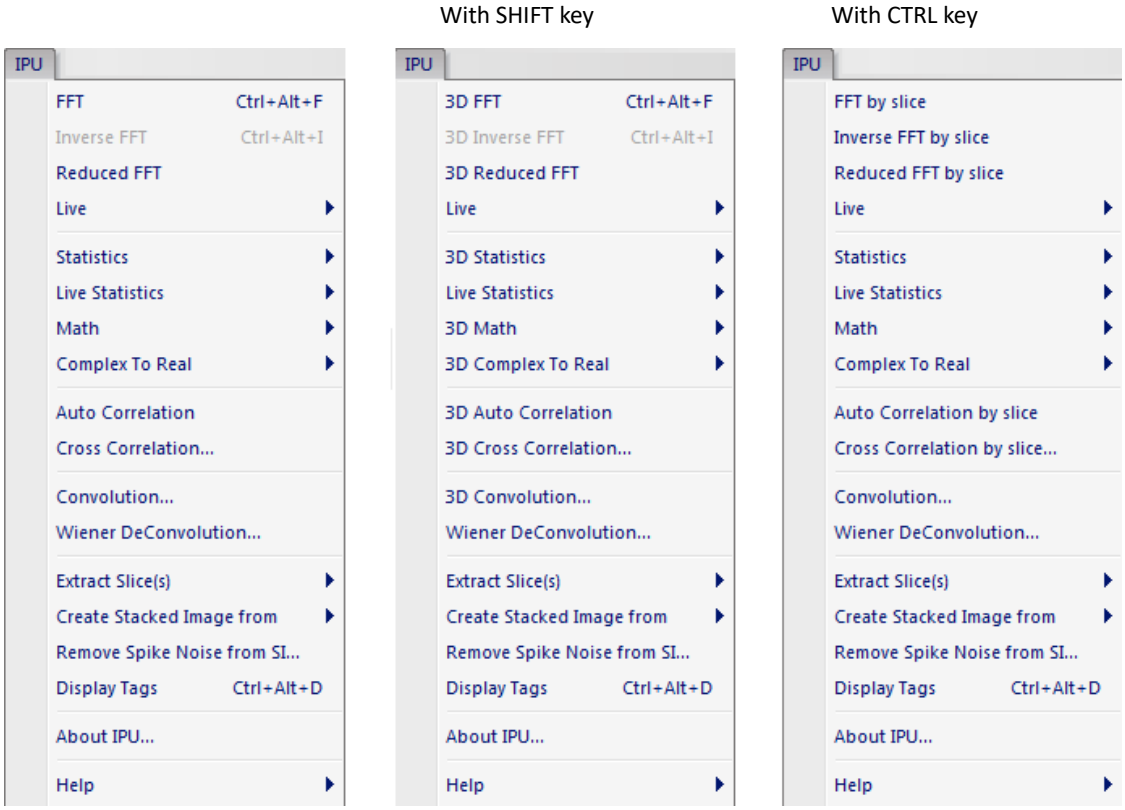
About IPU (Image Processing Utilities) Plug-in

2.12

Introduction

Image Processing Utilities (IPU) is a plug-in for Gatan's DigitalMicrograph™.

This plug-in will provide compatible commands of Process menu regarding FFT as shown below:



IPU Menu

Most of the commands switch to 3D processing by pressing the SHIFT key when the menu is selected, and to 2D processing for each slice by pressing the CTRL key.

These commands will work on an image with any image size, namely without a restriction of power of two, based on Intel's MKL (Math Kernel Library). We also provice with this plug-ins the FFT-relating functions that you can use from your script. These functions are also used internally by our several high-performance plug-ins.

This is a free plug-in, and we are happy to get your comments or suggestions on this plug-in.

Contents

Contact Us

General enquiries on the HREM Mouse should be sent to:

HREM Research Inc.
14-48 Matsukazedai
Higashimatsuyama
Saitama 355-0055
email: support@hremresearch.com
Website: http://www.hremresearch.com/

Copyright Statements

IPU

Acknowledgements

We are grateful to Gatan software term to support our plug-in development.

Installation

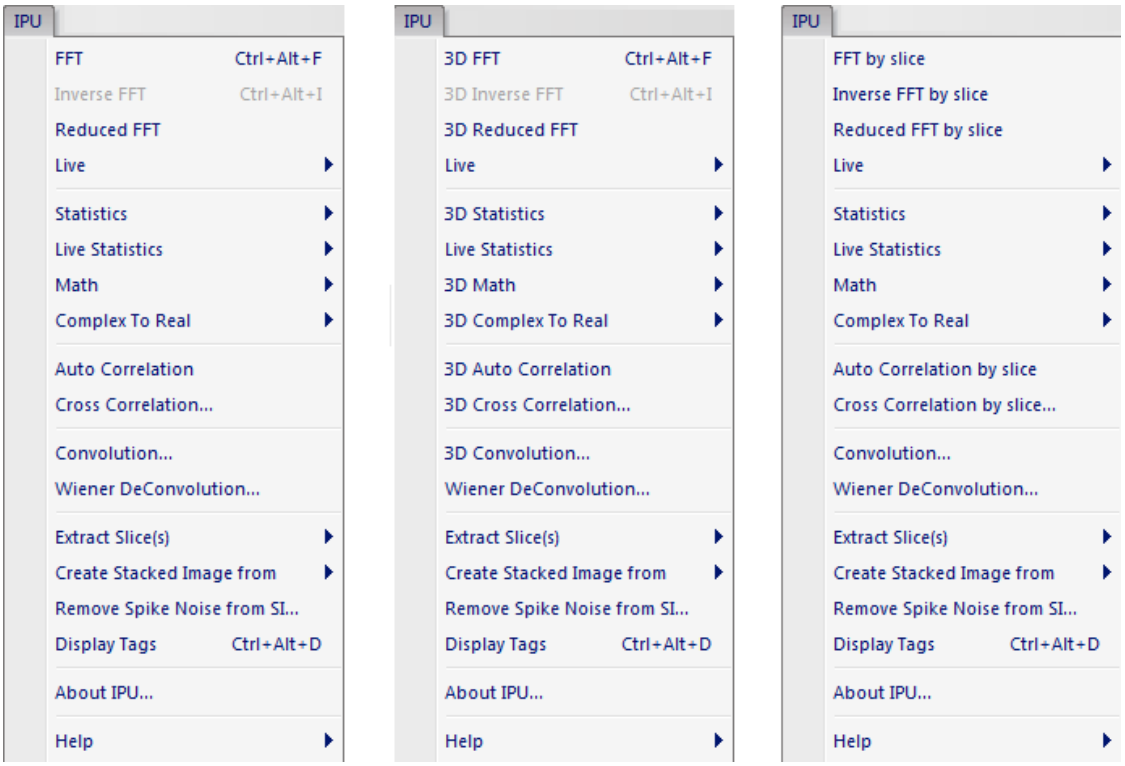
1. Exit (Quit) DigitalMicrograph if it is launched.
2. Copy IPU.dll and IPU.gtk to DigitalMicrograph/PlugIns.

