VSG Technical Report
OSMIA- D4.5(i)

Project	OSMIA: Open Source Medical Image Analysis EU Fifth Framework Programme (IST: Accompanying Measures)
Project number	IST-2001-34512
Deliverable number	D4.5(i)
Revision	2
Report Title	Integration of Tina into NeatVision: Developers Guide
Prepared by	Dr. Ovidiu Ghita, Prof. Paul F Whelan
Distribution List	All
Date	September 2003

Contents

1. Introduction

Tina 5 software (www.tina-vision.net) has been originally designed to provide a software environment for users involved in the development of machine vision and medical applications. Tina libraries are written in C and include mathematical, standard image processing and vision algorithms. NeatVision 2.1 (www.neatvision.com) is a user-friendly Java package that has been designed to facilitate teaching and develop image processing related applications. NeatVision can be easily extended by building on previously developed algorithms that are available as a collection of components. The mechanism to reuse NeatVision's internal components has been described in detail in the Developers Guide that is available for download from NeatVision's official website (www.neatvision.com). This document outlines the procedure that allows the developers to use Tina libraries in the development of new image processing solutions under NeatVision 2.1 graphical environment.

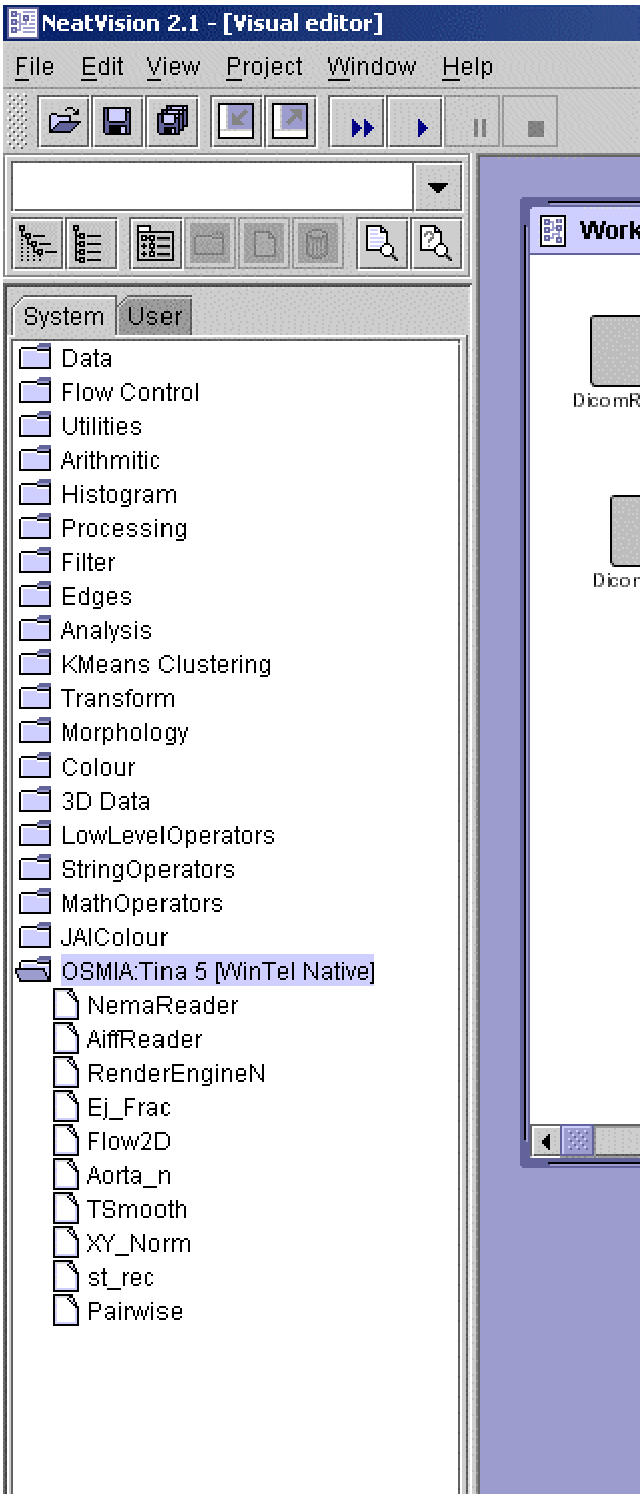
2. Installation

Information regarding installation of NeatVision can be obtained by accessing the following website: www.neatvision.com. After NeatVision has been successfully installed, the user can download the software required to interface Tina libraries and NeatVision 2.1. The software can be downloaded from DCU OSMIA website (http://www.eeng.dcu.ie/~whelanp/osmia/) and is distributed as a jar file (**tina.jar**) and a dll library (**tina.dll**). This website also contains the NeatVision / Tina software and documentation, NeatVision / Tina developers guide, sample applications along with an online java doc site. Currently only the users that have installed NeatVision 2.1 on Windows NT, Windows 2000 or Windows XP can use Tina in the development of new image processing components. In the future we plan to provide sharable libraries for Unix platforms such as Sun Solaris and Linux. The tina.jar and tina.dll dynamic library files have to be placed in the directory where the NeatVision software resides or in a directory that have been included in the classpath. NeatVision can be run by typing the following command (ignoring the spaces between new lines):

```
c:\jdk1.3.1_06\jre\bin\java.exe -classpath c:\NeatVision2.1\neatvision.jar;
c:\NeatVision2.1\developer.jar;c:\NeatVision2.1\tina.jar  NeatVision
```

The example provided above assumes that jdk1.3.1_06 is installed on your machine. For higher version should be specified the appropriate path for java.exe. At this stage we can start to use NeatVision and develop new software components using the information provided in NeatVision Developers Guide[1]. The remainder of this document will outline the standard procedures to exchange data between NeatVision and Tina and to describe how to access methods contained in Tina library. Also a number of examples will be provided.

[1] ftp://ftp.eeng.dcu.ie/pub/neatvis/downloads/release_2_1/NV_DevelopersGuide_2_1.pdf


NeatVision 2.1 - [Visual editor]
File Edit View Project Window Help
System User
Work
DicomR
Dicor

Data
Flow Control
Utilities
Arithmitic
Histogram
Processing
Filter
Edges
Analysis
KMeans Clustering
Transform
Morphology
Colour
3D Data
LowLevelOperators
StringOperators
MathOperators
JAIColour
OSMIA:Tina 5 [WinTel Native]
NemaReader
AiffReader
RenderEngineN
Ej_Frac
Flow2D
Aorta_n
TSmooth
XY_Norm
st_rec
Pairwise

NeatVision 2.1 / Tina integration

3. Using Tina libraries in NeatVision

In order to facilitate an easy access to Tina libraries from Java, a proxy class for each Tina data structure should be created. The generation of proxy classes can be performed manually using the strategy depicted at the following location: http://www.cs.up.ac.za/polelo/pieter_jni.html or using a software tool such as SWIG (http://www.swig.org), which is able to generate the interface code automatically. The steps required to build the native dll dynamic library and generate the proxy classes have been detailed in the technical report D4.2 (available for download from OSMIA website) and can be summarised as follows:

- Writing the SWIG *i* interface file
- Run the *swig* command to generate the proxy classes and lump the SWIG generated proxy classes into a jar file by using the *jar* tool that is provided as part of the jdk package.

It is important to note that SWIG creates a number of internal classes (SWIGTYPES) to handle multiple indirected pointers. These Java classes can be used by the user only in a limited number of cases but in many situations they limit the level of access to Tina libraries from Java. Fortunately, only a small number of Tina methods cannot be exposed to Java and in the future we intend to include a wrapper layer in Tina to hide multiple pointer indirection.

Tina.jar contains a large number of proxy classes that are associated with Tina data structures (e.g, *imregion* has a proxy class called *jinaImregion*) and a class named *tina* that assures the access to Tina functions in Java. As this explanation may appear a bit abstract to users that are not familiar with the JNI mechanism for interfacing C/C++ in Java we can get a better understanding by evaluating the mechanism for interfacing a Tina image data structure in Java, namely *imrect*.

Before we detail the mechanism to use *imrects*, it is important to mention that NeatVision uses the Java image class for image storage. But as the pixels are restricted to BYTE type data (for each color component) this fact limits its use only to data transfer between components or more often for displaying purposes. For other data representations the user can create either a data buffer where the data type has the required resolution (e.g. int [ ][ ], float [ ][ ], etc.) or a user defined class.

Imrect is a Tina data structure that assures the generic image storage and specifies the image type and dimensions and its region of interest that is specified by another data structure named *imregion*. In order to access these data structures in Java two proxy classes named *jinaImrect* and *jinaImregion* have been generated. In Tina the allocation of the image buffer is realised by the *im_alloc* function and the access to the image buffer for setting and getting a pixel value is achieved by functions such as *im_put_pix*, *im_put_pixf*, *im_get_pix*, *im_get_pixf* and their interface functions are included in *tina* class. Thus, *jinaImrect* is a Java interface object that assures the link with the native code that exists in a dynamic (dll) library. Hence, this object does not have allocated data space in Java. It just assures the interface to data that exists in the C side.

