a function (`imputfile`) that displays the file chooser dialog box and returns the user's selections.

New Programmable ROI Tools

The set of programmable ROI creation functions provided by the toolbox now includes three additional shapes:

- Polygons (`impoly`)

- Ellipses (`imellipse`)
- Freehand shapes (`imfreehand`)

The toolbox already includes ROI creation functions to create points, lines, and rectangles.

Each of the ROI creation functions supports an API that you can use to control aspects of its behavior and appearance. For example, you can use API functions to specify the position of the ROI or retrieve the coordinates of its current position.

Support for Reading NITF and HDR Images

- Read metadata from a National Imagery Transmission Format (NITF) file using `nitfinfo`.
- Read an image from a NITF file using `nitfread`.
- Read high dynamic range (HDR) images using `hdrread`.
- Convert high dynamic range images into a format that can be displayed using the `tonemap` function.

Enhanced Performance

- Enhanced performance for thinning and skeletonization using `bwmorph`.
- Enhanced performance for filtering RGB images using `imfilter`.

DICOM Dictionary Upgrade

The default DICOM dictionary has been upgraded to the 2007 version released by NEMA. A text version of this dictionary is included in the product, `dicom-dict.txt`. This upgrade fixes a problem with the earlier version of the dictionary which contained two instances of the same tag, which caused warnings.

Compatibility Considerations

If your DICOM code depends on hard-coded old attribute names, you may see failures. In addition, some DICOM files may no longer parse. Customers who require attribute settings from the 2005 version can use the `dicomdict` function to access the old data dictionary, which we are shipping in R2007b. That is, `dicom-dict.txt` will have 2007

values and `dicom-dict-2005.txt` is the version of `dicom-dict.txt` found in R2006a and R2007a.

Changes to Other Functions

This release includes changes to the following functions.

Function	Description of Change
<code>imshow</code>	Is not supported when MATLAB is started with the <code>-nojvm</code> option.
<code>imhist</code>	Can now be embedded in custom GUIs.
<code>fanbeam</code> , <code>ifanbeam</code> , <code>fan2para</code> , <code>para2fan</code>	The fan-beam functions now return different answers than in previous releases due to a bug fix.
<code>imadjdemo</code>	This demo has been deleted from the toolbox.

R2007a

Version: 5.4

New Features

Bug Fixes

Compatibility Considerations

Enhancements to `imresize` Function

`imresize` now runs faster, uses less memory, supports new interpolation methods, and supports new options for specifying output size.

Compatibility Considerations

The `imresize` function has been completely rewritten with new algorithms, new options, and new syntaxes. If you need the results produced by the version of `imresize` in previous releases, use the `imresize_old` function.

`applycform` Supports Tetrahedral Interpolation

The `applycform` function now uses tetrahedral interpolation for profiles containing multidimensional lookup tables, and returns more accurate results.

Compatibility Considerations

The results returned by `applycform` are more accurate but they are different than results returned in previous releases, for profiles containing multidimensional lookup tables.

Control Point Selection Tool Enhancements

The Control Point Selection Tool has enhanced visual appearance and usability. For example, points are now numbered for easier identification of matched pairs.

In addition, the tool now supports a `'wait'` option which enables `cpselect` to be used in scripts. When you specify this option, `cpselect` blocks the MATLAB command line until point selection is completed. For information about using the Control Point Selection Tool, see Image Registration and the reference page for the `cpselect` function.

Compatibility Considerations

The Control Point Selection Tool no longer includes the **Redo** or **Undo** options on the Edit menu.

Enhancements to `impoint`, `imline`, and `imrect` Functions

The `impoint`, `imline`, and `imrect` function now support an interactive placement capability. Using the mouse, you can specify the initial position of the point, line, or rectangle. In addition, the `imrect` function now supports interactive resizing using the mouse. See the reference pages for these functions for more information and examples.

Enhancements to `montage` Function

The `montage` function now supports parameters that control the arrangement and appearance of the images displayed. See the `montage` reference page for these functions for more information and examples

`makecform` Uses 'icc' Whitepoint for $L^*a^*b^*/sRGB$ Conversions

The `makecform` function now only uses the white point type 'icc' for color space conversions from $L^*a^*b^*$ to `srgb` (type = 'lab2srgb') and from `srgb` to $L^*a^*b^*$ (type = 'srgb2lab').

Compatibility Considerations

In previous releases, you could specify other white point values for these conversions, using the optional 'Whitepoint' parameter. This syntax now issues a warning when any other white point besides 'icc' is specified.

