Volume Reconstruction

1.0

Generated by Doxygen 1.8.0

Mon Jul 23 2012 15:08:35

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Namespace Documentation

4.1 Ui Namespace Reference

Chapter 5

Class Documentation

5.1 Calibration Class Reference

Implements LSQRRecepies methods.

```
#include <Calibration.h>
```

Public Types

- typedef lsqrRecipes::SingleUnknownPointTargetUSCalibrationParametersEstimator::DataType DataType

Public Member Functions

- void InsertTransformations (vnl_matrix< double > rotationMatrix, vnl_vector< double > translation)

    insert the rotation matrix of an image to transformations
- void ClearTransformations ()

    clear transformations
- void InsertImagePoints (double p[2])

    insert the crosswire point of an image to imagePoints
- void ClearImagePoints ()

    clear imagePoints
- bool Calibrate ()

    estimate calibration parameters
- std::vector< double > getEstimatedUSCalibrationParameters ()

Static Public Member Functions

- static Calibration $*$ New ()

    Constructor of the class.

Private Attributes

- std::vector< lsqrRecipes::Frame > transformations
- std::vector< lsqrRecipes::Point2D > imagePoints

    contains the crosswire point in all images
- std::vector< DataType > data

    contain the data of all images
- std::vector< double > estimatedUSCalibrationParameters

5.1.1 Detailed Description

Implements LSQRRecepies methods.

This classs have the calibration methods implemented in LSQRRecipes to calibrate an Ultra Sound Probe with a cross wire phantom.

5.1.2 Member Typedef Documentation

5.1.2.1 typedef lsqrRecipes::SingleUnknownPointTargetUSCalibrationParametersEstimator::DataType **Calibration::DataType**

5.1.3 Member Function Documentation

5.1.3.1 bool **Calibration::Calibrate ( )**

estimate calibration parameters

5.1.3.2 void **Calibration::ClearImagePoints ( )**

clear imagePoints

5.1.3.3 void **Calibration::ClearTransformations ( )**

clear transformations

5.1.3.4 std::vector< double > **Calibration::getEstimatedUSCalibrationParameters ( )**

5.1.3.5 void **Calibration::InsertImagePoints (** double *p[2]* **)**

insert the crosswire point of an image to imagePoints

5.1.3.6 void **Calibration::InsertTransformations (** vnl_matrix< double > *rotationMatrix,* vnl_vector< double > *translation* **)**

insert the rotation matrix of an image to transformations

5.1.3.7 static **Calibration*** **Calibration::New ( )** `[inline, static]`

Constructor of the class.

5.1.4 Member Data Documentation

5.1.4.1 std::vector< **DataType** > **Calibration::data** `[private]`

contain the data of all images

5.1.4.2 std::vector<double> **Calibration::estimatedUSCalibrationParameters** `[private]`

5.1.4.3 std::vector<lsqrRecipes::Point2D> **Calibration::imagePoints** `[private]`

contains the crosswire point in all images

5.1.4.4 std::vector<IsqrRecipes::Frame> Calibration::transformations [private]

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

- Calibration.h
- Calibration.cpp

5.2 CropImagesWidget Class Reference

Crop images.

`#include <CropImagesWidget.h>`

Public Member Functions

- CropImagesWidget (QWidget ∗parent=0)
- ∼CropImagesWidget ()
- void setImageStack (std::vector< vtkSmartPointer< vtkImageData > > imageStack)

  Set this stack of vtkImageData.
- void setImage (vtkSmartPointer< vtkImageData > image)

  Set this vtkImageData.
- void setMainWindow (MainWindow ∗mainwindow)

  Set the window to display the crop images.

Private Slots

- void crop ()

  calls the crop method when the crop buttom is clicked
- void save ()

  Save the cropped images in a folder.

Private Member Functions

- vtkSmartPointer< vtkImageData > cropProbeImage (vtkSmartPointer< vtkImageData > image, int depthType)

  Crop ultrasound image depnding of the depth type.