There are two possible scenarios to use the *imrect*s in Java. The first is to create imrects from Java using the *im_alloc* function while the second is to obtain the handle of the *imrec*ts that have been created in C. These situations will be exemplified below.

```
//create imrect from Java, transfer data and apply a simple a
//smoothing operator available in Tina library

int Width = 256;
int Height =256; //image dimensions
int x,y;

float buffer = new float [Width*Height]; //suppose is a Java buffer
                                        //store a 256*256 float image

//create a ROI
jinaImregion  reg = tina.roi_alloc(0, 0, Width, Height);

//create an imrect and allocate the space
jinaImrect im1 = tina.im_alloc(Height, Width, reg, tina.float_v);


//pass the buffer containing the data to im1
for(y=0;y<Height;y++)
{
    for(x=0;x<Width;x++)
    {
        //tina function to set a pixel value in an imrect
        tina.im_put_pixf(buffer[x+y*Width], im1, y, x);
    }
}

//apply the tangential smooth operation
im1 = tina.im_tsmooth(im1);

//write the output image in pgm format
tina.pgm_write_image(im1, "image.pgm");

//free the space that has been allocated for imrect
tina.im_free(im1);
```

The second scenario is to use the interface to the *imrects* that have already been created in C by a Tina method.

```
//create a 256*256 chequered image in Tina and transfer the image to
//a Java GrayImage

int Width =256;  //image dimensions
int Height = 256;
int x,y;

//obtain the handle of the C allocated imrect
jinaImrect im2 = tina.imf_checquer(Height, Width, 16, 16);

//create a NeatVision GrayImage
GrayImage output = new GrayImage(Width,Height);

for(y=0;y<Height;y++)
      for(x=0;x<Width;x++)
             output.setxy(x,y, (int) tina.im_get_pixf(im2, y, x));
```

```
//write the output image in pgm format
tina.pgm_write_image(im2, "im_cheq.pgm");

//free the space that has been allocated for imrect
tina.im_free(im2);
```

Although the code is executed in Java it can be noted that the allocation and de-allocation of memory space should be performed by the user. In the first example described above the user allocate the space for an *imrect* in C from Java trough the JNI interface and the user should free it when the object is not needed. In the second example the user also has to de-allocate the memory space that has been allocated by the *imf_chequer* function. More details about using Tina library in NeatVision can be found in the example components depicted in Appendix A. Documentation regarding Tina methods listed in Appendix B can be found by accessing the following website: www.tina-vision.net. The source code of the java classes can be browsed on-line at following location: http://www.tina-vision.net/lxr-cgi-bin/source/tina-libs/swig/java.

References:

- Tina 5 webpage: http://www.tina-vision.net
- NeatVision webpage: http://www.neatvision.com
- OSMIA report D.4.1:Integration of Tina 5 libraries and the NeatVision Development Environment
- OSMIA report D.4.2:Integration of Tina C libraries and the NeatVision 2.0 Development Environment
- Java webpage: http://java.sun.com
- Using peer classes with JNI: http://www.cs.up.ac.za/polelo/pieter_jni.html
- SWIG webpage: http://www.swig.org
- VSG OSMIA project webpage - tina.jar, tina.dll and associated documentation and software links: http://www.eeng.dcu.ie/~whelanp/osmia/
- P.F. Whelan and D. Molloy (2000), Machine Vision Algorithms in Java: Techniques and Implementation, Springer (London), 298 Pages. ISBN 1-85233-218-2.

OSMIA – Tina 5 Interface[2]			
NemaReader	Read a *Nema* image	Path of the file through the graphical interface	0: GrayImage 1: Double [minimum pixel value] 2: Double [maximum pixel value]
AiffReader	Read an *Aiff* image	Path of the file through the graphical interface	0: GrayImage 1: Double [minimum pixel value] 2: Double [maximum pixel value]
RenderEngineN	Render a binary volume image	0: VolumeImage [binary] 1: Integer [Thickness factor: Values: 1 to n]	
Ej_Frac	Compute the ejection fraction from two volume images.	0: VolumeImage [systole] 1: VolumeImage [diastole]	0: VolumeImage [binary] 1: VolumeImage [binary] 2: Double [Ejection fraction value]
Flow2D	Compute the 2D optical flow (Horn-Schunck or Lucas-Kanade). Original code by Prof. John Barron, UWO, Canada	0: String [Directory path for optical flow RAS sequence] 1: String [Stem name of the sequence] 2: Boolean [Swap data: PC should be TRUE] 3: Boolean [Method: TRUE : Horn-Schunck. FALSE: Lucas-Kanade] 4: Integer [Flow number: middle index of the sequence] 5: Double [Tau parameter, Default value: 0.5] 6: Double [Alpha parameter, Default value: 1.0] 7: Integer [Number of iterations, Default value: 50] 8: Integer [Offset parameter, Default value: 6] 9: Double [Scale parameter, Default value 12.0]	0: GrayImage [Output image illustrating the flow vectors]
Aorta_n	Detect the aorta outline in a greyscale image.	0: GrayImage [Input image] 1: String [Path of the model data] 2: String [Path of the pca_model data] 3: Double [minimum pixel value] 4: Double [maximum pixel value]	0: Image [RGB image highlighting the aorta outline]
TSmooth	Tangential smooth operator	0: GrayImage [Input image] 1: Integer [Number of iterations]	0: GrayImage [Output image]
XY_Norm	Coil correction algorithm	0: GrayImage [Input image] 1: Double [Standard deviation]	0: GrayImage [Output image]

[2] See http://www.eeng.dcu.ie/~whelanp/osmia/ for details on interfacing NeatVision with Tina 5.0

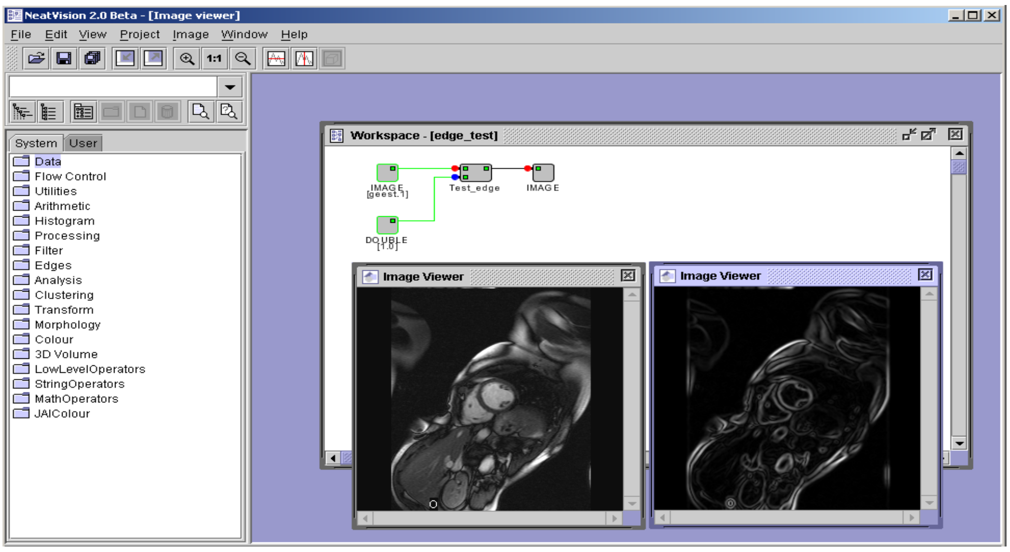
st_rec	Stereo rectification algorithm	0: String [Path of left image – aiff format] 1: String [Path of right image – aiff format] 2: String [Path of left cam file] 3: String [Path of right cam file]	0: GrayImage [Input left image] 1: GrayImage [Input right image] 2: GrayImage [Output left image] 3: GrayImage [Output right image]
Pairwise	Pairwise geometric histograms 2D object recognition	0: String [Directory path] 1: String [Filename – scene data] 2: String [Filename – model data]	0: Image [RGB image: Scene data] 1: Image [RGB image: Model data] 2: Image [RGB image: Recognised model superimposed on the input scene data]

Sample entry for Tina 5 functions fron the NeatVision developer's document. The full document can be downloaded from http://neatvision.eeng.dcu.ie/developer.html