Quick Reference

IPU Menu

With SHIFT key

With CTRL key



IPU Menu

Most of the commands switch to 3D processing by pressing the SHIFT key when the menu is selected, and to 2D processing for each slice by pressing the CTRL key.

FFT / 3D FFT

Same as DigitalMicrograph

Inverse FFT / 3D Inverse FFT

Same as DigitalMicrograph

Reduced FFT / 3D Reduced FFT

Same as DigitalMicrograph

Live

Same as DigitalMicrograph

Live Menu

FFT

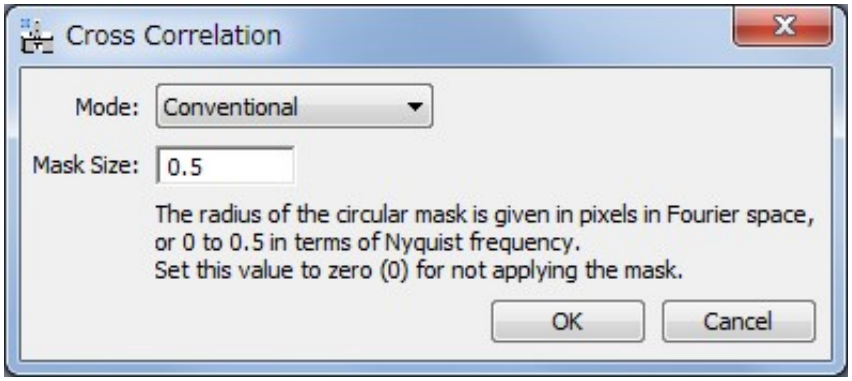
Reduced FFT

Auto Correlation / 3D Auto Correlation

Same as DigitalMicrograph

Cross Correlation / 3D Cross Correlation

Extended cross-correlation function. By default this behaves same as DigitalMicrograph. However, if you choose this command with ALT key down, you will get a dialog shown below where you can select a cross-correlation mode and a mask size.



Cross Correlation Dialog

Mode

Cross correlation mode
1..Conventional, 2..Square Root Moduli, 3..Phase

Mask size

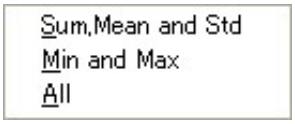
Mask radius given in pixels in Fourier space, or 0 to 0.5 in terms of Nyquist frequency.
Circular mask is used to remove high frequency noise.

Convolution / 3D Convolution

Calculates a convoluted image of selected two images.

Statistics / 3D Statistics

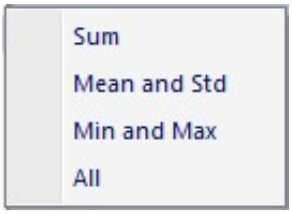
Same as DigitalMicrograph



Statistics Menu

Live Statistics

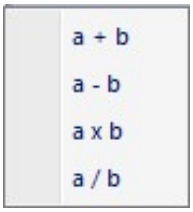
Use this command after placing a Rectangular ROI on an image. The selected statistics within the ROI will be dynamically updated when you move the ROI.



Live Statistics Menu

Math / 3D Math

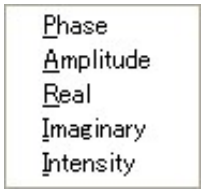
Same as DigitalMicrograph



Math Menu

Complex To Real / 3D Complex To Real

Creates a real image of the specified type from a complex image.



Complex To Real Menu

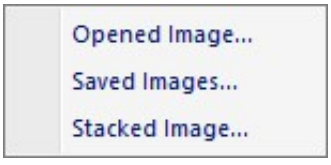
Extract Slice(s)

Extracts one currently displayed slice, or all silces, or selected slices from a front image.

Extract Slice(s) Menu

Create Stacked Image

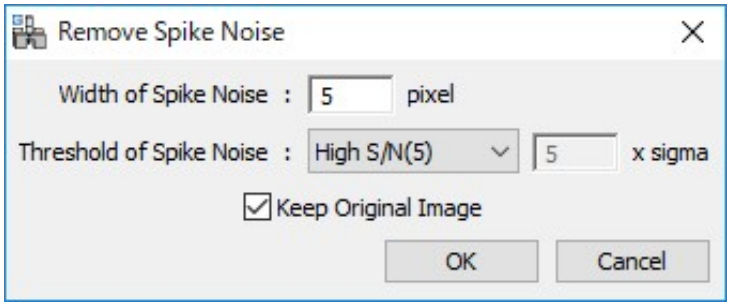
Creates a stacked image from the top most opened n-images in the image list of the window menu, or from the saved image in the selected folder, or from an opend stacked image.



Create Stacked Image Menu

Remove Spike Noise from SI

Remove a spike noise(s) from a SI data. A spike noise is defined as the data point that shows an abnormal value. Such a point is created, for example, by cosmic rays.



Remove Spike Noise Dialog

The spike noise is searched within a specified range of pixels (Width of Spike Noise). The data point that exceeds the specified noise level (Threshold of Spike Noise) is recognized as the spike noise, and replace by the local average. The number of the removed spike noise will be shown in the dialog.

Display Tags

Displays Image Tags of the front image.