Private Attributes

- Ui::CropImagesWidget ∗ ui
- bool workWithStack

  if there are multiple images to work with
- std::vector< vtkSmartPointer < vtkImageData > > imageStack

  an Array of vtkImageData to work
- MainWindow ∗ mainWindow
- vtkSmartPointer< vtkImageData > image

  the vtkImageData to work
- vtkSmartPointer< vtkImageData > cropImage

  the cropped image
- std::vector< vtkSmartPointer < vtkImageData > > cropStack

  the cropped imageStack

5.2.1 Detailed Description

Crop images.

This class crop one or multiple images deppendng on which range of the ultrasound machine was used. It can crop images in ranges 4, 5, 6 and 8. Images can be saved in a folder or used to calibrate the US probe.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 CropImagesWidget::CropImagesWidget (QWidget * *parent =* 0) [explicit]

5.2.2.2 CropImagesWidget::~CropImagesWidget ( )

5.2.3 Member Function Documentation

5.2.3.1 void CropImagesWidget::crop ( ) [private, slot]

calls the crop method when the crop buttom is clicked

5.2.3.2 vtkSmartPointer< vtkImageData > CropImagesWidget::cropProbeImage (vtkSmartPointer< vtkImageData > *image*, int *depthType*) [private]

Crop ultrasound image depnding of the depth type.

5.2.3.3 void CropImagesWidget::save ( ) [private, slot]

Save the cropped images in a folder.

5.2.3.4 void CropImagesWidget::setImage (vtkSmartPointer< vtkImageData > *image*)

Set this vtkImageData.

Parameters

in	*a*	smart Pointer of vtkImageData

5.2.3.5 void CropImagesWidget::setImageStack (std::vector< vtkSmartPointer< vtkImageData > > *imageStack*)

Set this stack of vtkImageData.

Parameters

in	*a*	std Vector of vtkImageData

5.2.3.6 void CropImagesWidget::setMainWindow (MainWindow * *mainwindow*)

Set the window to display the crop images.

5.2.4 Member Data Documentation

5.2.4.1 vtkSmartPointer<vtkImageData> CropImagesWidget::cropImage [private]

the cropped image

5.2.4.2 std::vector< vtkSmartPointer<vtkImageData> > CropImagesWidget::cropStack [private]

the cropped imageStack

5.2.4.3 vtkSmartPointer<vtkImageData> CropImagesWidget::image [private]

the vtkImageData to work

5.2.4.4 std::vector< vtkSmartPointer<vtkImageData> > CropImagesWidget::imageStack [private]

an Array of vtkImageData to work

5.2.4.5 MainWindow* CropImagesWidget::mainWindow [private]

the main window to call it

5.2.4.6 Ui::CropImagesWidget* CropImagesWidget::ui [private]

5.2.4.7 bool CropImagesWidget::workWithStack [private]

if there are multiple images to work with

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

- CropImagesWidget.h
- CropImagesWidget.cpp

5.3 MainWindow Class Reference

Main window for user interaction.

`#include <mainwindow.h>`

Public Member Functions

- MainWindow (QWidget *parent=0)
- ~MainWindow ()
- void addLogText (QString str)
- QVTKImageWidget * getDisplayWidget ()

  return this display widget

Private Slots

- void addImages ()

  Add image folder to application.
- void displaySelectedImage (int idx)

  Display selected image with the image slider.

- void setSelectedOpacity (int idx)

  Set selected opacity for the displayed volume.
- void probeCalibration ()

  Implements a ultrasound 3D probe calibration, for navigate with the probe.
- void openVolumeData ()

  Set the image, rotation, translation and calibration parameters file name.
- void print ()

  Print message in logger.
- void volumeReconstruction ()

  Calls the VolumeReconstructionWidget.h to generate a new volume ∗with the loaded data.
- void openVolume ()

  Set a volume filename.
- void cropImages ()

Private Attributes

- Ui::MainWindow ∗ ui
- QStringList imagesFilenames

  The filename of each selected image.
- QStringList volumeImagesFilenames

  The filenames of each selected volume image.
- QString volumeRotationData

  The filename of the rotation data for each image.
- QString volumeTranslationData

  The filename of the translation data for each image.
- QString volumeCalibrationData

  The filename of the estimated parameters in the calibration.
- QString volumeFilename

  The filename of the selected volume.
- QString textOnTextArea
- QVTKImageWidget ∗ displayWidget
- vtkSmartPointer
  < vtkEventQtSlotConnect > Connections

5.3.1 Detailed Description

Main window for user interaction.