Appendix A

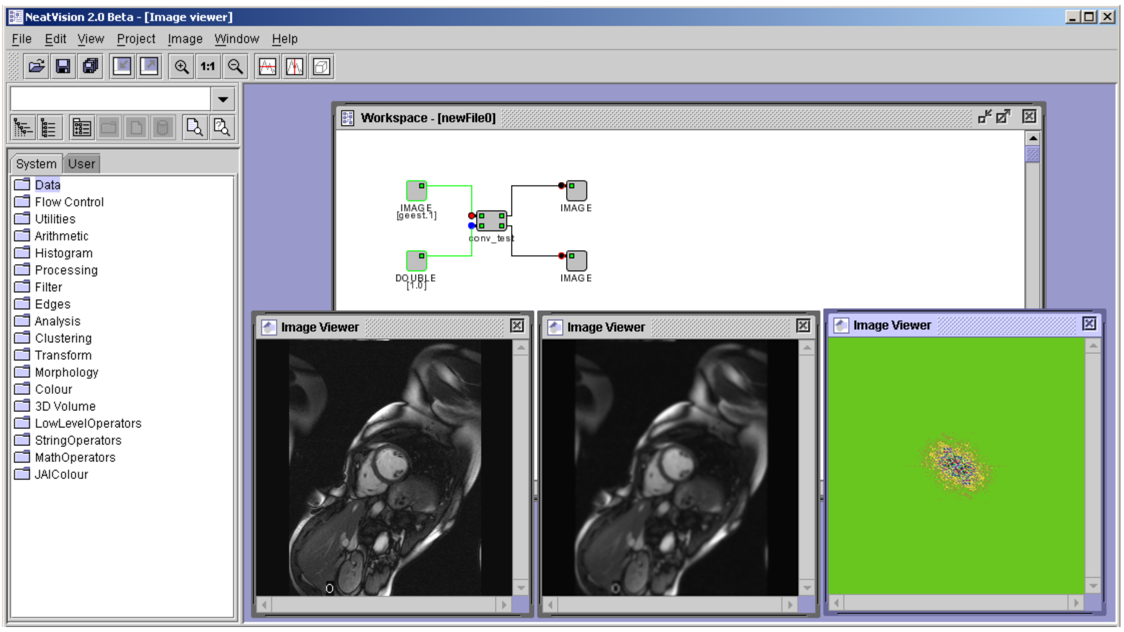
Appendix A1: Pseudo-LoG edge operator

```
//Project Name:         NeatVision 2.1
//Written by:           Ovidiu Ghita, VSG, DCU
//Initial Version:      Thursday, 11/09/03
//Description: Sample file to illustrate Tina integration in NV 2.1

import DataBlock;
import CoreInterface;

public class test extends CoreInterface
{
      public test()
      {
            name = "Test_edge";
            inputs = 2;
            outputs = 1;
            width = 30;
            height = 20;
      }

      public void setup()
      {
            Input[0].setConnectionType(IMAGE);
            Input[0].setConnectionMode(NORMAL);
            Input[0].shortDescription = "";
            Input[0].setConnectionDescription(new String[]{
                  "Input: grayscale image"      });

        Input[1].setConnectionType(DOUBLE);
            Input[1].setConnectionMode(NORMAL);
            Input[1].shortDescription = "";
            Input[1].setConnectionDescription(new String[]{
                  "Standard deviation"    });

            Output[0].setConnectionType(IMAGE);
            Output[0].setConnectionMode(NORMAL);
            Output[0].shortDescription = "";
            Output[0].setConnectionDescription(new String[]{
                  "Output: Grayscale edge information"      });

      }

      public void doubleClick()
      {

      }

      public Object create(DataBlock args)
      {
        GrayImage input = (GrayImage)args.getGrayImage(0);
        double sigma = args.getDouble(1);

        int Width = input.getWidth();
        int Height = input.getHeight();
        int i,j;
        double precision =0.01;

        System.out.println("Run Pseudo-LoG edge detector");
        //create imrect
        jinaImregion  reg = tina.roi_alloc(0, 0, Width, Height);
        jinaImrect inp_im = tina.im_alloc(Height, Width,
                              reg, tina.float_v);

        for(j=0;j<Height;j++)
            for(i=0;i<Width;i++)
                  tina.im_put_pixf(input.getxy(i,j), inp_im, j, i);

        //apply Gaussian smoothing
        jinaImrect gauss_im = tina.imf_gauss(inp_im, sigma,
                              precision);

        //free imrect
        tina.im_free(inp_im);

        //x and y derivatives
        jinaImrect gradx_im = tina.imf_grad_h(gauss_im);
        jinaImrect grady_im = tina.imf_grad_v(gauss_im);
        tina.im_free(gauss_im);
        jinaImrect gradsq = tina.imf_sumsq(gradx_im, grady_im);

        gradsq = tina.im_sqrt(gradsq);

        //normalise pixels values to 0-255
        gradsq = tina.imf_scale(gradsq, 0.0, 255.0);

        //free imrects
        tina.im_free(gradx_im);
        tina.im_free(grady_im);

        GrayImage output = new GrayImage(Width,Height);

        for(j=0;j<Height;j++)
            for(i=0;i<Width;i++)
               output.setxy(i,j,(int)tina.im_get_pixf(gradsq, j, i));

        //free imrect
        tina.im_free(gradsq);
        return output;
      }
}
```



Appendix A2: Convolution and Fourier power spectrum calculation

```
//Project Name:         NeatVision 2.1
//Written by:           Ovidiu Ghita, VSG, DCU
//Initial Version:      Thursday, 11/09/03
//Description: Sample file to illustrate Tina integration in NV 2.1
//Convolve the input image with a Gaussian kernel and compute the
//power spectrum of the convolved image

import DataBlock;
import CoreInterface;

public class conv_test extends CoreInterface
{
      public conv_test()
      {
            name = "conv_test";
            inputs = 2;
            outputs = 2;
            width = 30;
            height = 20;
      }

      public void setup()
      {
            Input[0].setConnectionType(UNDEFINED);
            Input[0].setConnectionMode(NORMAL);
            Input[0].shortDescription = "";
            Input[0].setConnectionDescription(new String[]{
                  "Input grayscale image" });

            Input[1].setConnectionType(DOUBLE);
            Input[1].setConnectionMode(NORMAL);
            Input[1].shortDescription = "";
            Input[1].setConnectionDescription(new String[]{
                  "Standard deviation"    });

            Output[0].setConnectionType(UNDEFINED);
            Output[0].setConnectionMode(NORMAL);
            Output[0].shortDescription = "";
            Output[0].setConnectionDescription(new String[]{
                  "Output grayscale image - convolution result"   });

            Output[1].setConnectionType(UNDEFINED);
            Output[1].setConnectionMode(NORMAL);
            Output[1].shortDescription = "";
            Output[1].setConnectionDescription(new String[]{
                  "Output image - power spectrum"     });
      }

      public void doubleClick()
      {

      }

      public Object create(DataBlock args)
      {
        GrayImage input = (GrayImage)args.getGrayImage(0);
        double sigma = args.getDouble(1);
        double precision = 0.01;
        int Width = input.getWidth();
        int Height = input.getHeight();
        int i,j;

        System.out.println("Run Tina convolution test");

        //create imrect
        jinaImregion  reg = tina.roi_alloc(0, 0, Width, Height);
        jinaImrect    inp_im = tina.im_alloc(Height, Width, reg,
                                    tina.float_v);

        for(j=0;j<Height;j++)
            for(i=0;i<Width;i++)
                  tina.im_put_pixf(input.getxy(i,j),inp_im,j,i);

        //gaussian profile
        jinaProf1 gauss_prof=tina.prof_gauss_simple(sigma,precision);
        //apply separable convolution
        inp_im=tina.im_conv_separable(inp_im,gauss_prof,gauss_prof);

        //free profile
        tina.prof1_free(gauss_prof);

        //normalise pixel values to 0-255
        inp_im = tina.imf_scale(inp_im, 0.0, 255.0);

        //compute the Fourier power spectrum
        jinaImrect im_fft = tina.im_fft(inp_im,reg);
        jinaImrect im_spect = tina.im_power_spectrum(im_fft);

        //free imrect
        tina.im_free(im_fft);

        //normalise pixel values to 0-255
        im_spect = tina.im_sqrt(im_spect);
        im_spect = tina.imf_scale(im_spect, 0.0, 255.0);

        GrayImage sm_im = new GrayImage(Width,Height);
        GrayImage output = new GrayImage(Width,Height);

        for(j=0;j<Height;j++)
        {
          for(i=0;i<Width;i++)
          {
             sm_im.setxy(i,j, (int) tina.im_get_pixf(inp_im,j,i));
             output.setxy(i,j, (int) tina.im_get_pixf(im_spect,j,i));
          }
        }

        //free imrects
        tina.im_free(inp_im);
        tina.im_free(im_spect);

        DataBlock Output = new DataBlock();
        Output.add(sm_im);
        Output.add(output);

        return Output;
      }
}
```


NeatVision 2.0 Beta - [Image viewer]
File
Edit
View
Project
Image
Window
Help
System
User
Data
Flow Control
Utilities
Arithmetic
Histogram
Processing
Filter
Edges
Analysis
Clustering
Transform
Morphology
Colour
3D Volume
LowLevelOperators
StringOperators
MathOperators
JAIColour
Workspace - [newFile0]
IMAGE [geest.1]
DOUBLE [1.0]
conv_test
IMAGE
IMAGE
Image Viewer
Image Viewer
Image Viewer

Appendix B

Appendix B1: Main Tina library methods

void im_free(jinaImrect image)

jinaImrect im_copy(jinaImrect image)

jinaImrect im_cast(jinaImrect image, int vtype)

void im_copy_inplace(jinaImrect image2, jinaImrect image1)

jinaImrect im_subim(jinaImrect image, jinaImregion region)

jinaImrect im_alloc(int height, int width, jinaImregion region, int vtype)

jinaComplex im_sub_pixz(jinaImrect im, double y, double x)

double im_sub_pixqf(jinaImrect image, double y, double x)

double im_sub_pixf(jinaImrect image, double r, double c)

int im_sub_pix(jinaImrect image, double r, double c)

jinaComplex im_get_pixz(jinaImrect image, int i, int j)

double im_get_pixf(jinaImrect image, int i, int j)

int im_get_pix(jinaImrect image, int i, int j)

jinaVector im_row_vector(jinaImrect im, int y, int lx, int ux, int vtype)

jinaVector im_col_vector(jinaImrect im, int x, int ly, int uy, int vtype)

void im_put_pixz(jinaComplex pixval, jinaImrect image, int i, int j)

void im_pixf_dec(jinaImrect image, int i, int j)

void im_pixf_inc(jinaImrect image, int i, int j)

void im_put_pixf(double pixval, jinaImrect image, int i, int j)

jinaImregion roi_inter(jinaImregion r1, jinaImregion r2)

void roi_fill(jinaImregion roi, int lx, int ly, int ux, int uy)

int roi_inregion(jinaImregion region, int x, int y)

jinaImregion roi_outer(jinaImregion r1, jinaImregion r2)

void roi_update(jinaImregion roi, jinaImregion copy)

jinaImregion roi_copy(jinaImregion roi)

jinaImregion roi_alloc(int lx, int ly, int ux, int uy)

jinaImrect imc_add(int k, jinaImrect im)

jinaImrect imi_add(int k, jinaImrect im)

jinaImrect imf_add(double k, jinaImrect im)

jinaImrect imz_add(jinaComplex k, jinaImrect im)