Functions

Fourier Transforms

IPUFFT()
IPUIFFT()
IPURealFFT()
IPURealIFFT()
IPUPureFFT()
IPUPureIFFT()
IPUFFTX()
IPUIFFTX()
IPUFFTY()
IPUIFFTY()
IPUFFTZ()
IPUIFFTZ()
IPUFFTXY()
IPUIFFTXY()

Cross Correlations

IPUAutoCorrelation()
IPUCrossCorrelation()
IPUFTdCrossCorrelation()

Image Shifting

IPUImageShift()

Matrix

IPUMatrixMultiply()
IPUMatrixInverse()
IPULUDecomposition()

Informations

IPUGetVersion()
IPUGetVersionString()
IPUGetMKLVersion()
IPUGetMKLVersionString()

Debugging

IPUTraceEnabled()
IPUTraceInit()
IPUTraceShowTotal() IPUTrace()

VML

IPUConj()
IPUSqr()
IPUAbs()
IPUInv()
IPUSqrt()

IPUInvSqrt()
IPUCbrt()
IPUInvCbrt()
IPUPow2o3()
IPUPow3o2()
IPUExp()
IPUExpm1()
IPULn()
IPULog10() IPULog1p()
IPUCos()
IPUSin()
IPUTan()
IPUCosh()
IPUSinh()
IPUTanh() IPUAcos()
IPUAsin() IPUAtan()
IPUAcosh() IPUAsinh()
IPUAtanh()
IPUErf()
IPUErfc()
IPUErfInv()
IPUFloor()
IPUCeil()
IPUTrunc()
IPURound()
IPUNearbyInt()
IPURint() IPUAdd()
IPUSub() IPUMul()
IPUDiv()
IPUPow()
IPUHypot() IPUAtan2()
IPUMulByConj()
IPUAbs()
IPUCIS()
IPUPowx()
IPUSinCos()
IPUModf()

File List

Here is a list of all documented files with brief descriptions:

Cross_Correlations.s	
Debugging.s	
Fourier_Transforms.s	
Image_Shifting.s	
Informations.s	
Matrix.s	
VML.s	

Cross_Correlations.s File Reference

Functions

Image **IPUAutoCorrelation** (Image imgSrc)
Auto Correlation.

Image **IPUCrossCorrelation** (Image imgTarg, Image imgRef) Cross Correlation.

Image **IPUCrossCorrelation** (Image imgTarg, Image imgRef, Number eMode, Number fSize)
Cross Correlation.

Image **IPUFTdCrossCorrelation** (ComplexImage cimTarg, ComplexImage cimRef, Number eMode, Number fSize)
Cross Correlation from Fourier Transformed Images.

Detailed Description

Function Documentation

Image IPUAutoCorrelation (Image imgSrc)

Auto Correlation.

Parameters:
imgSrc Source image

Returns:
Auto correlation image

Same as AutoCorrelation()

Image IPUCrossCorrelation (Image imgTarg,
Image imgRef
)

Cross Correlation.

Parameters:
imgTarg Target image
imgRef Reference image

Returns:
Cross correlation image

Same as CrossCorrelation()

Image IPUCrossCorrelation (Image imgTarg,
Image imgRef,
Number eMode,

Number fSize
)

Cross Correlation.

Parameters:

imgTarg	Target image
imgRef	Reference image
eMode	Cross correlation mode 1..Conventional, 2..Square Root Moduli, 3..Phase
fSize	Mask size to remove high frequency noise. fSize is a radius given in pixels in Fourier space, or 0 to 0.5 in terms of Nyquist frequency.

Returns:

Cross correlation image

Image IPUFTdCrossCorrelation (ComplexImage cimTarg,
ComplexImage cimRef,
Number eMode,
Number fSize
)

Cross Correlation from Fourier Transformed Images.

Parameters:

cimTarg	Fourier transformed target image
cimRef	Fourier transformed reference image
eMode	Cross correlation mode 1..Conventional, 2..Square Root Moduli, 3..Phase
fSize	Mask size to remove high frequency noise. fSize is a radius given in pixels in Fourier space, or 0 to 0.5 in terms of Nyquist frequency.

Returns:

Cross correlation image

Debugging.s File Reference

Functions

void	**IPUTraceInit** () Initialize Trace Routine.
void	**IPUTraceShowTotal** () Show Total Execution.
void	**IPUTraceEnabled** (Number isEnabled) Set Trace Status.
Object	**IPUTrace** (String strName) Enter a Trace Block.
Object	**IPUTrace** (String strName, String strArgs) Enter a Trace Block with Optional Message.

Detailed Description

These routines show call stacks and an execution profile.

Example:

```
Number MyFunc(Number n)
{
    Object tracer = IPUTrace("MyFunc", ""+n);
    if ( n > 1 ) {
        return n + MyFunc(n - 1);
    } else
{         return
n;
    }
}

{
    IPUTraceEnabled(1);
IPUTraceInit();

    Result(MyFunc(3)+"\n");

    IPUTraceShowTotal();
}
```

This will show the following results:

```
  MyFunc(3) {
    MyFunc(2) {
      MyFunc(1) {
      } 0.346692 ms
    } 4.86766 ms
  } 7.39452 ms
6

 Total Execution:
    MyFunc                          :     3 calls,      12.6 ms
```

Function Documentation

void IPUTraceInit ( )

Initialize Trace Routine.

example

You should call this before starting a trace.

void IPUTraceShowTotal ( )

Show Total Execution.

example

This shows total execution informtion, if the trace is enabled.

void IPUTraceEnabled (Number isEnabled)

Set Trace Status.

Parameters:

isEnabled 1/0 for enable/disable.

example

Object IPUTrace (String strName)

Enter a Trace Block.

Parameters:

strName Block name to be shown

Returns:

Tracer

example

Object IPUTrace (String strName,
String strArgs
)

Enter a Trace Block with Optional Message.

Parameters:

strName Block name to be shown

strArgs Any message to be shown

Returns:
Tracer

example

Fourier_Transforms.s File Reference

Functions

ComplexImage	**IPUFFT** (ComplexImage cimSrc) Fourier Transform (Origin at Center).
ComplexImage	**IPUFFT** (ComplexImage cimSrc, ComplexImage cimDst) Fourier Transform (Origin at Center).
ComplexImage	**IPUFFT** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) Fourier Transform (Origin at Center).
ComplexImage	**IPUIFFT** (ComplexImage cimSrc) Inverse Fourier Transform (Origin at Center).
ComplexImage	**IPUIFFT** (ComplexImage cimSrc, ComplexImage cimDst) Inverse Fourier Transform (Origin at Center).
ComplexImage	**IPUIFFT** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) Inverse Fourier Transform (Origin at Center).
ComplexImage	**IPURealFFT** (Image imgSrc) Fourier Transform with RealImage.
ComplexImage	**IPURealFFT** (Image imgSrc, ComplexImage cimDst) Fourier Transform with RealImage.
ComplexImage	**IPURealFFT** (Image imgSrc, Number fScale, ComplexImage cimDst) Fourier Transform with RealImage.
Image	**IPURealIFFT** (ComplexImage cimSrc) Inverse Fourier Transform to RealImage.
Image	**IPURealIFFT** (ComplexImage cimSrc, Image imgDst) Inverse Fourier Transform to RealImage.
Image	**IPURealIFFT** (ComplexImage cimSrc, Number fScale, Image imgDst) Inverse Fourier Transform to RealImage.
ComplexImage	**IPUPureFFT** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) Fourier Transform (Origin at Top-Left).
ComplexImage	**IPUPureIFFT** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) Inverse Fourier Transform (Origin at Top-Left).
ComplexImage	**IPUFFTX** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) 1D Fourier Transform at X axis (Origin at Center)
ComplexImage	**IPUFFTX** (ComplexImage cimSrc, ComplexImage cimDst) 1D Fourier Transform at X axis (Origin at Center)
ComplexImage	**IPUFFTX** (ComplexImage cimSrc) 1D Fourier Transform at X axis (Origin at Center)
ComplexImage	**IPUIFFTX** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) 1D Inverse Fourier Transform at X axis (Origin at Center)
ComplexImage	**IPUIFFTX** (ComplexImage cimSrc, ComplexImage cimDst) 1D Inverse Fourier Transform at X axis (Origin at Center)
ComplexImage	**IPUIFFTX** (ComplexImage cimSrc)

	1D Inverse Fourier Transform at X axis (Origin at Center)
ComplexImage	**IPUFFTY** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) 1D Inverse Fourier Transform at Y axis (Origin at Center)
ComplexImage	**IPUFFTY** (ComplexImage cimSrc, ComplexImage cimDst) 1D Fourier Transform at Y axis (Origin at Center)
ComplexImage	**IPUFFTY** (ComplexImage cimSrc) 1D Fourier Transform at Y axis (Origin at Center)
ComplexImage	**IPUIFFTY** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) 1D Inverse Fourier Transform at Y axis (Origin at Center)
ComplexImage	**IPUIFFTY** (ComplexImage cimSrc, ComplexImage cimDst) 1D Inverse Fourier Transform at Y axis (Origin at Center)
ComplexImage	**IPUIFFTY** (ComplexImage cimSrc) 1D Inverse Fourier Transform at Y axis (Origin at Center)
ComplexImage	**IPUFFTZ** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) 1D Fourier Transform at Z axis (Origin at Center)
ComplexImage	**IPUFFTZ** (ComplexImage cimSrc, ComplexImage cimDst) 1D Fourier Transform at Z axis (Origin at Center)
ComplexImage	**IPUFFTZ** (ComplexImage cimSrc) 1D Fourier Transform at Z axis (Origin at Center)
ComplexImage	**IPUIFFTZ** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) 1D Inverse Fourier Transform at Z axis (Origin at Center)
ComplexImage	**IPUIFFTZ** (ComplexImage cimSrc, ComplexImage cimDst) 1D Inverse Fourier Transform at Z axis (Origin at Center)
ComplexImage	**IPUIFFTZ** (ComplexImage cimSrc) 1D Inverse Fourier Transform at Z axis (Origin at Center)
ComplexImage	**IPUFFTXY** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) 2D Fourier Transform at XY Plane (Origin at Center)
ComplexImage	**IPUFFTXY** (ComplexImage cimSrc, ComplexImage cimDst) 2D Fourier Transform at XY Plane (Origin at Center)
ComplexImage	**IPUFFTXY** (ComplexImage cimSrc) 2D Fourier Transform at XY Plane (Origin at Center)
ComplexImage	**IPUIFFTXY** (ComplexImage cimSrc, Number fScale, ComplexImage cimDst) 2D Inverse Fourier Transform at XY Plane (Origin at Center)
ComplexImage	**IPUIFFTXY** (ComplexImage cimSrc, ComplexImage cimDst) 2D Inverse Fourier Transform at XY Plane (Origin at Center)
ComplexImage	**IPUIFFTXY** (ComplexImage cimSrc) 2D Inverse Fourier Transform at XY Plane (Origin at Center)

Detailed Description

Function Documentation

ComplexImage IPUFFT (ComplexImage cimSrc)

Fourier Transform (Origin at Center).

Parameters:

cimSrc Source image

Returns:

Fourier transformed image

Same as FFT().

ComplexImage IPUFFT (ComplexImage cimSrc,
ComplexImage cimDst
)

Fourier Transform (Origin at Center).

Parameters:

cimSrc Source image
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUFFT (ComplexImage cimSrc,
Number fScale,
ComplexImage cimDst
)

Fourier Transform (Origin at Center).

Parameters:

cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

Behavior:

```
cimDst = FFT(cimSrc) * fScale
```

Normally, `fScale` is 1.0.
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUIFFT (ComplexImage cimSrc)

Inverse Fourier Transform (Origin at Center).

Parameters:

cimSrc Source image

Returns:

Inverse fourier transformed image

Same as IFFT().

ComplexImage IPUIFFT (ComplexImage cimSrc,
ComplexImage cimDst
)

Inverse Fourier Transform (Origin at Center).

Parameters:

cimSrc Source image
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUIFFT (ComplexImage cimSrc,
Number fScale,
ComplexImage cimDst
)

Inverse Fourier Transform (Origin at Center).

Parameters:

cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

Behavior:

```
cimDst = IFFT(cimSrc) * fScale
```

Normally, `fScale` is 1.0/(Number of pixels).
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPURealFFT (Image imgSrc)

Fourier Transform with RealImage.

Parameters:

imgSrc Source image

Returns:

Fourier transformed image

Same as RealFFT().

ComplexImage IPURealFFT (Image imgSrc,
ComplexImage cimDst
)

Fourier Transform with RealImage.

Parameters:

imgSrc Source image
cimDst Destination image (can be NULL)

Returns:

cimDst

If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPURealFFT (Image imgSrc,
Number fScale,
ComplexImage cimDst
)

Fourier Transform with RealImage.

Parameters:

imgSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL)

Returns:

cimDst

Warning:

Image type of `imgSrc` should be Real 4 or Real 8.

Behavior:

```
cimDst = FFT(imgSrc) * fScale
```

Normally, `fScale` is 1.0.
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

Image IPURealIFFT (ComplexImage cimSrc)

Inverse Fourier Transform to RealImage.

Parameters:

cimSrc Source image

Returns:

Inverse fourier transformed image

Same as RealIFFT().

Image IPURealIFFT (ComplexImage cimSrc,
Image imgDst
)

Inverse Fourier Transform to RealImage.