This class has the main interaction function to load volume images.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 MainWindow::MainWindow (QWidget ∗ *parent* = 0) [explicit]

5.3.2.2 MainWindow::∼MainWindow ( )

5.3.3 Member Function Documentation

5.3.3.1 void MainWindow::addImages ( ) [private, slot]

Add image folder to application.

5.3.3.2 void MainWindow::addLogText (QString *str*)

5.3.3.3 void MainWindow::cropImages ( ) [private, slot]

5.3.3.4 void MainWindow::displaySelectedImage (int *idx*) [private, slot]

Display selected image with the image slider.

5.3.3.5 QVTKImageWidget ∗ MainWindow::getDisplayWidget ( )

return this display widget

Parameters

out	*this*	display widget

5.3.3.6 void MainWindow::openVolume ( ) [private, slot]

Set a volume filename.

5.3.3.7 void MainWindow::openVolumeData ( ) [private, slot]

Set the image, rotation, translation and calibration parameters file name.

5.3.3.8 void MainWindow::print ( ) [private, slot]

Print message in logger.

5.3.3.9 void MainWindow::probeCalibration ( ) [private, slot]

Implements a ultrasound 3D probe calibration, for navigate with the probe.

5.3.3.10 void MainWindow::setSelectedOpacity (int *idx*) [private, slot]

Set selected opacity for the displayed volume.

5.3.3.11 void MainWindow::volumeReconstruction ( ) [private, slot]

Calls the VolumeReconstructionWidget.h to generate a new volume ∗with the loaded data.

5.3.4 Member Data Documentation

5.3.4.1 vtkSmartPointer<vtkEventQtSlotConnect> MainWindow::Connections [private]

5.3.4.2 QVTKImageWidget∗ MainWindow::displayWidget [private]

Central widget for display image purposes

5.3.4.3 QStringList MainWindow::imagesFilenames [private]

The filename of each selected image.

5.3.4.4 QString MainWindow::textOnTextArea `[private]`

5.3.4.5 Ui::MainWindow∗ MainWindow::ui `[private]`

5.3.4.6 QString MainWindow::volumeCalibrationData `[private]`

The filename of the estimated parameters in the calibration.

5.3.4.7 QString MainWindow::volumeFilename `[private]`

The filename of the selected volume.

5.3.4.8 QStringList MainWindow::volumeImagesFilenames `[private]`

The filenames of each selected volume image.

5.3.4.9 QString MainWindow::volumeRotationData `[private]`

The filename of the rotation data for each image.

5.3.4.10 QString MainWindow::volumeTranslationData `[private]`

The filename of the translation data for each image.

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

- mainwindow.h
- mainwindow.cpp

5.4 ProbeCalibrationWidget Class Reference

Obtain data for calibration process.

`#include <ProbeCalibrationWidget.h>`

Public Member Functions

- ProbeCalibrationWidget (QWidget ∗parent=0)
- virtual ∼ProbeCalibrationWidget ()
- void setImageStack (std::vector< vtkSmartPointer< vtkImageData > > imageStack)

  Set this stack of vtkImageData.
- void setImage (vtkSmartPointer< vtkImageData > image)

  Set this vtkImageData.
- void setMainWindow (MainWindow ∗mainwindow)

Private Slots

- void crop ()

  crop the images to delete the extra information
- void getCoordinates ()

  get the cross point coordinates

- void calibrate ()

  Calls Calibrate.h to estimate the calibration parameters.
- void loadRotationsFile ()

  Load the rotations file.
- void loadTranslationsFile ()

  Load the translation file.
- void saveCalibration ()

  Save the Estimated Parameters in a .txt file.