jinaImrect im_add(double k, jinaImrect im)

jinaImrect im_arg(jinaImrect im)

jinaImrect im_im(jinaImrect im)

jinaImrect im_minus(jinaImrect im)

jinaImrect im_mod(jinaImrect im)

jinaImrect im_re(jinaImrect im)

```
jinaImrect imf_dfilter(jinaImrect im1)

jinaImrect imf_mod(jinaImrect im1)

jinaImrect imi_minus(jinaImrect im1)

jinaImrect imi_mod(jinaImrect im1)

jinaImrect imz_minus(jinaImrect im1)

jinaImrect imz_mod(jinaImrect im1)

jinaImrect imz_arg(jinaImrect im1)

jinaImrect imz_re(jinaImrect im1)

jinaImrect imz_im(jinaImrect im1)

jinaImrect imz_phase(jinaImrect im1)

jinaImrect im_phase(jinaImrect im)

jinaImrect im_cis(jinaImrect im1)

jinaImrect imi_sqr(jinaImrect im1)

jinaImrect imz_sqr(jinaImrect im1)

jinaImrect im_sqr(jinaImrect im)

jinaImrect imz_times(double k, jinaImrect im1)

jinaImrect im_times(double k, jinaImrect im)

jinaImrect im_conj(jinaImrect im1)

jinaImrect imf_minus(jinaImrect im1)

jinaImrect imf_power(double k, jinaImrect im1)

jinaImrect imf_rm_dc(jinaImrect im1)

jinaImrect imf_sqr(jinaImrect im1)

jinaImrect imf_times(double k, jinaImrect im1)

jinaImrect imf_aratio(double k, jinaImrect im)

jinaImrect imf_bratio(double k, jinaImrect im)

jinaImrect im_bshift(jinaImrect im, int y, int x)

jinaImrect imi_sum(jinaImrect im1, jinaImrect im2)

jinaImrect imf_sum(jinaImrect im1, jinaImrect im2)

jinaImrect imz_sum(jinaImrect im1, jinaImrect im2)

jinaImrect im_sum(jinaImrect im1, jinaImrect im2)

jinaImrect imi_diff(jinaImrect im1, jinaImrect im2)

jinaImrect imf_diff(jinaImrect im1, jinaImrect im2)

jinaImrect imz_diff(jinaImrect im1, jinaImrect im2)

jinaImrect im_diff(jinaImrect im1, jinaImrect im2)

jinaImrect imf_wsum(double a, double b, jinaImrect im1, jinaImrect im2)

jinaImrect imf_sumsq(jinaImrect im1, jinaImrect im2)

jinaImrect imi_maxsel(jinaImrect im1, jinaImrect im2)
```

jinaImrect imf_maxsel(jinaImrect im1, jinaImrect im2)

jinaImrect imz_maxsel(jinaImrect im1, jinaImrect im2)

jinaImrect im_maxsel(jinaImrect im1, jinaImrect im2)

jinaImrect imi_prod(jinaImrect im1, jinaImrect im2)

jinaImrect imf_prod(jinaImrect im1, jinaImrect im2)

jinaImrect imz_prod(jinaImrect im1, jinaImrect im2)

jinaImrect im_prod(jinaImrect im1, jinaImrect im2)

jinaImrect imf_div(jinaImrect im1, jinaImrect im2, double thresh, double val)

jinaImrect imz_div(jinaImrect im1, jinaImrect im2, double thresh, jinaComplex val)

jinaImrect im_div(jinaImrect im1, jinaImrect im2, double thresh, double val)

jinaImrect imz_cmplx(jinaImrect im1, jinaImrect im2)

jinaImrect im_conv_h(jinaImrect im1, jinaProf1 prof)

jinaImrect im_conv_v(jinaImrect im1, jinaProf1 prof)

jinaImrect im_conv_separable(jinaImrect im1, jinaProf1 prof_h, jinaProf1 prof_v)

jinaProf1 prof_gauss_simple(double sig, double precision)

jinaProf1 prof_gauss(double sig, double precision)

void im_convolve(jinaImrect new_im, jinaImrect im, jinaImrect kern)

jinaImrect imf_checquer(int width, int height, int dx, int dy)

jinaImrect imf_rect(int width, int height, int lx, int ly, int ux, int uy)

jinaImrect imf_ellipse(int width, int height, double cx, double cy, double ax, double ay)

jinaImrect imf_subpix_ellipse(int width, int height, double cx, double cy, double ax, double ay)

jinaImrect imf_subpix_ellipsoid(int width, int height, double cx, double cy, double ax, double ay)

jinaImrect imf_subpix_sellipse(int width, int height, double cx, double cy, double theta, double ax, double ay, double ex, double ey)

jinaImrect imf_delta(int width, int height, double cx, double cy)

jinaImrect imf_unif_noise(int width, int height, int dx, int dy, double a, double b)

jinaImrect imf_norm_noise(int width, int height, int dx, int dy, double a, double b)

jinaImrect im_corrupt(jinaImrect im, int dx, int dy, double a, double b)

jinaImrect imf_diffx(jinaImrect im1)

jinaImrect imf_diffy(jinaImrect im1)

jinaImrect imf_laplacian(jinaImrect im)

jinaImrect imf_sqrgrad(jinaImrect im)

jinaImrect imf_matop(jinaImrect ax, jinaImrect ay, jinaImrect mxx, jinaImrect mxy, jinaImrect myx, jinaImrect myy, jinaImrect bx, jinaImrect by)

jinaImrect imf_ddn(jinaImrect im)

jinaImrect imf_ddt(jinaImrect im)

jinaImrect imf_curv(jinaImrect im, double thresh, double val)

jinaImrect im_fireburn(jinaImrect im, double low_thresh, double high_thresh)

jinaImrect im_fft(jinaImrect im1, jinaImregion region) jinaImrect im_fft_inverse(jinaImrect im1, jinaImregion region)

jinaImrect im_power_spectrum(jinaImrect im1)

jinaProf1 gabor_profile(float phase, float sigma, float omega, float nsigma)

jinaImrect im_gabor(jinaImrect im1, double phasex, double sigmax, double omegax, double nsigmax, double phasey, double sigmay, double omegay, double nsigmay)

jinaImrect im_gabor_fft(jinaImrect im, double k, double b, double theta)

jinaImrect im_fgabor(jinaImregion roi, double k, double b, double theta)

jinaImrect imf_gauss(jinaImrect image, double sig, double precision)

jinaImrect imf_grad_h(jinaImrect image)

jinaImrect imf_grad_v(jinaImrect image)

jinaImrect imf_log(jinaImrect im1)

jinaImrect imz_log(jinaImrect im1)

jinaImrect im_log(jinaImrect im)

jinaImrect imz_exp(jinaImrect im1)

jinaImrect imf_exp(jinaImrect im)

jinaImrect im_exp(jinaImrect im)

jinaImrect imf_lsf_smooth(jinaImrect im1, double sigma)

jinaImrect imf_lsf_smooth_quad(jinaImrect im1, double sigma, int sidex, int sidey)

jinaImrect imc_median(jinaImrect im1)

jinaImrect imi_median(jinaImrect im1)

jinaImrect imf_median(jinaImrect im1)

jinaImrect imz_median(jinaImrect im)

jinaImrect im_median(jinaImrect im)

jinaImrect imf_erode(jinaImrect im1, jinaImrect el_val)

jinaProf1 prof1_alloc(int n1, int n2, int vtype)

void prof1_free(jinaProf1 prof)

jinaProf1 prof1_reverse(jinaProf1 prof)

im_vec2_free(jinaImrect im)

im_vec3_free(jinaImrect im)

void im_mat2_free(jinaImrect im)

jinaImrect im_vec2(jinaImrect im1, jinaImrect im2)

jinaImrect im_vec2_sum(jinaImrect u, jinaImrect v)

jinaImrect im_vec2_diff(jinaImrect u, jinaImrect v)

jinaImrect im_vec2_dot(jinaImrect u, jinaImrect v)

jinaImrect im_vec2_cross(jinaImrect u, jinaImrect v)

jinaImrect im_mat2_vprod(jinaImrect m, jinaImrect v)

jinaImrect im_mat2_sprod(jinaImrect u, jinaImrect m, jinaImrect v)

jinaImrect im_mat2_inverse(jinaImrect m) jinaImrect im_mat2_of_cols(jinaImrect cx, jinaImrect cy)

jinaImrect im_mat2_of_rows(jinaImrect rx, jinaImrect ry)

jinaImrect im_vec2_grad(jinaImrect im)

jinaImrect im_mat2_hessian(jinaImrect im)

jinaImrect im_mat2_det(jinaImrect m)

jinaImrect im_vec2_x(jinaImrect v)

jinaImrect im_vec2_y(jinaImrect v)

jinaImrect im_mat2_xx(jinaImrect m)

jinaImrect im_mat2_xy(jinaImrect m)

jinaImrect im_mat2_yx(jinaImrect m)

jinaImrect im_mat2_yy(jinaImrect m)

jinaImrect im_vec3(jinaImrect im1, jinaImrect im2, jinaImrect im3)

jinaImrect im_quad(jinaImrect im)

jinaImrect im_square(jinaImrect im)

jinaImrect im_quad2(jinaImrect im)

jinaImrect imi_rank(jinaImrect im, int size, double noise)

jinaImrect imf_rank(jinaImrect im, int size, double noise)

jinaImrect im_rank(jinaImrect im, int range, double noise)

jinaImrect im_rotate(jinaImrect im, double angle, jinaVec2 im_center)

jinaImrect imf_sample(double k, jinaImrect im)