Parameters:

cimSrc Source image
imgDst Destination image (can be NULL)

Returns:

cimDst

Warning:

Image type of `imgDst` should be Real 4 or Real 8.

If `imgDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

Image IPURealIFFT (ComplexImage cimSrc,
Number fScale,
Image imgDst
)

Inverse Fourier Transform to RealImage.

Parameters:

cimSrc Source image
fScale Scaling factor
imgDst Destination image (can be NULL)

Returns:

cimDst

Warning:
Image type of `imgDst` should be Real 4 or Real 8.

Behavior:

```
cimDst = IFFT(imgSrc) * fScale
```

Normally, `fScale` is 1.0/(Number of pixels).
If `imgDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUPureFFT (ComplexImage cimSrc,
Number fScale,
ComplexImage cimDst
)

Fourier Transform (Origin at Top-Left).

Parameters:
cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:
cimDst

Behavior:

```
cimDst = FFT(cimSrc) * fScale
```

Normally, `fScale` is 1.0.
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUPureIFFT (ComplexImage cimSrc,
Number fScale,
ComplexImage cimDst
)

Inverse Fourier Transform (Origin at Top-Left).

Parameters:
cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:
cimDst

Behavior:

```
cimDst = IFFT(cimSrc) * fScale
```

Normally, `fScale` is 1.0/(Number of pixels).
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUFFTX (ComplexImage cimSrc,
Number fScale,
ComplexImage cimDst
)

1D Fourier Transform at X axis (Origin at Center)

Parameters:

cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

Normally, `fScale` is 1.0.
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUFFTX (ComplexImage cimSrc,
ComplexImage cimDst
)

1D Fourier Transform at X axis (Origin at Center)

Parameters:

cimSrc Source image
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUFFTX (ComplexImage cimSrc)

1D Fourier Transform at X axis (Origin at Center)

Parameters:

cimSrc Source image

Returns:
Fourier transformed image

ComplexImage IPUIFFTX (ComplexImage cimSrc,
Number fScale,
ComplexImage cimDst
)

1D Inverse Fourier Transform at X axis (Origin at Center)

Parameters:
cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:
cimDst

Normally, `fScale` is 1.0/(Number of pixels).
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUIFFTX (ComplexImage cimSrc,
ComplexImage cimDst
)

1D Inverse Fourier Transform at X axis (Origin at Center)

Parameters:
cimSrc Source image
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:
cimDst

If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUIFFTX (ComplexImage cimSrc)

1D Inverse Fourier Transform at X axis (Origin at Center)

Parameters:
cimSrc Source image

Returns:
Inverse Fourier transformed image

ComplexImage IPUFFTY (ComplexImage cimSrc,

Number fScale,
ComplexImage cimDst
)

1D Inverse Fourier Transform at Y axis (Origin at Center)

Parameters:

cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

Normally, `fScale` is 1.0.
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUFFTY (ComplexImage cimSrc,
ComplexImage cimDst
)

1D Fourier Transform at Y axis (Origin at Center)

Parameters:

cimSrc Source image
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUFFTY (ComplexImage cimSrc)

1D Fourier Transform at Y axis (Origin at Center)

Parameters:

cimSrc Source image

Returns:

Fourier transformed image

ComplexImage IPUIFFTY (ComplexImage cimSrc,
Number fScale,
ComplexImage cimDst
)

1D Inverse Fourier Transform at Y axis (Origin at Center)

Parameters:

cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

Normally, `fScale` is 1.0/(Number of pixels).
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUIFFTY (ComplexImage cimSrc,
ComplexImage cimDst
)

1D Inverse Fourier Transform at Y axis (Origin at Center)

Parameters:

cimSrc Source image
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUIFFTY (ComplexImage cimSrc)

1D Inverse Fourier Transform at Y axis (Origin at Center)

Parameters:

cimSrc Source image

Returns:

Inverse Fourier transformed image

ComplexImage IPUFFTZ (ComplexImage cimSrc,
Number fScale,
ComplexImage cimDst
)

1D Fourier Transform at Z axis (Origin at Center)

Parameters:

cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:
cimDst

Normally, `fScale` is 1.0.
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUFFTZ (ComplexImage cimSrc,
ComplexImage cimDst
)

1D Fourier Transform at Z axis (Origin at Center)

Parameters:
cimSrc Source image
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:
cimDst

If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUFFTZ (ComplexImage cimSrc)

1D Fourier Transform at Z axis (Origin at Center)

Parameters:
cimSrc Source image

Returns:
Fourier transformed image

ComplexImage IPUIFFTZ (ComplexImage cimSrc,
Number fScale,
ComplexImage cimDst
)

1D Inverse Fourier Transform at Z axis (Origin at Center)

Parameters:
cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:
cimDst

Normally, `fScale` is 1.0/(Number of pixels).
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUIFFTZ (ComplexImage cimSrc,
ComplexImage cimDst
)

1D Inverse Fourier Transform at Z axis (Origin at Center)

Parameters:

cimSrc Source image
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUIFFTZ (ComplexImage cimSrc)

1D Inverse Fourier Transform at Z axis (Origin at Center)

Parameters:

cimSrc Source image

Returns:

Inverse Fourier transformed image

ComplexImage IPUFFTXY (ComplexImage cimSrc,
Number fScale,
ComplexImage cimDst
)

2D Fourier Transform at XY Plane (Origin at Center)

Parameters:

cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

Normally, `fScale` is 1.0.
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUFFTXY (ComplexImage cimSrc,
ComplexImage cimDst
)

2D Fourier Transform at XY Plane (Origin at Center)

Parameters:

cimSrc Source image
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUFFTXY (ComplexImage cimSrc)

2D Fourier Transform at XY Plane (Origin at Center)

Parameters:

cimSrc Source image

Returns:

Fourier transformed image

ComplexImage IPUIFFTXY (ComplexImage cimSrc,
Number fScale,
ComplexImage cimDst
)

2D Inverse Fourier Transform at XY Plane (Origin at Center)

Parameters:

cimSrc Source image
fScale Scaling factor
cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

Normally, `fScale` is 1.0/(Number of pixels).
If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUIFFTXY (ComplexImage cimSrc,
ComplexImage cimDst
)

2D Inverse Fourier Transform at XY Plane (Origin at Center)

Parameters:

cimSrc Source image

cimDst Destination image (can be NULL, or `cimSrc` for in-place calculation)

Returns:

cimDst

If `cimDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

ComplexImage IPUIFFTXY(ComplexImage cimSrc)

2D Inverse Fourier Transform at XY Plane (Origin at Center)

Parameters:

cimSrc Source image

Returns:

Inverse fourier transformed image

Image_Shifting.s File Reference

Functions

ComplexImage **IPUImageShift** (Image imgSrc, Number fShiftX, Number fShiftY, Image imgDst)
Translate Image using FFT.