`normxcorr2` Might Return Different Results

The `normxcorr2` function now returns values in the range $[-1, 1]$ for all inputs. In previous releases, `normxcorr2` returned values outside this range for certain inputs.

`watershed` Uses New Algorithm

The `watershed` transform algorithm used by the `watershed` function has changed. The previous algorithm occasionally produced labeled watershed basins that were not contiguous..

Compatibility Considerations

If you need to obtain the same results as the previous algorithm, use the function `watershed_old`.

Changes to Other Functions

This release includes changes to the following functions.

Function	Description of Change
<code>imshow</code>	New 'border' parameter, to control whether <code>imshow</code> includes a border around the image displayed, and 'parent' parameter, to specify the axes in which to display the image.
<code>imshowpanel</code>	New 'replaceImage' parameter lets you replace the image displayed in the scroll panel with a new image.
<code>iradon</code>	New 'none' value for the <code>filter</code> parameter returns an unfiltered backprojection; also supports new interpolation types.
<code>iptsetpref</code>	New UseIPPL preference.

R2006b

Version: 5.3

New Features

Bug Fixes

Compatibility Considerations

Enhancements to DICOM Capabilities

This release includes the following new features and enhancements to the DICOM capabilities of the Image Processing Toolbox:

- The toolbox includes a new function, `dicomlookup`, that provides a way to find the name of an attribute in a DICOM data dictionary by specifying its group and element tags, or find the group and element tags for an attribute by specifying its name.
- The performance of the `dicominfo` function has been significantly improved

New Symmetric Option with `graycomatrix` Function

The `graycomatrix` function now supports a new option: `'symmetric'`. With this option, you can create a gray-level co-occurrence matrix (GLCM) that is symmetric about its diagonal. This is consistent with the GLCM definition given by Haralick in his 1973 article. For more information, see `graycomatrix`.

Enhancements to ICC Color Capabilities

The toolbox includes the following enhancements to the ICC color capabilities:

- The `applycform` function can now transform colors using profiles that contain parametric curve types.
- The `iccread` function now supports named colors in ICC profiles.

Compatibility Considerations

The `whitepoint` function, when used with the `'d50'` argument, returns different results in R2006b than it did in R2006a. The previously returned XYZ color values were incorrect according to the current interpretation of standards. If your algorithm depended on the old values, you might see subtly different results.

Enhancements to the `imdistline` Function

This release includes the following enhancements to the `imdistline` function:

- The `imdistline` function now uses a different cursor shape at its endpoints to highlight that these endpoints can be grabbed to change the length or direction of the

line. The function uses a hand cursor over endpoints and a fleur cursor over the body of the line.

- The `imdistline` function reference page now includes an example that shows how to use the `XData` and `YData` properties of the associated image to express distance in non-pixel units.

Compatibility Considerations

The Distance Tool's `getAngleFromHorizontal` method now returns a value between 0 and 180 degrees. Previously, this function incorrectly returned a value between 0 and 90. For an explanation of how `getAngleFromHorizontal` calculates this angle, see the `imdistline` function.

setColor Method Accepts Predefined Color Strings

The `setColor` method of the `imdistline`, `imline`, `impoint`, and `imrect` functions accepts an RGB triplet or the short- or long-name version of the MATLAB predefined color names.

R2006a

Version: 5.2

New Features

Bug Fixes

Compatibility Considerations

Enhanced ICC Profile Capabilities

The `iccread` and `iccwrite` functions have been updated to support recent changes to the ICC specification.

In addition, `iccread` can now read and process the following additional profile types:

- DeviceLink profiles — Provide transformation from one device space to another.
- ColorSpace profiles — Provide transformation between a non-device color space and the profile connection space (PCS).
- Abstract profiles — Enable color transformations to be defined that provide specific color effects.
- Grayscale profiles — Specify the relationship between device values and the PCS for specific colors.

In addition, `iccread` can now read parametric curve types.

New Pointer Management Functions

The toolbox includes three new utility functions, `iptPointerManager`, `iptGetPointerBehavior`, and `iptSetPointerBehavior`, that you can use to manage changes to the pointer in GUIs. For example, you can use the pointer management functions to change the appearance of the pointer when it moves over objects in a figure. These functions can be useful when building GUIs with the toolbox modular GUI tools.

New Constraint Creation Function

The toolbox includes a new utility function, `makeConstrainToRectFcn`, that you can use to specify drag constraints for the `imdistline`, `imline`, `impoint`, and `imrect` functions. You specify the constraints as arguments to the `makeConstrainToRectFcn` and this function returns a handle to a constraint function. To use this constraint with an object, set the value of the `setConstraintFcn` API for the object to this function handle.