jinaImrect imf_halve(jinaImrect im)

jinaImrect imf_scale(jinaImrect im1, double low, double high)

jinaImrect imf_scale_nzero(jinaImrect im1, double low, double high)

void im_gamma_scale_range_inplace(jinaImrect im, double gamma, double oldlow, double oldhigh, double newlow, double newhigh, double threslow, double threshigh)

void im_scale_range_inplace(jinaImrect im, double oldlow, double oldhigh, double newlow, double newhigh, double threslow, double threshigh)

void imf_times_inplace(double k, jinaImrect im)

void imf_add_inplace(double k, jinaImrect im)

void imf_accum_inplace(jinaImrect im1, double k, jinaImrect im2)

double imf_mean(jinaImrect im)

void imf_scale_inplace(jinaImrect im, double low, double high)

jinaImrect imz_scat(jinaImrect im1, jinaImregion roi, float scale)

jinaImrect imz_iscat(jinaImrect im1, jinaImrect im2, jinaImregion roi, float scale)

jinaImrect imz_dscat(jinaImrect graph_im, jinaImrect complex_im, jinaImregion roi, float scale)

jinaImrect im_scat(jinaImrect im, jinaImregion roi, float scale)

jinaImrect im_iscat(jinaImrect im, jinaImrect im2, jinaImregion roi, float scale)

jinaImrect im_dscat(jinaImrect im, jinaImrect im2, jinaImregion roi, float scale) jinaImrect im_shading(jinaImrect im, double slant, double tilt, double scale)

jinaImrect shade_conv(double slant, double tilt, jinaImregion roi)

jinaImrect imz_fshade(jinaImrect im1, double slant, double tilt)

jinaImrect imz_fshape(jinaImrect im1, double slant, double tilt, double limit)

jinaImrect imz_fxgrad(jinaImrect im1)

jinaImrect imz_fygrad(jinaImrect im1)

jinaImrect imf_sin(jinaImrect im1)

jinaImrect imz_sin(jinaImrect im1)

jinaImrect im_sin(jinaImrect im)

jinaImrect imf_asin(jinaImrect im1)

jinaImrect imz_asin(jinaImrect im1)

jinaImrect im_asin(jinaImrect im)

jinaImrect im_tsmooth(jinaImrect im1)

jinaImrect imf_spiral(int x_centre, int y_centre, int height, int width, double overtake, double loops)

jinaImrect imf_sqrt(jinaImrect im1)

jinaImrect imz_sqrt(jinaImrect im1)

jinaImrect im_sqrt(jinaImrect im)

int im_sup_vtype(int vtype1, int vtype2)

void terrain_data_free(jinaTerrain surf)

jinaTerrain terrain_alloc(int type, int m, int n)

jinaTerrain terrain_copy(jinaTerrain surf)

jinaTerrain terrain_make(jinaImregion region, jinaImrect mask, int samplex, int sampley)

jinaTerrain im_surface(jinaImrect im, jinaImrect mask, int samplex, int sampley, double scale)

jinaImregion roi_rectify(jinaImregion roi, jinaMat3 rect)

jinaImrect im_rectify(jinaImrect im1, jinaMat3 rect)

jinaImrect imi_window(jinaImrect im1, double thresh, double constant)

jinaImrect imf_window(jinaImrect im1, double thresh, double constant)

jinaImrect im_window(jinaImrect im, double thresh, double constant)

jinaImrect im_zeropad(jinaImrect im, int auto_pad_depth)

jinaSequence seq_alloc()

void seq_rm(jinaSequence seq)

jinaSequence seq_copy(jinaSequence seq)

jinaSequence seq_init(jinaSequence seq)

jinaImrect seq_image_get()

void seq_image_set(jinaImrect im)

jinaBrImStack seq_imstack_make(int vtype, int lz, int uz, double zscale)

void seq_imstack_free(jinaBrImStack imstack)

double seq_imstack_zscaled(jinaBrImStack imstack, int z) int seq_imstack_zunscaled(jinaBrImStack imstack, double z)

jinaImrect seq_imstack_xslice(jinaBrImStack imstack, int x)

jinaImrect seq_imstack_yslice(jinaBrImStack imstack, int y)

jinaSequence seq_get_current()

void seq_set_current(jinaSequence seq)

jinaList get_seq_start_el(jinaSequence seq)

jinaList get_seq_end_el(jinaSequence seq)

jinaList get_seq_current_el(jinaSequence seq)

void set_seq_current_el(jinaSequence seq, jinaList el)

void set_seq_start_el(jinaSequence seq, jinaList el)

void set_seq_end_el(jinaSequence seq, jinaList el)

void set_seq_current_frame(jinaSequence seq, int frame)

int get_seq_current_frame(jinaSequence seq)

jinaList get_current_seq_start_el()

jinaList get_current_seq_end_el()

jinaList get_current_seq_current_el()

void set_current_seq_current_frame(int frame)

void set_current_seq_current_el(jinaList el)

void set_current_seq_start_el(jinaList el)

void set_current_seq_end_el(jinaList el)

void seq_voxel_vtype(int vtype)

void seq_init_interp(int nblx, int nbux, int nbly, int nbuy, int nblz, int nbuz)

void seq_interp_choice(int choice)

int get_end_frame(jinaSequence seq)

jinaSeqVoi svoi_alloc()

svoi_free(jinaSeqVoi voi)

void svoi_empty(jinaSeqVoi voi)

void svoi_string_changed(jinaSeqVoi voi)

jinaTString svoi_string_get(jinaSeqVoi voi)

jinaSeqVoi svoi_copy(jinaSeqVoi oldvoi)

void svoi_string_set(jinaSeqVoi voi, jinaTString str)

jinaSpline2 svoi_spline_get(jinaSeqVoi voi)

void svoi_spline_set(jinaSeqVoi voi, jinaSpline2 spline)

void svoi_shift(jinaSeqVoi voi, jinaVec2 dp)

jinaVec2 vec2_midpoint(jinaVec2 q1, jinaVec2 q2)

jinaVec2 vec2_projperp(jinaVec2 u, jinaVec2 v)

jinaVec2 vec2_projpar(jinaVec2 u, jinaVec2 v) jinaVec2 vec2_proj_on_line(jinaVec2 q, jinaVec2 l, jinaVec2 v)

jinaVec2 vec2_inter_lines(jinaVec2 l1, jinaVec2 v1, jinaVec2 l2, jinaVec2 v2)

void vec2_join_2_points(jinaVec2 q1, jinaVec2 q2, jinaVec2 l, jinaVec2 v)

double vec2_dist_point_line(jinaVec2 q, jinaVec2 l, jinaVec2 v)

jinaVec2 vec2_inter_par_test(jinaVec2 p, jinaVec2 v1, jinaVec2 q, jinaVec2 v2, double parallel)

jinaVec3 vec3_midpoint(jinaVec3 q1, jinaVec3 q2)

jinaVec3 vec3_projperp(jinaVec3 u, jinaVec3 v)

jinaVec3 vec3_projpar(jinaVec3 u, jinaVec3 v)

jinaVec3 vec3_proj_on_line(jinaVec3 q, jinaVec3 l, jinaVec3 v)

jinaVec3 vec3_proj_on_plane(jinaVec3 q, jinaVec3 p, jinaVec3 n)

jinaVec3 vec3_closest_lines(jinaVec3 l1, jinaVec3 v1, jinaVec3 l2, jinaVec3 v2)

jinaVec3 vec3_inter_lines(jinaVec3 l1, jinaVec3 v1, jinaVec3 l2, jinaVec3 v2)

jinaVec3 vec3_inter_line_plane(jinaVec3 l, jinaVec3 v, jinaVec3 p, jinaVec3 n)

void vec3_inter_planes(jinaVec3 p1, jinaVec3 n1, jinaVec3 p2, jinaVec3 n2, jinaVec3 l, jinaVec3 v)

void vec3_join_2_points(jinaVec3 q1, jinaVec3 q2, jinaVec3 l, jinaVec3 v)

void vec3_join_3_points(jinaVec3 q1, jinaVec3 q2, jinaVec3 q3, jinaVec3 p, jinaVec3 n)

void vec3_join_point_line(jinaVec3 q, jinaVec3 l, jinaVec3 v, jinaVec3 p, jinaVec3 n)

void vec3_join_lines(jinaVec3 l1, jinaVec3 v1, jinaVec3 l2, jinaVec3 v2, jinaVec3 p, jinaVec3 n)

double vec3_dist_point_plane(jinaVec3 q, jinaVec3 p, jinaVec3 n)

double vec3_dist_point_line(jinaVec3 q, jinaVec3 l, jinaVec3 v)

double vec3_dist_lines(jinaVec3 l1, jinaVec3 v1, jinaVec3 l2, jinaVec3 v2)

int vec3_collinear(jinaVec3 p1, jinaVec3 p2, jinaVec3 q1, jinaVec3 q2, double dotth1, double dotth2)

double vec3_closest_app(jinaVec3 p1, jinaVec3 v1, jinaVec3 p2, jinaVec3 v2, jinaVec3 c1, jinaVec3 c2)

int vec3_parallel(jinaVec3 v1, jinaVec3 v2, double dotthres)

void vec2_extend_hull(jinaVec2 vmin, jinaVec2 vmax, jinaVec2 v)

void vec3_extend_hull(jinaVec3 vmin, jinaVec3 vmax, jinaVec3 v)

jinaMat2 mat2_unit()

jinaMat2 mat2_zero()

jinaVec2 mat2_rowx(jinaMat2 m)

jinaVec2 mat2_rowy(jinaMat2 m)

