QtGabor2D Reference Manual

Generated by Doxygen 1.2.6

Sun May 27 23:06:30 2001

Contents

Chapter 1

QtGabor2D Hierarchical Index

1.1 QtGabor2D Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

QtGabor2D Compound Index

2.1 QtGabor2D Compound List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

QtGabor2D File Index

3.1 QtGabor2D File List

Here is a list of all files with brief descriptions:

Chapter 4

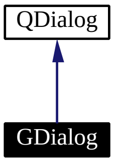
QtGabor2D Class Documentation

4.1 GDialog Class Reference

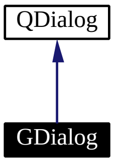
The class for the dialogbox for the setting the Gabor parameters.

`#include <gdialog.h>`

Inheritance diagram for GDialog:



Collaboration diagram for GDialog:



Public Methods

- **GDialog** (**QWidget** *parent=0, const char *name=0, bool modal=FALSE, WFlags f=0)

    Constructor for the GDialog class.

- **~GDialog** ()

Signals

- void **sendParam** (QString, QString, QString, QString, QString, QString, QString, QString, QString, QString)

Private Attributes

- QGridLayout* **topLayout**
- QLineEdit* **S**
- QLineEdit * **K**
- QLineEdit * **P**
- QLineEdit * **a**
- QLineEdit * **b**
- QLineEdit * **theta**
- QLineEdit * **x0**
- QLineEdit * **y0**
- QLineEdit * **u0**
- QLineEdit * **v0**
- QPushButton* **apply**
- QPushButton* **close**

Private Slots

- void **send** ()

4.1.1 Detailed Description

The class for the dialogbox for the setting the Gabor parameters.

This class is for the dialog box for Gabor filter.

Definition at line 11 of file gdialog.h.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 GDialog::GDialog (QWidget * *parent* = 0, const char * *name* = 0, bool *modal* = FALSE, WFlags *f* = 0)

Constructor for the GDialog class.

The dialog box is implemented with a QGridLayout. All the parameters for the Gabor filters are set using this dialog, and initially they have default values.

Definition at line 7 of file gdialog.cc.

4.1.2.2 GDialog::~GDialog ()

4.1.3 Member Function Documentation

4.1.3.1 void GDialog::send () `[private, slot]`

Referenced by GDialog().

4.1.3.2 void GDialog::sendParam (QString, QString, QString, QString, QString, QString, QString, QString, QString, QString) `[signal]`

4.1.4 Member Data Documentation

4.1.4.1 QLineEdit * GDialog::K `[private]`

Definition at line 17 of file gdialog.h.

4.1.4.2 QLineEdit * GDialog::P `[private]`

Definition at line 17 of file gdialog.h.

4.1.4.3 QLineEdit* GDialog::S `[private]`

Definition at line 17 of file gdialog.h.

4.1.4.4 QLineEdit * GDialog::a `[private]`

Definition at line 17 of file gdialog.h.

4.1.4.5 QPushButton* GDialog::apply `[private]`

Definition at line 18 of file gdialog.h.

4.1.4.6 QLineEdit * GDialog::b `[private]`

Definition at line 17 of file gdialog.h.

4.1.4.7 QPushButton* GDialog::close `[private]`

Definition at line 19 of file gdialog.h.

4.1.4.8 QLineEdit * GDialog::theta `[private]`

Definition at line 17 of file gdialog.h.

4.1.4.9 QGridLayout* GDialog::topLayout `[private]`

Definition at line 16 of file gdialog.h.

4.1.4.10 QLineEdit * GDialog::u0 `[private]`

Definition at line 17 of file gdialog.h.

4.1.4.11 QLineEdit * GDialog::v0 `[private]`

Definition at line 17 of file gdialog.h.

4.1.4.12 QLineEdit * GDialog::x0 `[private]`

Definition at line 17 of file gdialog.h.

4.1.4.13 QLineEdit * GDialog::y0 `[private]`

Definition at line 17 of file gdialog.h.

The documentation for this class was generated from the following files:

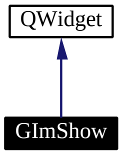
- **gdialog.h**
- **gdialog.cc**

4.2 GImShow Class Reference

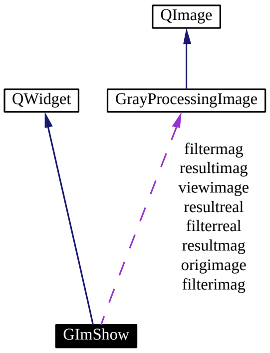
The imshow class for the QtGabor2D application.

`#include <gimshow.h>`

Inheritance diagram for GImShow:



Collaboration diagram for GImShow:



Public Methods

- int **width** ()
- int **height** ()
- bool **loadImage** (const char ∗filename)
- bool **saveImage** (const char ∗filename)
- **GImShow** (**QWidget** ∗parent=0, const char ∗name=0, WFlags f=0)

  Constructor for the GImShow class.

- **~GImShow** ()

Public Slots

- void **slotViewOrigImage** ()
- void **slotViewFilterReal** ()
- void **slotViewFilterImag** ()
- void **slotViewFilterMag** ()
- void **slotViewResultReal** ()
- void **slotViewResultImag** ()
- void **slotViewResultMag** ()

- void **slotSetFilter** (QString Ss, QString Ks, QString Ps, QString as, QString bs, QString x0s, QString y0s, QString u0s, QString v0s, QString rs)

  The most important function which takes care of all the computation.