Private Member Functions

- vtkSmartPointer< vtkImageData > cropProbeImage (vtkSmartPointer< vtkImageData > image, int depthType)

Private Attributes

- bool workWithStack
- std::vector< vtkSmartPointer
  < vtkImageData > > imageStack

  an Array of vtkImageData to work
- vnl_matrix< double > translations

  a vnl_matrix to store the translations of each image given by the tracker
- vnl_matrix< double > rotations

  a vnl_matrix to store the rotations of each image given by the tracker
- float rotations_2 [7][4]
- MainWindow ∗ mainWindow
- vtkSmartPointer< vtkImageData > image

  the vtkImageData to work
- vnl_matrix< double > coords

  a vnl_matrix to store the selected coordinates on each image
- std::vector< double > calibrationParameters

  the estimate calibration parameters by Calibration.h

5.4.1 Detailed Description

Obtain data for calibration process.

This class obtain the necesary data to use the Calibrtion class. It let the user to set the croos wire point on the images, load Rotation and Translation Data. It also allows the user to save the calibration estimated paraeters in a .txt file

5.4.2 Constructor & Destructor Documentation

5.4.2.1 ProbeCalibrationWidget::ProbeCalibrationWidget (QWidget ∗ *parent* = 0)

Constructor

5.4.2.2 ProbeCalibrationWidget::∼ProbeCalibrationWidget ( ) [virtual]

5.4.3 Member Function Documentation

5.4.3.1 void ProbeCalibrationWidget::calibrate ( ) [private, slot]

Calls Calibrate.h to estimate the calibration parameters.

5.4.3.2 void ProbeCalibrationWidget::crop ( ) [private, slot]

crop the images to delete the extra information

5.4.3.3 vtkSmartPointer< vtkImageData > ProbeCalibrationWidget::cropProbeImage (vtkSmartPointer< vtkImageData > *image,* int *depthType*) [private]

Crop ultrasound image depnding of the depth type

5.4.3.4 void ProbeCalibrationWidget::getCoordinates ( ) [private, slot]

get the cross point coordinates

5.4.3.5 void ProbeCalibrationWidget::loadRotationsFile ( ) [private, slot]

Load the rotations file.

5.4.3.6 void ProbeCalibrationWidget::loadTranslationsFile ( ) [private, slot]

Load the translation file.

5.4.3.7 void ProbeCalibrationWidget::saveCalibration ( ) [private, slot]

Save the Estimated Parameters in a .txt file.

5.4.3.8 void ProbeCalibrationWidget::setImage (vtkSmartPointer< vtkImageData > *image*)

Set this vtkImageData.

Parameters

in	*a*	smart Pointer of vtkImageData

5.4.3.9 void ProbeCalibrationWidget::setImageStack (std::vector< vtkSmartPointer< vtkImageData > > *imageStack*)

Set this stack of vtkImageData.

Parameters

in	*a*	std Vector of vtkImageData

5.4.3.10 void ProbeCalibrationWidget::setMainWindow (MainWindow ∗ *mainwindow*)

Set the window to display the crop images

5.4.4 Member Data Documentation

5.4.4.1 std::vector< double > ProbeCalibrationWidget::calibrationParameters [private]

the estimate calibration parameters by Calibration.h

5.4.4.2 vnl_matrix< double > ProbeCalibrationWidget::coords [private]

a vnl_matrix to store the selected coordinates on each image

5.4.4.3 vtkSmartPointer< vtkImageData > ProbeCalibrationWidget::image [private]

the vtkImageData to work

5.4.4.4 std::vector< vtkSmartPointer< vtkImageData > > ProbeCalibrationWidget::imageStack [private]

an Array of vtkImageData to work

5.4.4.5 MainWindow∗ ProbeCalibrationWidget::mainWindow [private]

the main window to call it

5.4.4.6 vnl_matrix< double > ProbeCalibrationWidget::rotations [private]

a vnl_matrix to store the rotations of each image given by the tracker

5.4.4.7 float ProbeCalibrationWidget::rotations_2[7][4] [private]

5.4.4.8 vnl_matrix< double > ProbeCalibrationWidget::translations [private]

a vnl_matrix to store the translations of each image given by the tracker

5.4.4.9 bool ProbeCalibrationWidget::workWithStack [private]

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

- ProbeCalibrationWidget.h
- ProbeCalibrationWidget.cpp

5.5 QVTKImageWidget Class Reference

Display VTK images.