jinaVec2 mat2_colx(jinaMat2 m)

jinaVec2 mat2_coly(jinaMat2 m)

jinaMat2 mat2_of_rows(jinaVec2 rx, jinaVec2 ry)

jinaMat2 mat2_of_cols(jinaVec2 cx, jinaVec2 cy)

jinaMat2 mat2_sum(jinaMat2 m, jinaMat2 n)

jinaMat2 mat2_diff(jinaMat2 m, jinaMat2 n)

jinaMat2 mat2_minus(jinaMat2 m)

jinaMat2 mat2_times(double k, jinaMat2 m) jinaMat2 mat2_prod(jinaMat2 m, jinaMat2 n)

jinaMat2 mat2_inverse(jinaMat2 m)

jinaMat2 mat2_transpose(jinaMat2 m)

jinaMat2 mat2_sym(jinaMat2 m)

double mat2_trace(jinaMat2 m)

double mat2_det(jinaMat2 m)

int mat2_posdef(jinaMat2 m)

jinaVec2 mat2_vprod(jinaMat2 m, jinaVec2 v)

double mat2_sprod(jinaVec2 v, jinaMat2 m, jinaVec2 w)

jinaMat2 mat2_tensor(jinaVec2 v, jinaVec2 w)

void mat2_format(jinaMat2 m)

jinaMat3 mat3_alloc()

jinaMat3 mat3_make(jinaMat3 n)

void mat3_free(jinaMat3 m)

double mat3_get_xx(jinaMat3 m)

double mat3_get_xy(jinaMat3 m)

double mat3_get_xz(jinaMat3 m)

double mat3_get_yy(jinaMat3 m)

double mat3_get_yz(jinaMat3 m)

double mat3_get_yx(jinaMat3 m)

double mat3_get_zx(jinaMat3 m)

double mat3_get_zy(jinaMat3 m)

double mat3_get_zz(jinaMat3 m)

jinaMat3 mat3_unit()

jinaMat3 mat3_zero()

jinaMat3 mat3_diag(double mxx, double myy, double mzz)

jinaVec3 mat3_rowx(jinaMat3 m)

jinaVec3 mat3_rowy(jinaMat3 m)

jinaVec3 mat3_rowz(jinaMat3 m)

jinaVec3 mat3_colx(jinaMat3 m)

jinaVec3 mat3_coly(jinaMat3 m)

jinaVec3 mat3_colz(jinaMat3 m)

jinaMat3 mat3_of_cols(jinaVec3 cx, jinaVec3 cy, jinaVec3 cz)

jinaMat3 mat3_of_rows(jinaVec3 rx, jinaVec3 ry, jinaVec3 rz)

jinaMat3 mat3_sum(jinaMat3 m, jinaMat3 n)

jinaMat3 mat3_sum3(jinaMat3 l, jinaMat3 m, jinaMat3 n)

jinaMat3 mat3_diff(jinaMat3 m, jinaMat3 n) jinaMat3 mat3_minus(jinaMat3 m)

jinaMat3 mat3_times(double k, jinaMat3 m)

jinaMat3 mat3_prod(jinaMat3 m, jinaMat3 n)

jinaMat3 mat3_inverse(jinaMat3 m)

jinaMat3 mat3_transpose(jinaMat3 m)

double mat3_trace(jinaMat3 m)

double mat3_det(jinaMat3 m)

int mat3_posdef(jinaMat3 m)

jinaVec3 mat3_vprod(jinaMat3 m, jinaVec3 v)

jinaVec3 mat3_transpose_vprod(jinaMat3 m, jinaVec3 v)

double mat3_sprod(jinaVec3 v, jinaMat3 m, jinaVec3 w)

jinaMat3 mat3_tensor(jinaVec3 v, jinaVec3 w)

jinaMat3 mat3_sum_tensor(jinaMat3 m, jinaVec3 v, jinaVec3 w)

void mat3_format(jinaMat3 m)

jinaMat4 mat4_alloc()

jinaMat4 mat4_make(jinaMat4 n)

void mat4_free(jinaMat4 m)

double mat4_get_xx(jinaMat4 m)

double mat4_get_xy(jinaMat4 m)

double mat4_get_xz(jinaMat4 m)

double mat4_get_xw(jinaMat4 m)

double mat4_get_yx(jinaMat4 m)

double mat4_get_yy(jinaMat4 m)

double mat4_get_yz(jinaMat4 m)

double mat4_get_yw(jinaMat4 m)

double mat4_get_zx(jinaMat4 m)

double mat4_get_zy(jinaMat4 m)

double mat4_get_zz(jinaMat4 m)

double mat4_get_zw(jinaMat4 m)

double mat4_get_wx(jinaMat4 m)

double mat4_get_wy(jinaMat4 m)

double mat4_get_wz(jinaMat4 m)

double mat4_get_ww(jinaMat4 m)

jinaMat4 mat4_unit()

jinaMat4 mat4_zero()

jinaVec4 mat4_rowx(jinaMat4 m)

jinaVec4 mat4_rowy(jinaMat4 m)

jinaVec4 mat4_rowz(jinaMat4 m)

jinaVec4 mat4_roww(jinaMat4 m

jinaVec4 mat4_colx(jinaMat4 m)

jinaVec4 mat4_coly(jinaMat4 m)

jinaVec4 mat4_colz(jinaMat4 m)

jinaVec4 mat4_colw(jinaMat4 m)

jinaMat4 mat4_of_cols(jinaVec4 cx, jinaVec4 cy, jinaVec4 cz, jinaVec4 cw)

jinaMat4 mat4_of_rows(jinaVec4 rx, jinaVec4 ry, jinaVec4 rz, jinaVec4 rw)

jinaMat4 mat4_sum(jinaMat4 m, jinaMat4 n)

jinaMat4 mat4_diff(jinaMat4 m, jinaMat4 n)

jinaMat4 mat4_minus(jinaMat4 m)

jinaMat4 mat4_prod(jinaMat4 m, jinaMat4 n)

jinaMat4 mat4_inverse(jinaMat4 m)

jinaMat4 mat4_transpose(jinaMat4 m)

double mat4_trace(jinaMat4 m)

jinaVec4 mat4_vprod(jinaMat4 m, jinaVec4 v)

jinaVec4 mat4_transpose_vprod(jinaMat4 m, jinaVec4 v)

double mat4_sprod(jinaVec4 v, jinaMat4 m, jinaVec4 w)

jinaMat4 mat4_tensor(jinaVec4 v, jinaVec4 w)

void mat4_format(jinaMat4 m)

jinaVec2 vec2_of_proj2(jinaVec3 v)

jinaVec3 proj2_of_vec2(jinaVec2 v)

jinaVec3 proj2_rectify(jinaMat3 m, jinaVec3 v)

jinaVec2 vec2_rectify(jinaMat3 m, jinaVec2 v

jinaVec3 proj2_join(jinaVec3 p, jinaVec3 q)

jinaVec3 proj2_intersect(jinaVec3 p, jinaVec3 q)

jinaMat3 proj2_to_frame(jinaVec3 p00, jinaVec3 p10, jinaVec3 p01, jinaVec3 p11)

jinaMat3 proj2_between(jinaVec2 p00, jinaVec2 p10, jinaVec2 p01, jinaVec2 p11, jinaVec2 q00, jinaVec2 q10, jinaVec2 q01, jinaVec2 q11)

jinaMat3 proj2_between_proj2(jinaVec3 p00, jinaVec3 p10, jinaVec3 p01, jinaVec3 p11, jinaVec3 q00, jinaVec3 q10, jinaVec3 q01, jinaVec3 q11)

jinaMat3 proj_between_ls(int n, jinaVec3 p, jinaVec3 q)

jinaVec4 proj3_rectify(jinaMat4 m, jinaVec4 v)

jinaVec3 vec3_rectify(jinaMat4 m, jinaVec3 v

jinaVec2 vec2_alloc()

jinaVec2 vec2_copy(jinaVec2 vec2)

jinaVec2 vec2_make(jinaVec2 u)

jinaVec2 vec2_zero()

jinaVec2 vec2_ex()

jinaVec2 vec2_ey()

double vec2_get_x(jinaVec2 v)

double vec2_get_y(jinaVec2 v

jinaVec2 vec2_sum(jinaVec2 v, jinaVec2 w)

jinaVec2 vec2_sum3(jinaVec2 u, jinaVec2 v, jinaVec2 w)

jinaVec2 vec2_sum4(jinaVec2 u, jinaVec2 v, jinaVec2 w, jinaVec2 x)

jinaVec2 vec2_minus(jinaVec2 v)

jinaVec2 vec2_diff(jinaVec2 v, jinaVec2 w)

jinaVec2 vec2_times(double k, jinaVec2 v)

jinaVec2 vec2_interp(double k, jinaVec2 v1, jinaVec2 v2)

double vec2_dot(jinaVec2 v, jinaVec2 w)

double vec2_cross(jinaVec2 v, jinaVec2 w)

double vec2_mod(jinaVec2 v)

double vec2_sqrmod(jinaVec2 v)

double vec2_modunit(jinaVec2 v, jinaVec2 e)

jinaVec2 vec2_unit(jinaVec2 v)

jinaVec2 vec2_of_polar(double r, double theta)

jinaVec2 vec2_rand_circle(jinaVec2 centre, double radius)

double vec2_dist(jinaVec2 v, jinaVec2 w)

double vec2_sqrdist(jinaVec2 v, jinaVec2 w)

double vec2_angle(jinaVec2 v, jinaVec2 w)

jinaVec2 vec2_perp(jinaVec2 v)

double vec2_perp_dist(jinaVec2 p, jinaVec2 v, jinaVec2 d)