Functions `cp2tform`, `tforminv`, `imtransform`

When using the `cp2tform`, `tforminv`, or `imtransform` functions with the transform type `'piecewise linear'` you might get different answers from previous versions due to a bug fix. If you have a transformation structure (TFORM) saved from an older version, you may want to regenerate it from control points to get improved performance.

Compatibility Considerations

If you have a transformation structure (TFORM) saved from an older version, you may want to regenerate it from control points to get improved performance.

IPPL Not Used on 64-Bit Systems

Certain functions in the Image Processing Toolbox, such as the image arithmetic functions, use the Intel Performance Primitives Library (IPPL), if it's available. (See `ipp1` for more information.) Note that these functions do not use the IPPL on 64-bit systems.

R14SP3

Version: 5.1

New Features

Bug Fixes

Compatibility Considerations

Support for Two New Medical Image File Formats

The toolbox now includes functions for reading metadata and image data from two additional medical image file formats. Analyze 7.5 and Interfile. For more information, see “Mayo Analyze 7.5 Files” and “Interfile Files”.

New Point, Rectangle, and Line Functions

The toolbox includes three functions, `impoint`, `imline`, and `imrect`, that you can use to create draggable points, lines, and rectangles in a figure window. These functions can be used as building blocks for other GUI tools.

Image Tool Enhancements and Improvements

New Distance Tool

The Image Tool now includes a new Distance tool that you can use to determine the distance between any two points in an image. This tool is also available in the toolbox's suite of modular interactive GUI tools. Using the `imdistline` function you can add the Distance tool to GUIs of your own creation. For more information, see [Measuring Features in an Image](#)

Adjust Contrast Tool Enhancements and Improvements

The Adjust Contrast tool has been redesigned to provide better usability. For examples, the Adjust Contrast tool Window/Level capability is now a separate mode with its own activation button.

New Utility Functions for Use with Profile-Based Color Space Conversion Functions

The toolbox has two new utility functions, `iccroot` and `iccfind`, for use with the profile-based color conversion functions. For more information, see [Performing Profile-based Color Space Conversions](#).

New Documentation on Processing Image Sequences

The Image Processing Toolbox User's Guide includes a new section, Working with Image Sequences, that describes which toolbox functions can be used with sequences of image, also known as image stacks

Control Point Selection Tool Now Works on Macintosh Systems

The Control Point Selection Tool now works on Macintosh systems.

Obsolete and Deleted Functions

Compatibility Considerations

The following table lists toolbox functions that have been made obsolete or removed in this version.

Function	Enhancement
<code>impositionrect</code>	This function is obsolete. Use <code>imrect</code> to perform the same tasks.
<code>pixval</code>	This function is obsolete. It now issues a warning when used. Use <code>impixelinfo</code> for pixel reporting and use <code>imdistline</code> for measuring distance

Image Tool is Not Compilable

Compatibility Considerations

The `imtool` function is not compilable with the MATLAB Compiler.

R14SP2

Version: 5.0.2

New Features

Major Bug Fixes

This release contains the following bug fixes.

Major Revisions to Fan-Beam Functions

This release includes numerous updates and improvements to the fan-beam functions: `fanbeam`, `ifanbeam`, `fan2para`, and `para2fan`. The fixes include improved calculations, improved documentation, and examples.

For example, `fanbeam` now returns the correct sensor locations when the geometry is `'line'`. The `ifanbeam` and `fan2para` now consistently use the correct default value for the `'FanSensorSpacing'` parameter. If you tried the fan-beam functions in a previous release, you might try them again to take advantage of these improvements.

In addition to the functional changes, many improvements to the documentation of the fan-beam functions have been made.

`fanbeam` help now includes

- An example that shows how to extract projection data at a specific rotation angle from the fan-beam data returned
- An explanation of how `fanbeam` calculates the number of rows and columns in `F`, the fan-beam data returned
- The default value for the `'FanSensorSpacing'` parameter for both `'line'` and `'arc'` geometries
- Guidelines for setting the value of the `D` parameter

The help for the `ifanbeam` function now includes an example that shows how to use the `'minimal'` coverage parameter.

Compatibility Considerations

Results computed with earlier versions of the fan-beam functions cannot be used with the new versions of these functions.

Changes to the DICOM Functions

The following fixes have been made to the `dicomread` and `dicomwrite` functions.