`#include <QVTKImageWidget.h>`

Public Member Functions

- QVTKImageWidget (QWidget ∗parent=0)
- virtual ∼QVTKImageWidget ()
- void setAndDisplayImage (QString imageFilename)
- void setAndDisplayImage (vtkSmartPointer< vtkImageData > image)

  Sets and display the given vtkImageData.
- void setAndDisplayMultipleImages (QStringList filenames)

  Set and display multiple images from a given images filenames. Display the image corresponding to the first element on the filenmaes list.
- void setAndDisplayVolumeImages (QStringList ImagesFilenames, QString rotationFilename, QString translatoinFilename, QString calibrationFilename)

  Set and display images from a given images filenames that represent the volume data. Display the image corresponding to the first element on the filenames list.
- void setAndDisplayVolume (QString volumeFilename)

  Set and display volume data.
- void setAndDisplayVolume (vtkSmartPointer< vtkImageData > volumeData)

  Set and display volume data.
- void setAndDisplayMultipleImages (std::vector< vtkSmartPointer< vtkImageData > > imageStack)

  Set and display multiple images from a given vtkImageData Array.
- void displaySelectedImage (int idx)

  display an image stored in this imageStack.
- QString getPixelType ()

  Returns the pixel type in loaded images.
- QString getImageType ()

  Returns the type of image displayed.
- QString getNumOfDimesions ()

  Return the numer of dimensions of the image.
- std::vector< vtkSmartPointer
  < vtkImageData > > getImageStack ()

  return this image stack
- std::vector< vtkSmartPointer
  < vtkImageData > > getVolumeImageStack ()

  return this volume image stack
- std::vector< vnl_matrix< double > > getTransformStack ()

  return this transform stack
- int ∗ getImageSize ()
- int getImageWidth ()
- int getImageHeigth ()
- int getXPicked ()

  Return the mouse x coordinate position when mouse left button is pressed.
- int getYPicked ()

  Return the mouse y coordinate position when mouse left button is pressed.
- vnl_vector< double > getTransformScale ()
- void setXPicked (int xPosition)

  Set the mouse x coordinate position when mouse left button is pressed.
- void setYPicked (int yPosition)

  Set the mouse y coordinate position when mouse left button is pressed.
- void setVolumeData (vtkSmartPointer< vtkImageData > volumeData)

  Set the data of the display volume.
- void setVolumeOpacity (int opacity)

  Set the opacity for the displayed volume.

- void setVolumeOrigin (vnl_vector< double > volumeOrigin)
- vtkSmartPointer< vtkImageViewer2 > getImageViewer ()

  Return this widget image viewer.
- QVTKWidget ∗ getQVTKWidget ()

  Return this qvtkWidget.
- int getImageDisplayedIndex ()

  If an image stack is loaded, then return the index in the image stack of displayed image.

Public Attributes

- bool isImageStackLoaded

  Flag to know if it's displayed an image stack.
- bool isVolumeImageStackLoaded

  Flag to know if it's displayed an volume image stack.

Private Member Functions

- void setImageProperties (bool verbose)
- void displayImage (vtkImageData ∗image)
- void displayVolumeImages (std::vector< vtkSmartPointer< vtkImageData > > volumeImageStack, vnl_matrix< double > volumeDataRotations, vnl_matrix< double > volumeDataTranslations, std::vector< double > volumeDataCalibration)
- void displayVolume (vtkSmartPointer< vtkVolume > volume)
- vnl_matrix< double > computeTransformation (vnl_vector< double > quaternion, vnl_vector< double > translation, std::vector< double > calibration)

Private Attributes

- QVTKWidget ∗ qvtkWidget
- ImageType::Pointer itkImage
- RGBImageType::Pointer rgbItkImage
- vtkSmartPointer< vtkImageData > vtkImage
- vtkSmartPointer< vtkRenderWindow > renwin
- vtkSmartPointer< vtkRenderer > renderer
- vtkSmartPointer< vtkVolume > volume
- vtkSmartPointer< vtkImageData > volumeData
- vtkSmartPointer
  < vtkPiecewiseFunction > volumeScalarOpacity
- vtkSmartPointer
  < vtkVolumeProperty > volumeProperty
- int opacityPoint
- std::vector< vtkSmartPointer
  < vtkImageData > > imageStack

  A vtkImageData Vector for keep the image references when load an image stack.
- std::vector< vtkSmartPointer
  < vtkImageData > > volumeImageStack

  A vtkImageData Vector for keep the volume image references when load an image stack.
- std::vector< vnl_matrix< double > > transformStack

  A vtkImageData Vector for keep the volume image references when load an image stack Transformed.
- std::vector< vtkSmartPointer
  < vtkImageActor > > volumeImageActorStack

  A vtkImageData Vector for keep the volume image actor references when load an image stack.