void vec2_basis(jinaVec2 up, jinaVec2 ex, jinaVec2 ey)

int vec2_parallel(jinaVec2 v1, jinaVec2 v2, double dotthres)

void vec2_format(jinaVec2 v)

jinaVec2 vec2_less(jinaVec2 v1, jinaVec2 v2)

jinaVec2 vec2_greater(jinaVec2 v1, jinaVec2 v2)

void vec2_ranges(jinaVec2 v1, jinaVec2 v2, jinaVec2 vec)

jinaVec3 vec3_alloc()

jinaVec3 vec3_make(jinaVec3 u)

jinaVec3 vec3_copy(jinaVec3 u)

jinaVec3 vec3_zero()

jinaVec3 vec3_ex()

jinaVec3 vec3_ey()

jinaVec3 vec3_ez()

double vec3_get_x(jinaVec3 v)

double vec3_get_y(jinaVec3 v)

double vec3_get_z(jinaVec3 v

jinaVec3 vec3_sum(jinaVec3 v, jinaVec3 w)

jinaVec3 vec3_sum3(jinaVec3 u, jinaVec3 v, jinaVec3 w)

jinaVec3 vec3_sum4(jinaVec3 u, jinaVec3 v, jinaVec3 w, jinaVec3 x)

jinaVec3 vec3_minus(jinaVec3 v)

jinaVec3 vec3_diff(jinaVec3 v, jinaVec3 w

jinaVec3 vec3_times(double k, jinaVec3 v)

jinaVec3 vec3_interp(double k, jinaVec3 v1, jinaVec3 v2)

double vec3_dot(jinaVec3 v, jinaVec3 w)

jinaVec3 vec3_cross(jinaVec3 v, jinaVec3 w)

jinaVec3 vec3_unitcross(jinaVec3 v, jinaVec3 w)

double vec3_mod(jinaVec3 v)

double vec3_sqrmod(jinaVec3 v)

double vec3_modunit(jinaVec3 v, jinaVec3 e)

jinaVec3 vec3_unit(jinaVec3 v)

double vec3_dist(jinaVec3 v, jinaVec3 w)

double vec3_sqrdist(jinaVec3 v, jinaVec3 w)

double vec3_angle(jinaVec3 v, jinaVec3 w)

jinaVec3 vec3_perp(jinaVec3 v)

void vec3_basis(jinaVec3 aim, jinaVec3 down, jinaVec3 ex, jinaVec3 ey, jinaVec3 ez)

void vec3_format(jinaVec3 v)

jinaVec3 vec3_rand_sphere(jinaVec3 c, double r)

jinaVec4 vec4_alloc()

jinaVec4 vec4_make(jinaVec4 u)

jinaVec4 vec4_zero()

jinaVec4 vec4_ex()

jinaVec4 vec4_ey()

jinaVec4 vec4_ez()

jinaVec4 vec4_ew()

double vec4_get_x(jinaVec4 v)

double vec4_get_y(jinaVec4 v)

double vec4_get_z(jinaVec4 v)

double vec4_get_w(jinaVec4 v)

jinaVec4 vec4_sum(jinaVec4 v, jinaVec4 w)

jinaVec4 vec4_sum3(jinaVec4 u, jinaVec4 v, jinaVec4 w)

jinaVec4 vec4_sum4(jinaVec4 u, jinaVec4 v, jinaVec4 w, jinaVec4 x)

jinaVec4 vec4_minus(jinaVec4 v)

jinaVec4 vec4_diff(jinaVec4 v, jinaVec4 w)

jinaVec4 vec4_times(double k, jinaVec4 v)

jinaVec4 vec4_interp(double k, jinaVec4 v1, jinaVec4 v2)

double vec4_dot(jinaVec4 v, jinaVec4 w)

jinaMat4 vec4_cross(jinaVec4 v, jinaVec4 w)

double vec4_mod(jinaVec4 v)

double vec4_sqrmod(jinaVec4 v)

double vec4_modunit(jinaVec4 v, jinaVec4 e)

jinaVec4 vec4_unit(jinaVec4 v)

double vec4_dist(jinaVec4 v, jinaVec4 w)

double vec4_sqrdist(jinaVec4 v, jinaVec4 w)

double vec4_angle(jinaVec4 v, jinaVec4 w)

void vec4_format(jinaVec4 v)

jinaMatrix matrix_sum(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix matrix_add(jinaMatrix m1, jinaMatrix m2)

jinaMatrix imatrix_add(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix fmatrix_add(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix dmatrix_add(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix imatrix_add_inplace(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix fmatrix_add_inplace(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix dmatrix_add_inplace(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix mat_alloc(int m, int n)

jinaMatrix matrix_alloc(int m, int n, int shape, int vtype)

jinaMatrix cmatrix_alloc(int m, int n, int shape, int vtype)

jinaMatrix smatrix_alloc(int m, int n, int shape, int vtype)

jinaMatrix imatrix_alloc(int m, int n, int shape, int vtype)

jinaMatrix fmatrix_alloc(int m, int n, int shape, int vtype)

jinaMatrix dmatrix_alloc(int m, int n, int shape, int vtype)

jinaMatrix zmatrix_alloc(int m, int n, int shape, int vtype)

jinaMatrix pmatrix_alloc(int m, int n, int shape, int vtype)

jinaMatrix matrix_expand(jinaMatrix mat, int m, int n)

jinaMatrix matrix_cast(jinaMatrix mat, int vtype)

jinaMatrix imatrix_cast(jinaMatrix mat)

jinaMatrix fmatrix_cast(jinaMatrix mat)

jinaMatrix dmatrix_cast(jinaMatrix mat)

jinaMatrix matrix_copy(jinaMatrix mat)

jinaMatrix imatrix_copy(jinaMatrix mat)

jinaMatrix fmatrix_copy(jinaMatrix mat)

jinaMatrix dmatrix_copy(jinaMatrix mat)

jinaMatrix matrix_copy_inplace(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix matrix_fill(jinaMatrix mat)

jinaMatrix matrix_cast_fill(jinaMatrix mat, int vtype)

void mat_format(jinaMatrix mat)

void matrix_format(jinaMatrix mat)

void cmatrix_format(jinaMatrix mat)

void cmatrix_format_full(jinaMatrix mat)

void cmatrix_format_lower(jinaMatrix mat)

void cmatrix_format_gen(jinaMatrix mat)

void smatrix_format(jinaMatrix mat)

void smatrix_format_full(jinaMatrix mat)

void smatrix_format_lower(jinaMatrix mat)

void smatrix_format_gen(jinaMatrix mat)

void imatrix_format(jinaMatrix mat)

void imatrix_format_full(jinaMatrix mat)

void imatrix_format_lower(jinaMatrix mat)

void imatrix_format_gen(jinaMatrix mat)

void fmatrix_format(jinaMatrix mat)

void fmatrix_format_full(jinaMatrix mat)

void fmatrix_format_lower(jinaMatrix mat)

void fmatrix_format_gen(jinaMatrix mat)

void dmatrix_format(jinaMatrix mat)

void dmatrix_format_full(jinaMatrix mat)

void dmatrix_format_lower(jinaMatrix mat)

void dmatrix_format_gen(jinaMatrix mat)

void zmatrix_format(jinaMatrix mat)

void zmatrix_format_full(jinaMatrix mat)

void zmatrix_format_lower(jinaMatrix mat)

void zmatrix_format_gen(jinaMatrix mat)

void pmatrix_format(jinaMatrix mat)

void pmatrix_format_full(jinaMatrix mat)

void pmatrix_format_lower(jinaMatrix mat)

void pmatrix_format_gen(jinaMatrix mat)

void format_shape(int shape)

void matrix_free(jinaMatrix mat)

void cmatrix_free(jinaMatrix mat)

void smatrix_free(jinaMatrix mat)

void imatrix_free(jinaMatrix mat)

void fmatrix_free(jinaMatrix mat)

void dmatrix_free(jinaMatrix mat)

void pmatrix_free(jinaMatrix mat)

void zmatrix_free(jinaMatrix mat)

void matrix_set_default_val(int ival)

void matrix_set_default_fval(double fval)

void matrix_set_default_zval(jinaComplex zval)

double mat_getf(jinaMatrix mat, int i, int j)

int matrix_get(jinaMatrix mat, int i, int j)

int matrix_get_full(jinaMatrix mat, int i, int j)

double matrix_getf(jinaMatrix mat, int i, int j)

double matrix_getf_full(jinaMatrix mat, int i, int j)

jinaComplex matrix_getz(jinaMatrix mat, int i, int j)

jinaComplex matrix_getz_full(jinaMatrix mat, int i, int j)

jinaMatrix matrix_invert(jinaMatrix mat)

jinaMatrix dmatrix_invert(jinaMatrix mat)

jinaMatrix matrix_invsvd(jinaMatrix a, double condition)

jinaMatrix matrix_itimes(int k, jinaMatrix mat)

jinaMatrix matrix_mat2(jinaMat2 m)

jinaMat2 mat2_matrix(jinaMatrix mat)

jinaMatrix matrix_mat3(jinaMat3 m)

jinaMat3 mat3_matrix(jinaMatrix mat)

jinaMatrix matrix_minus(jinaMatrix mat)

jinaMatrix fmatrix_minus(jinaMatrix mat)

jinaMatrix fmatrix_minus_inplace(jinaMatrix mat)

jinaMatrix matrix_mult(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix imatrix_mult(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix fmatrix_mult(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix fmatrix_mult(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix matrix_prod(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix imatrix_prod(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix fmatrix_prod(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix dmatrix_prod(jinaMatrix mat1, jinaMatrix mat2)