Protected Methods

- void **paintEvent** (QPaintEvent *e)

  paintEvent function which controls the size of the entire application.

Private Attributes

- **GrayProcessingImage* viewimage**
- **GrayProcessingImage * origimage**
- **GrayProcessingImage * filterreal**
- **GrayProcessingImage * filterimag**
- **GrayProcessingImage * filtermag**
- **GrayProcessingImage * resultreal**
- **GrayProcessingImage * resultimag**
- **GrayProcessingImage * resultmag**
- QProgressDialog* **progress**
- QPainter* **paint**
- QRgb **temprgb**
- QRect **rec**
- complex<double>** **filter**
- complex<double> ** **result**
- double **maxfilterreal**
- double **minfilterreal**
- double **maxfilterimag**
- double **minfilterimag**
- double **maxfiltermag**
- double **minfiltermag**
- double **maxresultreal**
- double **minresultreal**
- double **maxresultimag**
- double **minresultimag**
- double **maxresultmag**
- double **minresultmag**
- double** **procimg**
- double **K**
- double **P**
- double **a**
- double **b**
- double **theta**
- double **x0**
- double **y0**
- double **u0**
- double **v0**

- double **xoff**
- double **yoff**
- double **re**
- double **im**
- double **rx**
- double **ry**
- int **totprogsteps**
- int **temp**
- int **S**
- int **size**

4.2.1 Detailed Description

The imshow class for the QtGabor2D application.

This class is just a wrapper for the grayprocessingimage to be displayed.

Definition at line 16 of file gimshow.h.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 GImShow::GImShow (QWidget ∗ *parent* = 0, const char ∗ *name* = 0, WFlags *f* = 0)

Constructor for the GImShow class.

Keeps the QImages and displays either the original image, the real, imag og magnitude part of the filter or of the filtered image.

Definition at line 12 of file gimshow.cc.

4.2.2.2 GImShow::~GImShow ()

4.2.3 Member Function Documentation

4.2.3.1 int GImShow::height ()

Referenced by slotSetFilter().

4.2.3.2 bool GImShow::loadImage (const char ∗ *filename*)

4.2.3.3 void GImShow::paintEvent (QPaintEvent ∗ *e*) [protected]

paintEvent function which controls the size of the entire application.

This is clumsy implemented since there should be a separate resize event handler. The paintEvent function shoud simply handle the event generated repaints.

Definition at line 80 of file gimshow.cc.

4.2.3.4 bool GImShow::saveImage (const char * *filename*)

4.2.3.5 void GImShow::slotSetFilter (QString *Ss*, QString *Ks*, QString *Ps*, QString *as*, QString *bs*, QString *x0s*, QString *y0s*, QString *u0s*, QString *v0s*, QString *thetas*) `[slot]`

The most important function which takes care of all the computation.

The size of the filter is set dependant on the size of the Gabor kernel determined by the parameters set by the user.

Definition at line 149 of file gimshow.cc.

4.2.3.6 void GImShow::slotViewFilterImag () `[slot]`

4.2.3.7 void GImShow::slotViewFilterMag () `[slot]`

4.2.3.8 void GImShow::slotViewFilterReal () `[slot]`

4.2.3.9 void GImShow::slotViewOrigImage () `[slot]`

4.2.3.10 void GImShow::slotViewResultImag () `[slot]`

4.2.3.11 void GImShow::slotViewResultMag () `[slot]`

4.2.3.12 void GImShow::slotViewResultReal () `[slot]`

4.2.3.13 int GImShow::width ()

Referenced by slotSetFilter().

4.2.4 Member Data Documentation

4.2.4.1 double GImShow::K `[private]`

Definition at line 29 of file gimshow.h.

4.2.4.2 double GImShow::P `[private]`

Definition at line 29 of file gimshow.h.

4.2.4.3 int GImShow::S `[private]`

Definition at line 31 of file gimshow.h.

4.2.4.4 double GImShow::a `[private]`

Definition at line 29 of file gimshow.h.

4.2.4.5 double GImShow::b [private]

Definition at line 29 of file gimshow.h.

4.2.4.6 complex<double>** GImShow::filter [private]

Definition at line 26 of file gimshow.h.

4.2.4.7 GrayProcessingImage * GImShow::filterimag [private]

Definition at line 21 of file gimshow.h.

4.2.4.8 GrayProcessingImage * GImShow::filtermag [private]

Definition at line 21 of file gimshow.h.

4.2.4.9 GrayProcessingImage * GImShow::filterreal [private]

Definition at line 21 of file gimshow.h.

4.2.4.10 double GImShow::im [private]

Definition at line 29 of file gimshow.h.

4.2.4.11 double GImShow::maxfilterimag [private]

Definition at line 27 of file gimshow.h.

4.2.4.12 double GImShow::maxfiltermag [private]

Definition at line 27 of file gimshow.h.

4.2.4.13 double GImShow::maxfilterreal [private]

Definition at line 27 of file gimshow.h.

4.2.4.14 double GImShow::maxresultimag [private]

Definition at line 27 of file gimshow.h.

4.2.4.15 double GImShow::maxresultmag [private]

Definition at line 27 of file gimshow.h.

4.2.4.16 double GImShow::maxresultreal [private]

Definition at line 27 of file gimshow.h.