Detailed Description

Function Documentation

ComplexImage IPUImageShift(Image imgSrc,
Number fShiftX,
Number fShiftY,
Image imgDst
)

Translate Image using FFT.

Parameters:

imgSrc	Source image
fShiftX	X shift
fShiftY	Y shift
imgDst	Destination image (can be NULL, or `imgSrc` for in-place calculation)

Returns:
imgDst

This function shifts an image using the shift theorem of Fourier Transform. Thus, you can specify sub-pixel numbers to `fShiftX` and `fShiftY`.
If `imgDst` is allocated, this function put a result to it. Otherwise, this function allocates a new destination area and returns the result on it.

Example:

```
Image img := GetFrontImage().ImageClone();

img.ConvertToFloat();

// translate (X, Y) = (10.5, 3)
IPUImageShift(img, 10.5, 3, img);

img.ShowImage();
```

Informations.s File Reference

Number	**IPUGetVersion** Get Version Number.
String	**IPUGetVersionString** Get Version String.
Number	**IPUGetMKLVersion** Get MKL Version Number.
String	**IPUGetMKLVersionString** Get MKL Version String.

Detailed Description

Function Documentation

Number IPUGetVersion ( )

Get Version Number.

Returns:
Version number

It returns 0xXXYYZZWW.

XX: Major Version
YY: Minor Version
ZZ: Revision Number
WW: Build Number

String IPUGetVersionString ( )

Get Version String.

Returns:
Version string

Number IPUGetMKLVersion ( )

Get MKL Version Number.

It returns 0xXXYYZZZZ.

XX: Major Version
YY: Minor Version
ZZZZ: Build Number

String IPUGetMKLVersionString ( )

Get MKL Version String.

Returns:
Version string

Matrix.s File Reference

Image	**IPUMatrixMultiply** (Image imgA, Image imgB) Matrix Multiply.
Image	**IPUMatrixInverse** (Image imgA) Matrix Inverse.
Image	**IPULUDecomposition** (Image imgA, Image imgB) LUDecomposition.
Image	**IPUMatrixTranspose** (Image imgA) Matrix Transpose.

Detailed Description

Function Documentation

Image IPUMatrixMultiply (Image imgA,
Image imgB
)

Matrix Multiply.

Parameters:
imgA 2D image
imgB 2D image

Returns:
Matrix Multiplied 2D image

Same as MatrixMultiply()

Image IPUMatrixInverse (Image imgA)

Matrix Inverse.

Parameters:
imgA 2D image

Returns:
Inversed Matrix (2D image)

Same as MatrixInverse()

Image IPULUDecomposition (Image imgA,
Image imgB
)

LUDecomposition.

Parameters:

imgA 2D image

imgB 2D image

Returns:

Solution X (2D image) of AX = B

Image IPUMatrixTranspose(Image imgA)

Matrix Transpose.

Parameters:

imgA 2D image

Returns:

Transposed Matrix (2D image)

Same as MatrixTranspose()

VML.s File Reference

Image **IPUConj** (ComplexImage imgSrc, ComplexImage imgDst)

Conjugation of vector elements.

Image **IPUSqr** (Image imgSrc, Image imgDst) Squaring of vector elements.

Image **IPUAbs** (Image imgSrc, Image imgDst) Absolute value of vector elements.

Image **IPUInv** (Image imgSrc, Image imgDst) Inversion of vector elements.

Image **IPUSqrt** (Image imgSrc, Image imgDst)

Square root of vector elements.

Image **IPUInvSqrt** (Image imgSrc, Image imgDst)

Inverse square root of vector elements.

Image **IPUCbrt** (Image imgSrc, Image imgDst)

Cube root of vector elements.

Image **IPUInvCbrt** (Image imgSrc, Image imgDst)

Inverse cube root of vector elements.

Image **IPUPow2o3** (Image imgSrc, Image imgDst)

Each vector element raised to 2/3.

Image **IPUPow3o2** (Image imgSrc, Image imgDst)

Each vector element raised to 3/2.

Image **IPUExp** (Image imgSrc, Image imgDst) Exponential of vector elements.

Image **IPUExpm1** (Image imgSrc, Image imgDst)

Exponential of vector elements decreased by 1.

Image **IPULn** (Image imgSrc, Image imgDst)

Natural logarithm of vector elements.

Image **IPULog10** (Image imgSrc, Image imgDst)

Denary logarithm of vector elements.

Image **IPULog1p** (Image imgSrc, Image imgDst)
Natural logarithm of vector elements that are increased by 1.

Image **IPUCos** (Image imgSrc, Image imgDst)

Cosine of vector elements.

Image **IPUSin** (Image imgSrc, Image imgDst)

Sine of vector elements.

Image **IPUTan** (Image imgSrc, Image imgDst)

Tangent of vector elements.

Image **IPUCosh** (Image imgSrc, Image imgDst)

Hyperbolic cosine of vector elements.

Image **IPUSinh** (Image imgSrc, Image imgDst)

Hyperbolic sine of vector elements.

Image **IPUTanh** (Image imgSrc, Image imgDst)

Hyperbolic tangent of vector elements.

Image **IPUAcos** (Image imgSrc, Image imgDst)

Inverse cosine of vector elements.

Image **IPUAsin** (Image imgSrc, Image imgDst)

Inverse sine of vector elements.

Image **IPUAtan** (Image imgSrc, Image imgDst)

Inverse tangent of vector elements.

Image **IPUAcosh** (Image imgSrc, Image imgDst)

Inverse hyperbolic cosine of vector elements.

Image **IPUAsinh** (Image imgSrc, Image imgDst)

Inverse hyperbolic sine of vector elements.

Image **IPUAtanh** (Image imgSrc, Image imgDst)

Inverse hyperbolic tangent of vector elements.

Image **IPUErf** (Image imgSrc, Image imgDst)

Error function value of vector elements.

Image **IPUErfc** (Image imgSrc, Image imgDst)
Complementary error function value of vector elements.

Image **IPUErfInv** (Image imgSrc, Image imgDst)

Inverse error function value of vector elements.