void mat_putf(float val, jinaMatrix mat, int i, int j)

void matrix_put(int val, jinaMatrix mat, int i, int j) void matrix_put_full(int val, jinaMatrix mat, int i, int j)

void matrix_putf(double val, jinaMatrix mat, int i, int j)

void matrix_putf_full(double val, jinaMatrix mat, int i, int j)

void matrix_putz(jinaComplex val, jinaMatrix mat, int i, int j)

void matrix_putz_full(jinaComplex val, jinaMatrix mat, int i, int j)

jinaMatrix matrix_diff(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix matrix_sub(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix imatrix_sub(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix fmatrix_sub(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix dmatrix_sub(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix imatrix_sub_inplace(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix fmatrix_sub_inplace(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix dmatrix_sub_inplace(jinaMatrix mat1, jinaMatrix mat2)

jinaMatrix matrix_tensor(jinaVector v1, jinaVector v2)

jinaMatrix matrix_times(double k, jinaMatrix mat)

jinaMatrix fmatrix_times(double k, jinaMatrix mat)

jinaMatrix dmatrix_times(double k, jinaMatrix mat)

jinaMatrix matrix_transp(jinaMatrix mat)

jinaMatrix imatrix_transp(jinaMatrix mat)

jinaMatrix fmatrix_transp(jinaMatrix mat)

jinaMatrix dmatrix_transp(jinaMatrix mat)

jinaMatrix matrix_transform2(jinaTransform2 transf)

jinaTransform2 trans2_matrix(jinaMatrix mat)

jinaMatrix matrix_transform3(jinaTransform3 transf)

jinaTransform3 trans3_matrix(jinaMatrix mat)

jinaMatrix matrix_unit(int m, int n, int shape, int vtype)

int matrix_sup_shape(int shape1, int shape2)

int matrix_sup_vtype(int vtype1, int vtype2)

int matrix_swap_rows(jinaMatrix mat, int r1, int r2)

int matrix_swap_cols(jinaMatrix mat, int c1, int c2)

jinaVector matrix_col_vector(jinaMatrix mat, int c)

jinaVector matrix_vprod(jinaMatrix mat, jinaVector vec)

jinaVector imatrix_vprod(jinaMatrix mat, jinaVector vec)

jinaVector fmatrix_vprod(jinaMatrix mat, jinaVector vec)

jinaVector dmatrix_vprod(jinaMatrix mat, jinaVector vec)

int mat_eigen(jinaMat a, jinaMat evec, jinaVec eval)

void mat_invert(jinaMat a)

void mat_solve(jinaMat a, jinaVec y)

double mat_det(jinaMat a)

jinaMat mat_make(int m, int n)

void mat_free(jinaMat a)

void mat_copy(jinaMat a, jinaMat b)

void mat_zero(jinaMat a)

void mat_unit(jinaMat a)

void mat_rand_unif(jinaMat a, double p, double q)

void mat_rand_normal(jinaMat a, double m, double s)

void mat_transp(jinaMat at, jinaMat a)

void mat_stransp(jinaMat a)

void mat_prod(jinaMat ab, jinaMat a, jinaMat b)

void mat_sqr(jinaMat aa, jinaMat a)

void mat_vprod(jinaVec av, jinaMat a, jinaVec v)

void mat_dprod(jinaVec d, jinaMat a)

double mat_sprod(jinaVec v, jinaMat a, jinaVec w)

void mat_sum(jinaMat a, jinaMat b)

void mat_diff(jinaMat a, jinaMat b)

void mat_times(double k, jinaMat a)

void mat_minus(jinaMat a)

void mat_accum(jinaMat a, double k, jinaMat b)

void mat_sum_tensor(jinaMat a, jinaVec v, jinaVec w)

void mat_accum_tensor(jinaMat a, double k, jinaVec v, jinaVec w)

void mat_row(jinaVec row, jinaMat a, int i)

void mat_row_get(jinaMat a, int i, jinaVec row)

void mat_row_set(jinaMat a, int i, jinaVec row)

void mat_col_get(jinaMat a, int j, jinaVec col)

void mat_col_set(jinaMat a, int j, jinaVec col)

void mat_block_get(jinaMat a, int li, int lj, jinaMat b)

void mat_block_set(jinaMat a, int li, int lj, jinaMat b)

void mat_index_get(jinaMat a, jinaIvec indi, jinaIvec indj, jinaMat b)

void mat_index_set(jinaMat a, jinaIvec indi, jinaIvec indj, jinaMat b)

void mat_mat3(jinaMat a, jinaMat3 a3)

jinaMat3 mat3_mat(jinaMat a)

void mat_svd(jinaMat a, jinaMat p, jinaVec d, jinaMat q)

void vec_copy(jinaVec v, jinaVec w)

jinaVec vec_make_copy(jinaVec w)

void vec_zero(jinaVec v)

void vec_rand_unif(jinaVec v, double p, double q)

void vec_rand_normal(jinaVec v, double m, double s)

double vec_dot(jinaVec v, jinaVec w)

void vec_sum(jinaVec v, jinaVec w)

void vec_diff(jinaVec v, jinaVec w)

void vec_prod(jinaVec v, jinaVec w)

void vec_divide(jinaVec v, jinaVec w)

void vec_times(double k, jinaVec v)

void vec_minus(jinaVec v)

void vec_accum(jinaVec v, double k, jinaVec w)

double vec_sqrmod(jinaVec v)

double vec_mod(jinaVec v)

void vec_unit(jinaVec v)

double vec_mod1(jinaVec v)

double vec_sqrdist(jinaVec v, jinaVec w)

double vec_dist(jinaVec v, jinaVec w)

void vec_reverse(jinaVec v, jinaVec w)

void vec_block_get(jinaVec u, int li, jinaVec v)

vec_block_set(jinaVec u, int li, jinaVec v)

void vec_index_get(jinaVec u, jinaIvec index, jinaVec v)

void vec_index_set(jinaVec u, jinaIvec index, jinaVec v)

jinaSpline spline_make(int type, int n)

jinaSpline spline_copy(jinaSpline spline)

void spline_free(jinaSpline spline)

double spline_eval(jinaSpline spline, double t)

double spline_periodic_param(jinaSpline spline, double p)

double spline_param(jinaSpline spline, double p)

void spline_replace_point(jinaSpline spline, int ip, double p)

void spline_add_point(jinaSpline spline, int ibelow, double p)

void spline_delete_point(jinaSpline spline, int ip)

jinaSpline2 spline2_make(int type, int n)

jinaSpline2 spline2_alloc(int type, int n)

jinaSpline2 spline2_copy(jinaSpline2 spline)

void spline2_free(jinaSpline2 spline)

jinaVec2 spline2_eval(jinaSpline2 spline, double t)

double spline2_natural_param(jinaSpline2 spline, jinaVec2 p)

double spline2_param(jinaSpline2 spline, jinaVec2 p)

jinaVec2 spline2_closest(jinaSpline2 spline, jinaVec2 p) double spline2_dist(jinaSpline2 spline, jinaVec2 p)

void spline2_interpolate(jinaSpline2 spline, jinaVec2 p)

jinaSpline2 spline2_interpolate_list(int type, jinaList points)

jinaSpline2 spline2_interpolate_ddlist(int type, jinaList points)

void spline2_replace_point(jinaSpline2 spline, int ip, jinaVec2 p)

void spline2_add_point(jinaSpline2 spline, int ibelow, jinaVec2 p)

void spline2_delete_point(jinaSpline2 spline, int i)

void spline2_rts(jinaSpline2 spline, jinaMat2 r, jinaVec2 t, double s)

jinaVec3 vec3_quat(jinaVec4 q)

jinaVec4 quat_vec3(jinaVec3 v)

jinaVec4 quat_rot3(jinaMat3 r)

jinaMat3 rot3_quat(jinaVec4 q)

jinaVec4 quat_prod(jinaVec4 p, jinaVec4 q)

jinaVec4 quat_conj(jinaVec4 q)

jinaVec4 quat_inverse(jinaVec4 q)

jinaVec3 vec3_rot_quat(jinaVec4 q, jinaVec3 v)

jinaMat2 rot2(double theta)

double rot2_angle(jinaMat2 m)

jinaMat2 rot2_with_scale(double theta, double scale_factor)

jinaMat3 rot3(double theta, jinaVec3 axis)

jinaMat3 rot3_1(jinaVec3 axis)

jinaMat3 rot3_euler(double theta, double phi, double psi)

double asinh(double x)

double tina_acos(double c)

double sqr(double x)

double dist2(double x1, double y1, double x2, double y2)

int imin(int x, int y)

int imax(int x, int y)

double tina_fmin(double x, double y)

double tina_fmax(double x, double y)

int imin3(int x, int y, int z)

int imax3(int x, int y, int z)

double fmin3(double x, double y, double z)

double fmax3(double x, double y, double z)

double im_fraction(jinaImrect cut, jinaImrect image1, jinaImrect image2, int nbin, double sig_noise)

double im_corscale(jinaImrect im0, jinaImrect corr, double noise)

double im_corscale2(jinaImrect im0, jinaImrect corr, double noise) jinaImrect im_integrate(jinaImrect imdxf, jinaImrect imdyf)

double imf_diffx_noise(jinaImrect im1, jinaImregion roi)

double imf_diffy_noise(jinaImrect im1, jinaImregion roi)

jinaImrect xy_norm(jinaImrect im, double constant, double sigma, double thresh)

double imf_sigma(jinaImrect im2, jinaImrect immask, jinaImregion roi)