- std::string pixelType
- vnl_vector< double > scale
- int imageType
- size_t numDimensions
- int imageWidth
- int imageHeight
- int xPosition
- int yPosition
- int xPicked
- int yPicked
- int imageDisplayedIndex
- vnl_matrix< double > volumeDataRotations

    a vnl_matrix to store the rotations of each image given by the tracker
- vnl_matrix< double > volumeDataTranslations

    a vnl_matrix to store the translations of each image given by the tracker
- std::vector< double > volumeDataCalibration
- vtkSmartPointer< vtkImageViewer2 > imageViewer
- vtkSmartPointer
  < vtkCornerAnnotation > cornerAnnotation

    Object for display information in the corners of the vtkImageViewer2.

5.5.1 Detailed Description

Display VTK images.

This class allows the user to display vtkImages on QT. It can display 2D and 3D scenes

5.5.2 Constructor & Destructor Documentation

5.5.2.1 QVTKImageWidget::QVTKImageWidget (QWidget ∗ *parent* = 0)

Constructor for this ImageWidget

5.5.2.2 QVTKImageWidget::∼QVTKImageWidget ( ) [virtual]

Destructor

5.5.3 Member Function Documentation

5.5.3.1 vnl_matrix< double > QVTKImageWidget::computeTransformation (vnl_vector< double > *quaternion*, vnl_vector< double > *translation*, std::vector< double > *calibration*) [private]

Compute the transformation matricez of each image

5.5.3.2 void QVTKImageWidget::displayImage (vtkImageData ∗ *image*) [private]

Display the given vtkImage

5.5.3.3 void **QVTKImageWidget::displaySelectedImage** (int *idx*)

display an image stored in this imageStack.

Parameters

`in`	*the*	index in the stack position of the image

5.5.3.4 void **QVTKImageWidget::displayVolume** (vtkSmartPointer< vtkVolume > *volume*) `[private]`

Display the given volume

5.5.3.5 void **QVTKImageWidget::displayVolumeImages** (std::vector< vtkSmartPointer< vtkImageData > > *volumeImageStack,* vnl_matrix< double > *volumeDataRotations,* vnl_matrix< double > *volumeDataTranslations,* std::vector< double > *volumeDataCalibration*) `[private]`

Display the given volume images

5.5.3.6 int **QVTKImageWidget::getImageDisplayedIndex** ( )

If an image stack is loaded, then return the index in the image stack of displayed image.

5.5.3.7 int **QVTKImageWidget::getImageHeigth** ( )

returns this image heigth

5.5.3.8 int* **QVTKImageWidget::getImageSize** ( )

returns an array with the width and height of the image

5.5.3.9 std::vector< vtkSmartPointer< vtkImageData > > **QVTKImageWidget::getImageStack** ( )

return this image stack

5.5.3.10 QString **QVTKImageWidget::getImageType** ( )

Returns the type of image displayed.

5.5.3.11 vtkSmartPointer< vtkImageViewer2 > **QVTKImageWidget::getImageViewer** ( )

Return this widget image viewer.

Parameters

`out`	*imageViewer*	vtkImageViewer2 target 2D image.

5.5.3.12 int **QVTKImageWidget::getImageWidth** ( )

returns this image width

5.5.3.13 QString QVTKImageWidget::getNumOfDimesions ( )

Return the numer of dimensions of the image.

5.5.3.14 QString QVTKImageWidget::getPixelType ( )

Returns the pixel type in loaded images.

5.5.3.15 QVTKWidget ∗ QVTKImageWidget::getQVTKWidget ( )

Return this qvtkWidget.