Image **IPUFloor** (Image imgSrc, Image imgDst)

Rounding towards minus infinity. Image **IPUCeil** (Image imgSrc, Image imgDst)

Rounding towards plus infinity.

Image **IPUTrunc** (Image imgSrc, Image imgDst)

Rounding towards zero infinity.

Image **IPURound** (Image imgSrc, Image imgDst)

Rounding to nearest integer.

Image **IPUNearbyInt** (Image imgSrc, Image imgDst)

Rounding according to current mode.

Image **IPURint** (Image imgSrc, Image imgDst)
Rounding according to current mode and raising inexact result exception.

Image **IPUAdd** (Image imgA, Image imgB, Image imgDst)

Addition of vector elements.

Image **IPUSub** (Image imgA, Image imgB, Image imgDst)

Subtraction of vector elements.

Image **IPUMul** (Image imgA, Image imgB, Image imgDst)

Multiplication of vector elements.

Image **IPUDiv** (Image imgA, Image imgB, Image imgDst)
Division of elements of one vector by elements of the second vector.

Image **IPUPow** (Image imgA, Image imgB, Image imgDst)

Each vector element raised to the specified power.

Image **IPUHypot** (Image imgA, Image imgB, Image imgDst)

Square root of sum of squares.

Image **IPUAtan2** (Image imgA, Image imgB, Image imgDst)
Four-quadrant inverse tangent of elements of two vectors.

Image **IPUMulByConj** (ComplexImage imgA, ComplexImage imgB, ComplexImage imgDst)

Multiplication of elements of one vector by conjugated elements of the second vector.

Image **IPUAbs** (ComplexImage imgSrc, Image imgDst)

Absolute value of vector elements.

Image **IPUCIS** (Image imgSrc, ComplexImage imgDst)
Complex exponent of vector elements(cosine and sine combined to complex value).

Image **IPUPowx** (Image imgSrc, Number nB, Image imgDst)

vector elements raised to the constant power

void **IPUSinCos** (Image imgSrc, Image imgSin, Image imgCos)

Sine and cosine of vector elements.

void **IPUModf** (Image imgSrc, Image imgVal, Image imgRem)
Integer and fraction parts.

Detailed Description

Function Documentation

Image IPUConj (ComplexImage imgSrc,
ComplexImage imgDst
)

Conjugation of vector elements.

Parameters:

imgSrc Source image
imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUSqr (Image imgSrc,
Image imgDst
)

Squaring of vector elements.

Parameters:

imgSrc Source image
imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUAbs (Image imgSrc,
Image imgDst

)

Absolute value of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUInv (Image imgSrc,
Image imgDst
)

Inversion of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUSqrt (Image imgSrc,
Image imgDst
)

Square root of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUInvSqrt (Image imgSrc,
Image imgDst
)

Inverse square root of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUCbrt (Image imgSrc,
Image imgDst
)

Cube root of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUInvCbrt (Image imgSrc,
Image imgDst
)

Inverse cube root of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUPow2o3 (Image imgSrc,
Image imgDst
)

Each vector element raised to 2/3.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUPow3o2 (Image imgSrc,
Image imgDst
)

Each vector element raised to 3/2.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:
imgDst

Image IPUExp (Image imgSrc,
Image imgDst
)

Exponential of vector elements.

Parameters:
imgSrc Source image
imgDst Destination image (can be NULL)

Returns:
imgDst

Image IPUExpm1 (Image imgSrc,
Image imgDst
)

Exponential of vector elements decreased by 1.

Parameters:
imgSrc Source image
imgDst Destination image (can be NULL)

Returns:
imgDst

Image IPULn (Image imgSrc,
Image imgDst
)

Natural logarithm of vector elements.

Parameters:
imgSrc Source image
imgDst Destination image (can be NULL)

Returns:
imgDst

Image IPULog10 (Image imgSrc,
Image imgDst
)

Denary logarithm of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPULog1p (Image imgSrc,
Image imgDst
)

Natural logarithm of vector elements that are increased by 1.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUCos (Image imgSrc,
Image imgDst
)

Cosine of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUSin (Image imgSrc,
Image imgDst
)

Sine of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUTan (Image imgSrc,
Image imgDst

)

Tangent of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUCosh (Image imgSrc,
Image imgDst
)

Hyperbolic cosine of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUSinh (Image imgSrc,
Image imgDst
)

Hyperbolic sine of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUTanh (Image imgSrc,
Image imgDst
)

Hyperbolic tangent of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUAcos (Image imgSrc,
Image imgDst
)

Inverse cosine of vector elements.

Parameters:

imgSrc Source image
imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUAsin (Image imgSrc,
Image imgDst
)

Inverse sine of vector elements.

Parameters:

imgSrc Source image
imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUAtan (Image imgSrc,
Image imgDst
)

Inverse tangent of vector elements.

Parameters:

imgSrc Source image
imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUAcosh (Image imgSrc,
Image imgDst
)

Inverse hyperbolic cosine of vector elements.

Parameters:

imgSrc Source image
imgDst Destination image (can be NULL)

Returns:
imgDst

Image IPUAsinh (Image imgSrc,
Image imgDst
)

Inverse hyperbolic sine of vector elements.

Parameters:
imgSrc Source image
imgDst Destination image (can be NULL)

Returns:
imgDst

Image IPUAtanh (Image imgSrc,
Image imgDst
)

Inverse hyperbolic tangent of vector elements.

Parameters:
imgSrc Source image
imgDst Destination image (can be NULL)

Returns:
imgDst

Image IPUErf (Image imgSrc,
Image imgDst
)

Error function value of vector elements.

Parameters:
imgSrc Source image
imgDst Destination image (can be NULL)

Returns:
imgDst

Image IPUErfc (Image imgSrc,
Image imgDst
)

Complementary error function value of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUErfInv (Image imgSrc,
Image imgDst
)

Inverse error function value of vector elements.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUFloor (Image imgSrc,
Image imgDst
)

Rounding towards minus infinity.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUCeil (Image imgSrc,
Image imgDst
)

Rounding towards plus infinity.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUTrunc (Image imgSrc,
Image imgDst

)

Rounding towards zero infinity.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPURound (Image imgSrc,
Image imgDst
)

Rounding to nearest integer.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUNearbyInt (Image imgSrc,
Image imgDst
)

Rounding according to current mode.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPURint (Image imgSrc,
Image imgDst
)

Rounding according to current mode and raising inexact result exception.

Parameters:

imgSrc Source image

imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUAdd (Image imgA,
Image imgB,
Image imgDst
)

Addition of vector elements.