4.2.4.17 double GImShow::minfilterimag [private]

Definition at line 27 of file gimshow.h.

4.2.4.18 double GImShow::minfiltermag [private]

Definition at line 27 of file gimshow.h.

4.2.4.19 double GImShow::minfilterreal [private]

Definition at line 27 of file gimshow.h.

4.2.4.20 double GImShow::minresultimag [private]

Definition at line 27 of file gimshow.h.

4.2.4.21 double GImShow::minresultmag [private]

Definition at line 27 of file gimshow.h.

4.2.4.22 double GImShow::minresultreal [private]

Definition at line 27 of file gimshow.h.

4.2.4.23 GrayProcessingImage * GImShow::origimage [private]

Definition at line 21 of file gimshow.h.

4.2.4.24 QPainter* GImShow::paint [private]

Definition at line 23 of file gimshow.h.

4.2.4.25 double** GImShow::procimg [private]

Definition at line 28 of file gimshow.h.

4.2.4.26 QProgressDialog* GImShow::progress [private]

Definition at line 22 of file gimshow.h.

4.2.4.27 double GImShow::re [private]

Definition at line 29 of file gimshow.h.

4.2.4.28 QRect GImShow::rec [private]

Definition at line 25 of file gimshow.h.

4.2.4.29 complex<double> ** GImShow::result [private]

Definition at line 26 of file gimshow.h.

4.2.4.30 GrayProcessingImage * GImShow::resultimag [private]

Definition at line 21 of file gimshow.h.

4.2.4.31 GrayProcessingImage * GImShow::resultmag [private]

Definition at line 21 of file gimshow.h.

4.2.4.32 GrayProcessingImage * GImShow::resultreal [private]

Definition at line 21 of file gimshow.h.

4.2.4.33 double GImShow::rx [private]

Definition at line 29 of file gimshow.h.

4.2.4.34 double GImShow::ry [private]

Definition at line 29 of file gimshow.h.

4.2.4.35 int GImShow::size [private]

Definition at line 31 of file gimshow.h.

4.2.4.36 int GImShow::temp [private]

Definition at line 30 of file gimshow.h.

4.2.4.37 QRgb GImShow::temprgb [private]

Definition at line 24 of file gimshow.h.

4.2.4.38 double GImShow::theta [private]

Definition at line 29 of file gimshow.h.

4.2.4.39 int GImShow::totprogsteps [private]

Definition at line 30 of file gimshow.h.

4.2.4.40 double GImShow::u0 [private]

Definition at line 29 of file gimshow.h.

4.2.4.41 double GImShow::v0 [private]

Definition at line 29 of file gimshow.h.

4.2.4.42 GrayProcessingImage* GImShow::viewimage [private]

Definition at line 21 of file gimshow.h.

4.2.4.43 double GImShow::x0 [private]

Definition at line 29 of file gimshow.h.

4.2.4.44 double GImShow::xoff [private]

Definition at line 29 of file gimshow.h.

4.2.4.45 double GImShow::y0 [private]

Definition at line 29 of file gimshow.h.

4.2.4.46 double GImShow::yoff [private]

Definition at line 29 of file gimshow.h.

The documentation for this class was generated from the following files:

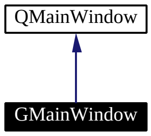
- **gimshow.h**
- **gimshow.cc**

4.3 GMainWindow Class Reference

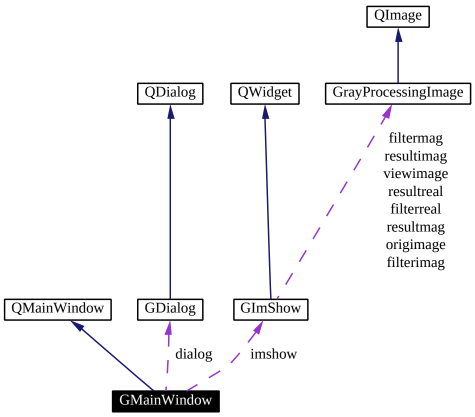
The main class for the QtGabor2D application.

`#include <gmainwindow.h>`

Inheritance diagram for GMainWindow:



Collaboration diagram for GMainWindow:



Public Methods

- **GMainWindow** ()

  Constructor for the GMainWindow class.

- **~GMainWindow** ()

Protected Methods

- void **paintEvent** (QPaintEvent *e)

Private Methods

- void **setFileMenuItems** (bool open, bool save, bool quit)
- void **setViewMenuItems** (bool origimage, bool filterreal, bool filterimag, bool filtermag, bool resultreal, bool resultimag, bool resultmag)
- void **setSettingsMenuItems** (bool filter)

Private Attributes

- QPainter* **paint**
- **GImShow* imshow**
- **GDialog* dialog**
- QMenuBar* **_menubar**
- QPopupMenu* **_filemenu**
- QPopupMenu* **_helpmenu**
- QPopupMenu* **_viewmenu**
- QPopupMenu* **_settingsmenu**
- int **openID**
- int **saveID**
- int **quitID**
- int **origimageID**
- int **filterrealID**
- int **filterimagID**
- int **filtermagID**
- int **resultrealID**
- int **resultimagID**
- int **resultmagID**
- int **filterID**

Private Slots

- void **slotAboutQtGabor2D** ()

  Simple dialogbox on the Help menu.