Function	Bug Fixes
dicomread	No longer errors when reading files that contain extraneous pixel data; instead, <code>dicomread</code> issues a warning message. However, if the file does not contain enough pixel data, <code>dicomread</code> issues an error.
dicomwrite	<ul style="list-style-type: none"> • No longer is case sensitive when parsing input parameters. For example, you can specify either <code>'CreateMode'</code> or <code>'createmode'</code>. • Preserves the full precision of data converted to decimal string metadata. Previously, <code>dicomwrite</code> limited precision to six digits. • No longer errors when writing files with metadata values that must be stored as a decimal string or integer string. Now, when writing private data attributes (attributes that are not listed in the DICOM data dictionary), <code>dicomwrite</code> assigns the attributes the type UN (for unknown) and writes the data to the file as a byte-for-byte copy of its in-memory representation. Because <code>dicomwrite</code> writes the file with explicit value representation (VR), the file might have a different VR value, but the data will be the same. • Includes the <code>TriggerTime</code> field for additional values of <code>ScanOptions</code>, including <code>'CT'</code>. Previously, <code>dicomwrite</code> only included the <code>TriggerTime</code> attribute if the <code>ScanOptions</code> field indicated a gated heart MR. • No longer issues an <code>Unsupported SOP class</code> error message if, when <code>'create'</code> mode is specified, semantic verification is not available for an information object. Instead, <code>dicomwrite</code> issues a more helpful message indicating that it might be able to write the data if the mode was <code>'copy'</code>, rather than <code>'create'</code>. In <code>'copy'</code> mode, <code>dicomwrite</code> only performs syntactic checking, not semantic verification. Consequently, <code>dicomwrite</code> can write many more types of DICOM files in <code>'copy'</code> mode than it can in <code>'create'</code> mode. See the <code>dicomwrite</code> reference page for important information about data integrity.

Changes to Image Tool and Modular Interactive Tools

The following fixes have been made to the Image Tool and other modular interactive tools:

- The Image Tool now always makes the **Open** and **Import from Workspace** options available on its **File** menu. Previously, the Image Tool disabled these options if the tool

contained an image. If the Image Tool contains an image, the newly imported image is displayed in a new Image Tool using the default preferences.

- The Image Tool zoom buttons can now be used on an image that has superimposed vector data.
- The Image Tool toolbar buttons no longer create multiple versions of the modular interactive tools when clicked rapidly in quick succession.
- The Image Information tool now displays correctly on Linux systems. Previously, it displayed as a blank window.
- The Overview tool can now be resized from any corner. Previously, resizing the tool using a corner other than the lower left caused the image to become progressively smaller until it disappeared.
- The Overview tool zoom buttons now provide an affordance that informs users when they cannot use these buttons to zoom in or out on the image displayed in the associated scroll panel.
- The Pixel Region tool now displays floating-point values correctly. Previously, the pixel value text strings displayed spilled over into adjacent pixels for some floating-point images.
- The Pixel Region tool now works correctly with images displayed in subplots.
- The Pixel Region tool no longer causes the target image to become tiny and move to a different position in the figure.

Changes to the imshow Function

- The `imshow` function no longer overwrites nondefault axes in a figure.
- The `imshow` function ignores any initial magnification value you specify when used to display an image in a figure that is docked (the figure's `WindowState` property is set to `'docked'`). In these cases, `imshow` displays the image at the largest magnification that fits the window (`'fit'` magnification) and issues a warning.

Fixes to Other Functions

The following tables lists fixes that have been done to other toolbox functions.

Function	Enhancement
<code>applycform</code>	Now correctly handles profiles that contain a <code>gamut</code> tag.

Function	Enhancement
cpcorr	Now is more numerically robust. For this release, the subfunction <code>findpeak</code> , which <code>cpcorr</code> calls, has been improved and is now a private function, rather than a subfunction.
imhist	No longer causes a docked figure window to become undocked.
imrotate	Now correctly rotates N-dimensional arrays, where N is greater than 3. In previous releases, <code>imrotate</code> would accept N-D arrays but only return a 3-D array.
normxcorr2	Now always returns real values. In previous releases, due to round-off error, some sets of input data caused the <code>normxcorr2</code> function to return a complex valued matrix of correlation coefficients.
pixval	Now works correctly with binary images.
rgb2ind	Now returns a correct output image when called with the syntax <code>rgb2ind(rgb,n,'nodither')</code> where n is greater than 256.

Fixes to Image Processing Toolbox Deployment Issues

- Performance issues that occurred when deploying compiled image processing toolbox functions that call IPPL routines have been fixed.
- Running compiled versions of `imtool` and some of the other modular interactive tools no longer generates the following warning messages about classes not being cleared:

```
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.
```

New Directory Needed

The Image Processing Toolbox software now requires the following new directory on the MATLAB path:

toolbox\shared\imageslib