Parameters:

imgA an image
imgB an image
imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUSub (Image imgA,
Image imgB,
Image imgDst
)

Subtraction of vector elements.

Parameters:

imgA an image
imgB an image
imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUMul (Image imgA,
Image imgB,
Image imgDst
)

Multiplication of vector elements.

Parameters:

imgA an image
imgB an image
imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUDiv (Image imgA,
Image imgB,

```
                Image  imgDst
              )
```

Division of elements of one vector by elements of the second vector.

Parameters:

imgA an image
imgB an image
imgDst Destination image (can be NULL)

Returns:

imgDst

```
Image IPUPow ( Image  imgA,
               Image  imgB,
               Image  imgDst
             )
```

Each vector element raised to the specified power.

Parameters:

imgA an image
imgB an image
imgDst Destination image (can be NULL)

Returns:

imgDst

```
Image IPUHypot ( Image  imgA,
                 Image  imgB,
                 Image  imgDst
               )
```

Square root of sum of squares.

Parameters:

imgA an image
imgB an image
imgDst Destination image (can be NULL)

Returns:

imgDst

```
Image IPUAtan2 ( Image  imgA,
                 Image  imgB,
                 Image  imgDst
               )
```

Four-quadrant inverse tangent of elements of two vectors.

Parameters:

imgA an image
imgB an image
imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUMulByConj (ComplexImage imgA,
ComplexImage imgB,
ComplexImage imgDst
)

Multiplication of elements of one vector by conjugated elements of the second vector.

Parameters:

imgA an image
imgB an image
imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUAbs (ComplexImage imgSrc,
Image imgDst
)

Absolute value of vector elements.

Parameters:

imgSrc Source image
imgDst Destination image (can be NULL)

Returns:

imgDst

Image IPUCIS (Image imgSrc,
ComplexImage imgDst
)

Complex exponent of vector elements(cosine and sine combined to complex value).

Parameters:

imgSrc Source image
imgDst Destination image (can be NULL)

Returns:
imgDst

Image IPUPowx (Image imgSrc,
Number nB,
Image imgDst
)

vector elements raised to the constant power

Parameters:

imgSrc	Source image
nb	Constant value for power b.
imgDst	Destination image (can be NULL)

Returns:
imgDst

void IPUSinCos (Image imgSrc,
Image imgSin,
Image imgCos
)

Sine and cosine of vector elements.

Parameters:

imgSrc	Source image
imgSin	Sine values
imgCos	Cosine values

void IPUModf (Image imgSrc,
Image imgVal,
Image imgRem
)

Integer and fraction parts.

Parameters:

imgSrc	Source image
imgVal	Integer values
imgRem	Fraction values

Here is a list of all documented functions, variables, defines, enums, and typedefs with links to the documentation:

- i -

- IPUAbs() : **VML.s**
- IPUAcos() : **VML.s**
- IPUAcosh() : **VML.s**
- IPUAdd() : **VML.s**
- IPUAsin() : **VML.s**
- IPUAsinh() : **VML.s**
- IPUAtan() : **VML.s**
- IPUAtan2() : **VML.s** IPUAtanh() :
- **VML.s**
- IPUAutoCorrelation() : **Cross_Correlations.s**
- IPUCbrt() : **VML.s**
- IPUCeil() : **VML.s**
- IPUCIS() : **VML.s**
- IPUConj() : **VML.s**
- IPUCos() : **VML.s**
- IPUCosh() : **VML.s**
- IPUCrossCorrelation() : **Cross_Correlations.s**
- IPUDiv() : **VML.s**
- IPUErf() : **VML.s**
- IPUErfc() : **VML.s**
- IPUErfInv() : **VML.s**
- IPUExp() : **VML.s**
- IPUExpm1() : **VML.s**
- IPUFFT() : **Fourier_Transforms.s**
- IPUFFTX() : **Fourier_Transforms.s**
- IPUFFTXY() : **Fourier_Transforms.s**
- IPUFFTY() : **Fourier_Transforms.s** IPUFFTZ() :
- **Fourier_Transforms.s**
- IPUFloor() : **VML.s**
- IPUFTdCrossCorrelation() : **Cross_Correlations.s**
- IPUGetMKLVersion() : **Informations.s**
- IPUGetMKLVersionString() : **Informations.s**
- IPUGetVersion() : **Informations.s**
- IPUGetVersionString() : **Informations.s**
- IPUHypot() : **VML.s**
- IPUIFFT() : **Fourier_Transforms.s**
- IPUIFFTX() : **Fourier_Transforms.s**
- IPUIFFTXY() : **Fourier_Transforms.s**
- IPUIFFTY() : **Fourier_Transforms.s** IPUIFFTZ() :
- **Fourier_Transforms.s**
- IPUImageShift() : **Image_Shifting.s**
- IPUInv() : **VML.s**
- IPUInvCbrt() : **VML.s**
- IPUInvSqrt() : **VML.s**
- IPULn() : **VML.s**

IPULog10() : **VML.s**

- IPULog1p() : **VML.s**
- IPULUDecomposition() : **Matrix.s**
- IPUMatrixInverse() : **Matrix.s**
- IPUMatrixMultiply() : **Matrix.s**
- IPUMatrixTranspose() : **Matrix.s**
- IPUModf() : **VML.s**
- IPUMul() : **VML.s**
- IPUMulByConj() : **VML.s**
- IPUNearbyInt() : **VML.s**
- IPUPow() : **VML.s**
- IPUPow2o3() : **VML.s**
- IPUPow3o2() : **VML.s**
- IPUPowx() : **VML.s**
- IPUPureFFT() : **Fourier_Transforms.s**
- IPUPureIFFT() : **Fourier_Transforms.s**
- IPURealFFT() : **Fourier_Transforms.s**
- IPURealIFFT() : **Fourier_Transforms.s**
- IPURint() : **VML.s**
- IPURound() : **VML.s**
- IPUSin() : **VML.s**
- IPUSinCos() : **VML.s**
- IPUSinh() : **VML.s**
- IPUSqr() : **VML.s**
- IPUSqrt() : **VML.s**
- IPUSub() : **VML.s**
- IPUTan() : **VML.s**
- IPUTanh() : **VML.s**
- IPUTrace() : **Debugging.s**
- IPUTraceEnabled() : **Debugging.s**
- IPUTraceInit() : **Debugging.s**
- IPUTraceShowTotal() : **Debugging.s**
- IPUTrunc() : **VML.s**