- void **slotOpenFile** ()
- void **slotSaveFile** ()
- void **slotOpenGDialog** ()
- void **slotFullViewMenu** ()

4.3.1 Detailed Description

The main class for the QtGabor2D application.

This class owns the menubar and the qimage displayed. By using the menus, the user can control what is displayed. The file menu lets the user open and save images, the view menu controls the viewer, but the most important functionality is in the settings menu, where the user can set the parameters for the Gabor filters.

Definition at line 11 of file gmainwindow.h.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 GMainWindow::GMainWindow ()

Constructor for the GMainWindow class.

Generates the menus.

Definition at line 13 of file gmainwindow.cc.

4.3.2.2 GMainWindow::~GMainWindow ()

4.3.3 Member Function Documentation

4.3.3.1 void GMainWindow::paintEvent (QPaintEvent * *e*) `[protected]`

4.3.3.2 void GMainWindow::setFileMenuItems (bool *open*, bool *save*, bool *quit*) `[private]`

Referenced by GMainWindow().

4.3.3.3 void GMainWindow::setSettingsMenuItems (bool *filter*) `[private]`

Referenced by GMainWindow().

4.3.3.4 void GMainWindow::setViewMenuItems (bool *origimage*, bool *filterreal*, bool *filterimag*, bool *filtermag*, bool *resultreal*, bool *resultimag*, bool *resultmag*) `[private]`

Referenced by GMainWindow().

4.3.3.5 void GMainWindow::slotAboutQtGabor2D () `[private, slot]`

Simple dialogbox on the Help menu.

Definition at line 70 of file gmainwindow.cc.

Referenced by GMainWindow().

4.3.3.6 void GMainWindow::slotFullViewMenu () `[private, slot]`

Referenced by GMainWindow().

4.3.3.7 void GMainWindow::slotOpenFile () `[private, slot]`

Referenced by GMainWindow().

4.3.3.8 void GMainWindow::slotOpenGDialog () `[private, slot]`

Referenced by GMainWindow().

4.3.3.9 void GMainWindow::slotSaveFile () `[private, slot]`

Referenced by GMainWindow().

4.3.4 Member Data Documentation

4.3.4.1 QPopupMenu* GMainWindow::_filemenu `[private]`

Definition at line 20 of file gmainwindow.h.

4.3.4.2 QPopupMenu* GMainWindow::_helpmenu [private]

Definition at line 21 of file gmainwindow.h.

4.3.4.3 QMenuBar* GMainWindow::_menubar [private]

Definition at line 19 of file gmainwindow.h.

4.3.4.4 QPopupMenu* GMainWindow::_settingsmenu [private]

Definition at line 23 of file gmainwindow.h.

4.3.4.5 QPopupMenu* GMainWindow::_viewmenu [private]

Definition at line 22 of file gmainwindow.h.

4.3.4.6 GDialog* GMainWindow::dialog [private]

Definition at line 18 of file gmainwindow.h.

4.3.4.7 int GMainWindow::filterID [private]

Definition at line 24 of file gmainwindow.h.

4.3.4.8 int GMainWindow::filterimagID [private]

Definition at line 24 of file gmainwindow.h.

4.3.4.9 int GMainWindow::filtermagID [private]

Definition at line 24 of file gmainwindow.h.

4.3.4.10 int GMainWindow::filterrealID [private]

Definition at line 24 of file gmainwindow.h.

4.3.4.11 GImShow* GMainWindow::imshow [private]

Definition at line 17 of file gmainwindow.h.

4.3.4.12 int GMainWindow::openID [private]

Definition at line 24 of file gmainwindow.h.

4.3.4.13 int GMainWindow::origimageID [private]

Definition at line 24 of file gmainwindow.h.

4.3.4.14 QPainter* GMainWindow::paint [private]

Definition at line 16 of file gmainwindow.h.

4.3.4.15 int GMainWindow::quitID [private]

Definition at line 24 of file gmainwindow.h.

4.3.4.16 int GMainWindow::resultimagID [private]

Definition at line 24 of file gmainwindow.h.

4.3.4.17 int GMainWindow::resultmagID [private]

Definition at line 24 of file gmainwindow.h.

4.3.4.18 int GMainWindow::resultrealID [private]

Definition at line 24 of file gmainwindow.h.

4.3.4.19 int GMainWindow::saveID [private]

Definition at line 24 of file gmainwindow.h.

The documentation for this class was generated from the following files:

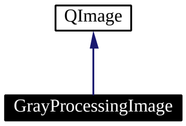
- **gmainwindow.h**
- **gmainwindow.cc**

4.4 GrayProcessingImage Class Reference

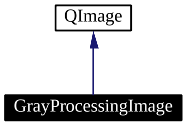
This class defines various image processing operations.

`#include <grayprocessingimage.h>`

Inheritance diagram for GrayProcessingImage:



Collaboration diagram for GrayProcessingImage:



Public Types

- enum **PaddingType** { **zeroPadding**, **copyPadding**, **wrapPadding** }
  Handy stuff.
- enum **ImageType** { **realImage**, **imagImage**, **magImage** }

Public Methods

- **GrayProcessingImage** ()
  Constructors.
- **GrayProcessingImage** (**QImage** image)
  Constructor taking a **QImage** (p. 33) *as argument. Converts the image to grayscale.*
- **GrayProcessingImage** (const GrayProcessingImage &image)
  Copy constructor. Send the image to **QImage** (p. 33).
- **GrayProcessingImage** (int width, int height, int depth=8, int numColors=256, Endian bitOrder=IgnoreEndian)
  Constructor taking size, depth etc. as arguments. Passes to **QImage** (p. 33).
- **GrayProcessingImage** (const QSize &size, int depth=8, int numColors=256, Endian bit-Order=IgnoreEndian)
  Constructor taking size, depth etc. as arguments. Passes to **QImage** (p. 33).
- bool **load** (const QString &fileName, const char *format=0)
  Load image from file, convert to grayscale.

- GrayProcessingImage **bilinearResize** (int h, int w)

  Image operations.
- GrayProcessingImage **intensityPlane** ()

  Convert image to its own intensity plane.
- GrayProcessingImage **subtractIntensityPlane** ()

  Subtract the image's intensity plane.
- GrayProcessingImage **histogramEqualize** ()

  Perform histogram equalization.
- GrayProcessingImage **convolution3x3** (double matrix[3][3], int add=0, **PaddingType** p=zeroPadding)

  Applies 3x3 convolution given by matrix on image.
- GrayProcessingImage **average** (vector< GrayProcessingImage > images)

  Make the average of the given images a new image.
- GrayProcessingImage **faceNormalize** (int lx, int ly, int rx, int ry, int pctabove=30, int pctofwidth=60, int neww=20, int newh=20)

  Normalize (geometrically) image of face with given eye coordinates.
- GrayProcessingImage **normal1** ()

  Calculate first component of image normal.
- GrayProcessingImage **normal2** ()

  Calculate second component of image normal.
- pair<GrayProcessingImage, GrayProcessingImage> **dirImage** (int windowSize)

  Returns two images describing the directional image.
- GrayProcessingImage **visualiseDirImage** (int windowSize)

  Creates an image that visualises the directional image.
- GrayProcessingImage **visualiseDirImage** (pair< GrayProcessingImage, GrayProcessing-Image > dir)

  Creates an image that visualises the directional image.
- double **average** ()

  Return the average pixel value in the image.
- double **variance** ()

  Return the variance of the pixel values in the image.
- double **correlcoeff** (GrayProcessingImage image)

  Return the correlation coefficient between two images.
- void **setImage** (complex< double > **img, int w, int h, double max, double min, int i)

  Changes the content of the image.

Private Methods

- void **fillGrayPalette** ()

  Fill the palette with 256 gray values.

- void **toGrayscale** ()

  Converts the **QImage** (p. 33) *in any format to a similar 8 bpp grayscale image.*

- void **calculateHistogram** (unsigned long ∗histogram)

  Generate histogram of image.

- void **calculateIntensityPlane** (double ∗a, double ∗b, double ∗c)

  Calculate coefficients for intensity plane.

- uchar **interpolate** (double x, double y)

  Bilinear interpolation of pixel values.

4.4.1 Detailed Description

This class defines various image processing operations.

The class derives from the **QImage** (p. 33) class. When the image is instantiated, it is converted to gray scale once and for all.

Definition at line 11 of file grayprocessingimage.h.

4.4.2 Member Enumeration Documentation

4.4.2.1 enum GrayProcessingImage::ImageType

Enumeration values:

realImage

imagImage

magImage

Definition at line 31 of file grayprocessingimage.h.

4.4.2.2 enum GrayProcessingImage::PaddingType

Handy stuff.

Enumeration values:

zeroPadding

copyPadding

wrapPadding

Definition at line 30 of file grayprocessingimage.h.

4.4.3 Constructor & Destructor Documentation

4.4.3.1 GrayProcessingImage::GrayProcessingImage ()

Constructors.

Definition at line 11 of file grayprocessingimage.cc.

Referenced by visualiseDirImage().

4.4.3.2 GrayProcessingImage::GrayProcessingImage (QImage *image*)

Constructor taking a **QImage** (p. 33) as argument. Converts the image to grayscale.

Definition at line 24 of file grayprocessingimage.cc.

4.4.3.3 GrayProcessingImage::GrayProcessingImage (const GrayProcessingImage & *image*)

Copy constructor. Send the image to **QImage** (p. 33).

Definition at line 18 of file grayprocessingimage.cc.

4.4.3.4 GrayProcessingImage::GrayProcessingImage (int *width*, int *height*, int *depth* = 8, int *numColors* = 256, Endian *bitOrder* = IgnoreEndian)

Constructor taking size, depth etc. as arguments. Passes to **QImage** (p. 33).

Definition at line 40 of file grayprocessingimage.cc.

4.4.3.5 GrayProcessingImage::GrayProcessingImage (const QSize & *size*, int *depth* = 8, int *numColors* = 256, Endian *bitOrder* = IgnoreEndian)

Constructor taking size, depth etc. as arguments. Passes to **QImage** (p. 33).

Definition at line 31 of file grayprocessingimage.cc.

4.4.4 Member Function Documentation

4.4.4.1 double GrayProcessingImage::average ()

Return the average pixel value in the image.

Definition at line 520 of file grayprocessingimage.cc.

Referenced by correlcoeff(), and variance().

4.4.4.2 GrayProcessingImage GrayProcessingImage::average (vector< GrayProcessingImage > *images*)

Make the average of the given images a new image.

Definition at line 329 of file grayprocessingimage.cc.

4.4.4.3 GrayProcessingImage GrayProcessingImage::bilinearResize (int *h*, int *w*)

Image operations.

For enlargements or small reductions this is better than the function QImage::smoothScale(), Copyright (C) 1989, 1991 by Jef Poskanzer.

Definition at line 166 of file grayprocessingimage.cc.

4.4.4.4 void GrayProcessingImage::calculateHistogram (unsigned long * *histogram*) `[private]`

Generate histogram of image.

Definition at line 138 of file grayprocessingimage.cc.

Referenced by histogramEqualize().

4.4.4.5 void GrayProcessingImage::calculateIntensityPlane (double * *a*, double * *b*, double * *c*) `[private]`

Calculate coefficients for intensity plane.

Definition at line 111 of file grayprocessingimage.cc.

Referenced by intensityPlane(), and subtractIntensityPlane().

4.4.4.6 GrayProcessingImage GrayProcessingImage::convolution3x3 (double *matrix*[3][3], int *add* = 0, PaddingType *pad* = zeroPadding)

Applies 3x3 convolution given by matrix on image.

Last parameter indicate type of padding; one of the following: **GrayProcessingImage::zeroPadding** (p. 26), **GrayProcessingImage::copyPadding** (p. 26) or **GrayProcessingImage::wrapPadding** (p. 26). Only zeropadding is implemented so far.

Definition at line 243 of file grayprocessingimage.cc.

4.4.4.7 double GrayProcessingImage::correlcoeff (GrayProcessingImage *image*)

Return the correlation coefficient between two images.

As defined in Gonzales & Woods p. 584, eq. 9.3-8. Assumes that the two images have the same dimensions; if not, the result is undefined. In order to make the correlation act as a distance, it is inverted and scaled such that correlation 1 corresponds to a return value '0' (no distance between images, and correlation -1 corresponds to 1 ('100% distance').

Definition at line 551 of file grayprocessingimage.cc.

4.4.4.8 pair< GrayProcessingImage, GrayProcessingImage > GrayProcessingImage::dirImage (int *windowSize*)

Returns two images describing the directional image.

The parameters windowSize gives the size of the NxN average frame. The first of the returned images is the direction (0: -90, 255: +90), and the second is a weighted sum of the contrast and the consistency.

Definition at line 423 of file grayprocessingimage.cc.

Referenced by visualiseDirImage().

4.4.4.9 GrayProcessingImage GrayProcessingImage::faceNormalize (int *lx*, int *ly*, int *rx*, int *ry*, int *pctabove* = 30, int *pctofwidth* = 60, int *neww* = 20, int *newh* = 20)

Normalize (geometrically) image of face with given eye coordinates.

Normalization involves rotating the image so that the eyes are located on an horizontal line, scaling the image so that the distance between the eyes are fixed, and making a new image of given size and position of the face. The other parameters mean: pctabove - the amount of the image above the eyeline (in pct), pctofwidth - the ratio of distance between eyes to the width of the image (in pct), neww & newh - the dimensions of the new image.

Definition at line 355 of file grayprocessingimage.cc.

4.4.4.10 void GrayProcessingImage::fillGrayPalette () `[private]`

Fill the palette with 256 gray values.

Definition at line 51 of file grayprocessingimage.cc.

Referenced by intensityPlane(), setImage(), and toGrayscale().

4.4.4.11 GrayProcessingImage GrayProcessingImage::histogramEqualize ()

Perform histogram equalization.

Definition at line 214 of file grayprocessingimage.cc.

4.4.4.12 GrayProcessingImage GrayProcessingImage::intensityPlane ()

Convert image to its own intensity plane.

Definition at line 184 of file grayprocessingimage.cc.

4.4.4.13 uchar GrayProcessingImage::interpolate (double *x*, double *y*) `[private]`

Bilinear interpolation of pixel values.

Definition at line 59 of file grayprocessingimage.cc.

Referenced by bilinearResize(), and faceNormalize().

4.4.4.14 bool GrayProcessingImage::load (const QString & *fileName*, const char * *format* = 0)

Load image from file, convert to grayscale.

Definition at line 155 of file grayprocessingimage.cc.

4.4.4.15 GrayProcessingImage GrayProcessingImage::normal1 ()

Calculate first component of image normal.

Definition at line 390 of file grayprocessingimage.cc.

Referenced by dirImage().

4.4.4.16 GrayProcessingImage GrayProcessingImage::normal2 ()

Calculate second component of image normal.

Definition at line 404 of file grayprocessingimage.cc.

Referenced by dirImage().

4.4.4.17 void GrayProcessingImage::setImage (complex< double > ** *img*, int *w*, int *h*, double *max*, double *min*, int *i*)

Changes the content of the image.

From a complex double array, sets the content of the image to either the real, imaginary or magnitude of the array.

Definition at line 570 of file grayprocessingimage.cc.

Referenced by GImShow::slotSetFilter().

4.4.4.18 GrayProcessingImage GrayProcessingImage::subtractIntensityPlane ()

Subtract the image's intensity plane.

Definition at line 199 of file grayprocessingimage.cc.

4.4.4.19 void GrayProcessingImage::toGrayscale () `[private]`

Converts the **QImage** (p. 33) in any format to a similar 8 bpp grayscale image.

Currently, only 8 bpp and 32 bpp images are supported.

Definition at line 80 of file grayprocessingimage.cc.

Referenced by GrayProcessingImage(), and load().

4.4.4.20 double GrayProcessingImage::variance ()

Return the variance of the pixel values in the image.

Definition at line 531 of file grayprocessingimage.cc.

Referenced by correlcoeff().

4.4.4.21 GrayProcessingImage GrayProcessingImage::visualiseDirImage (pair< GrayProcessingImage, GrayProcessingImage > *dir*)

Creates an image that visualises the directional image.

Definition at line 495 of file grayprocessingimage.cc.

4.4.4.22 GrayProcessingImage GrayProcessingImage::visualiseDirImage (int *windowSize*)

Creates an image that visualises the directional image.

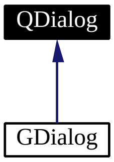
Definition at line 514 of file grayprocessingimage.cc.

The documentation for this class was generated from the following files:

- **grayprocessingimage.h**
- **grayprocessingimage.cc**

4.5 QDialog Class Reference

Inheritance diagram for QDialog:

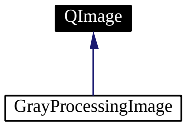


The documentation for this class was generated from the following file:

- **gdialog.h**

4.6 QImage Class Reference

Inheritance diagram for QImage:

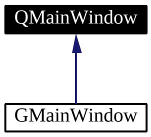


The documentation for this class was generated from the following file:

- **grayprocessingimage.h**

4.7 QMainWindow Class Reference

Inheritance diagram for QMainWindow:



The documentation for this class was generated from the following file:

- **gmainwindow.h**

4.8 QWidget Class Reference

Inheritance diagram for QWidget:

The documentation for this class was generated from the following file:

- **gimshow.h**

Chapter 5

QtGabor2D File Documentation

5.1 commongrayproc.h File Reference

Functions

- uchar **toChar** (NumberType d)

  Convert double to uchar in the range 0..255 by clamping if out of range.

- NumberType **sqr** (NumberType d)

  Returns the square of the given number.

- NumberType **absnum** (NumberType d)

  Returns the absolute value of argument.

- NumberType **deltaAngle** (NumberType phi1, NumberType phi2)

  Returns the undirected angle between the two given directions.

5.1.1 Function Documentation

5.1.1.1 NumberType absnum (NumberType *d*)

Returns the absolute value of argument.

Definition at line 23 of file commongrayproc.h.

Referenced by deltaAngle().

5.1.1.2 NumberType deltaAngle (NumberType *phi1*, NumberType *phi2*)

Returns the undirected angle between the two given directions.

The angles are defined such that 0 corresponds to -90, and 255 to 90.

Definition at line 34 of file commongrayproc.h.

5.1.1.3 NumberType sqr (NumberType *d*)

Returns the square of the given number.

Definition at line 16 of file commongrayproc.h.

5.1.1.4 uchar toChar (NumberType *d*)

Convert double to uchar in the range 0..255 by clamping if out of range.

Works for all number types

Definition at line 7 of file commongrayproc.h.

Referenced by GrayProcessingImage::convolution3x3(), GrayProcessingImage::dirImage(), GrayProcessingImage::intensityPlane(), GrayProcessingImage::interpolate(), and GrayProcessingImage::subtractIntensityPlane().

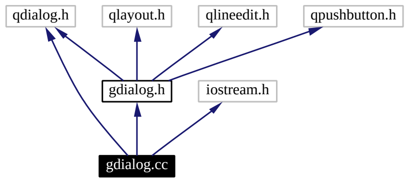
5.2 gdialog.cc File Reference

`#include "gdialog.h"`

`#include <qdialog.h>`

`#include <iostream.h>`

Include dependency graph for gdialog.cc:



5.3 gdialog.h File Reference

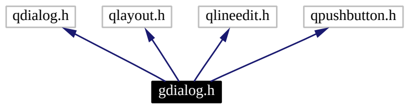
`#include <qdialog.h>`

`#include <qlayout.h>`

`#include <qlineedit.h>`

`#include <qpushbutton.h>`

Include dependency graph for gdialog.h:



Compounds

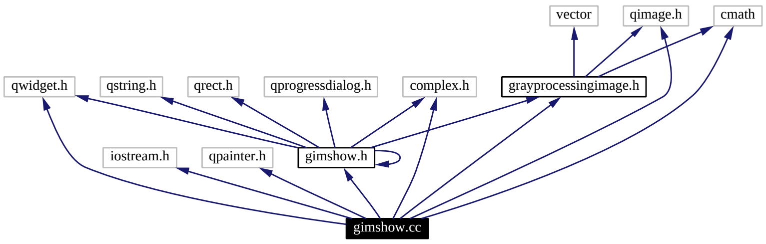
- class **GDialog**

  The class for the dialogbox for the setting the Gabor parameters.

5.4 gimshow.cc File Reference

```
#include <qpainter.h>
#include <qwidget.h>
#include <qimage.h>
#include <iostream.h>
#include <cmath>
#include <complex.h>
#include "gimshow.h"
#include "grayprocessingimage.h"
```

Include dependency graph for gimshow.cc:



5.5 gimshow.h File Reference

`#include <qwidget.h>`

`#include <qstring.h>`

`#include <qrect.h>`

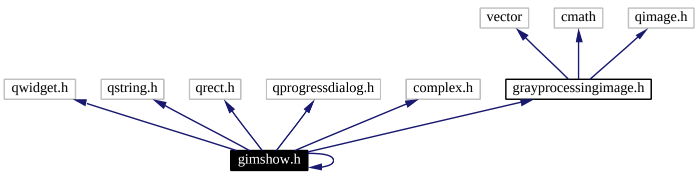
`#include <qprogressdialog.h>`

`#include <complex.h>`

`#include "gimshow.h"`

`#include "grayprocessingimage.h"`

Include dependency graph for gimshow.h:



Compounds

- class **GImShow**

  The imshow class for the QtGabor2D application.

Defines

- #define **MINWIDGETSIZE** 200

5.5.1 Define Documentation

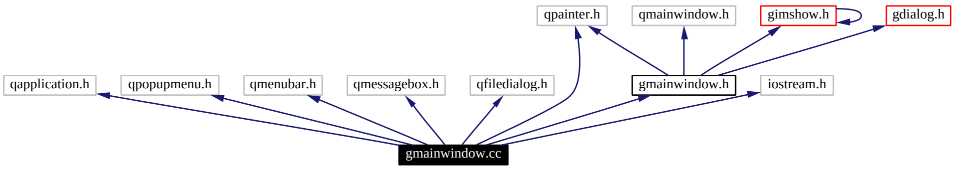
5.5.1.1 #define MINWIDGETSIZE 200

Definition at line 12 of file gimshow.h.

5.6 gmainwindow.cc File Reference

```
#include <qapplication.h>
#include <qpopupmenu.h>
#include <qmenubar.h>
#include <qmessagebox.h>
#include <qfiledialog.h>
#include <qpainter.h>
#include <iostream.h>
#include "gmainwindow.h"
```

Include dependency graph for gmainwindow.cc:



5.7 gmainwindow.h File Reference

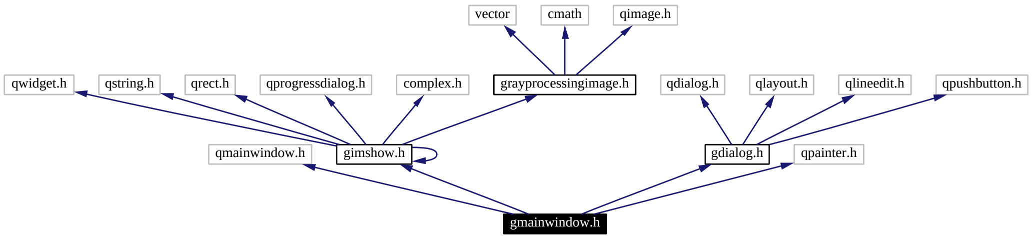
`#include <qmainwindow.h>`

`#include "gimshow.h"`

`#include "gdialog.h"`

`#include <qpainter.h>`

Include dependency graph for gmainwindow.h:



Compounds

- class **GMainWindow**

  The main class for the QtGabor2D application.

5.8 grayprocessingimage.cc File Reference

`#include <cmath>`

`#include <complex>`

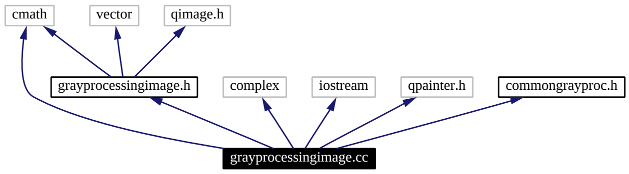
`#include <iostream>`

`#include <qpainter.h>`

`#include "grayprocessingimage.h"`

`#include "commongrayproc.h"`

Include dependency graph for grayprocessingimage.cc:



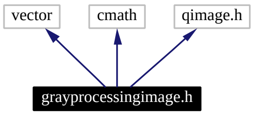
5.9 grayprocessingimage.h File Reference

`#include <vector>`

`#include <cmath>`

`#include <qimage.h>`

Include dependency graph for grayprocessingimage.h:



Compounds

- class **GrayProcessingImage**

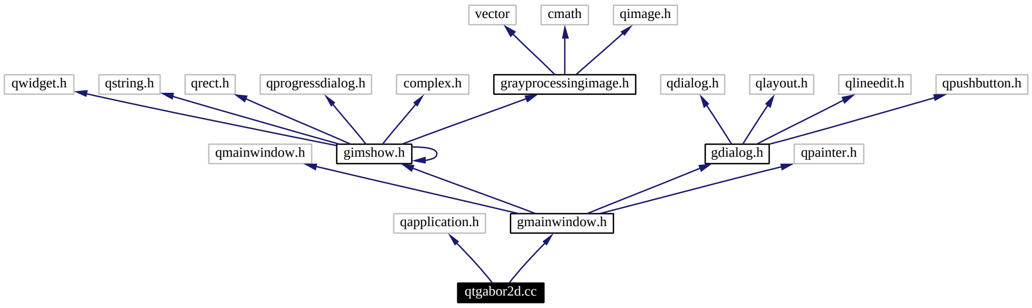
  This class defines various image processing operations.

5.10 qtgabor2d.cc File Reference

`#include <qapplication.h>`

`#include "gmainwindow.h"`

Include dependency graph for qtgabor2d.cc:



Functions

- int **main** (int argc, char **argv)

  QtGabor2D - a program for manual inspection of 2D Gabor filtering.

5.10.1 Function Documentation

5.10.1.1 int main (int *argc*, char ** *argv*)

QtGabor2D - a program for manual inspection of 2D Gabor filtering.

QtGabor2D is a simple GUI program for testing the applicability of 2D Gabor filters for filtering digital images.

Definition at line 6 of file qtgabor2d.cc.

Index