Parameters

out	*the*	QVTKWidget

5.5.3.16 vnl_vector< double > QVTKImageWidget::getTransformScale ( )

5.5.3.17 std::vector< vnl_matrix< double > > QVTKImageWidget::getTransformStack ( )

return this transform stack

5.5.3.18 std::vector< vtkSmartPointer< vtkImageData > > QVTKImageWidget::getVolumeImageStack ( )

return this volume image stack

5.5.3.19 int QVTKImageWidget::getXPicked ( )

Return the mouse x coordinate position when mouse left button is pressed.

Parameters

out	*int*	x position

5.5.3.20 int QVTKImageWidget::getYPicked ( )

Return the mouse y coordinate position when mouse left button is pressed.

Parameters

out	*int*	y position

5.5.3.21 void QVTKImageWidget::setAndDisplayImage (QString *imageFilename*)

Sets and display an image from a given image path

5.5.3.22 void QVTKImageWidget::setAndDisplayImage (vtkSmartPointer< vtkImageData > *image*)

Sets and display the given vtkImageData.

Parameters

in	*a*	vtkImageData to set and display

5.5.3.23 void **QVTKImageWidget::setAndDisplayMultipleImages** (QStringList *filenames*)

Set and display multiple images from a given images filenames. Display the image corresponding to the first element on the filenmaes list.

Parameters

in	*a*	QStringList that contain the filename of each image

5.5.3.24 void **QVTKImageWidget::setAndDisplayMultipleImages** (std::vector< vtkSmartPointer< vtkImageData > > *imageStack*)

Set and display multiple images from a given vtkImageData Array.

Parameters

in	*a*	std::vector of vtkImageData

5.5.3.25 void **QVTKImageWidget::setAndDisplayVolume** (QString *volumeFilename*)

Set and display volume data.

Parameters

in	*a*	QString that contain the filename of the volume

5.5.3.26 void **QVTKImageWidget::setAndDisplayVolume** (vtkSmartPointer< vtkImageData > *volumeData*)

Set and display volume data.

Parameters

in	*a*	vtkImageData generated by VolumeReconstruction.h

5.5.3.27 void **QVTKImageWidget::setAndDisplayVolumeImages** (QStringList *ImagesFilenames,* QString *rotationFilename,* QString *translatoinFilename,* QString *calibrationFilename*)

Set and display images from a given images filenames that represent the volume data. Display the image corresponding to the first element on the filenames list.

Parameters

in	*a*	QStringList that contain the filename of each image, a QStringList that contain the filename of the rotation data of each image and a QStringList that contain the filename the translation data of each image.

5.5.3.28 void QVTKImageWidget::setImageProperties (bool *verbose*) [private]

Set the needed image properties (pixelType, imageType, num of dimensions)

5.5.3.29 void QVTKImageWidget::setVolumeData (vtkSmartPointer< vtkImageData > *volumeData*)

Set the data of the display volume.

5.5.3.30 void QVTKImageWidget::setVolumeOpacity (int *opacity*)

Set the opacity for the displayed volume.

5.5.3.31 void QVTKImageWidget::setVolumeOrigin (vnl_vector< double > *volumeOrigin*)

5.5.3.32 void QVTKImageWidget::setXPicked (int *xPosition*)

Set the mouse x coordinate position when mouse left button is pressed.

5.5.3.33 void QVTKImageWidget::setYPicked (int *yPosition*)

Set the mouse y coordinate position when mouse left button is pressed.

5.5.4 Member Data Documentation

5.5.4.1 vtkSmartPointer<vtkCornerAnnotation> QVTKImageWidget::cornerAnnotation [private]

Object for display information in the corners of the vtkImageViewer2.

5.5.4.2 int QVTKImageWidget::imageDisplayedIndex [private]

If image stack is displayed this sets a reference to current image displayed

5.5.4.3 int QVTKImageWidget::imageHeight [private]

Heigth of the image

5.5.4.4 std::vector< vtkSmartPointer<vtkImageData> > QVTKImageWidget::imageStack [private]

A vtkImageData Vector for keep the image references when load an image stack.

5.5.4.5 int QVTKImageWidget::imageType [private]

the number of scalar components in the image 1 => grayscale, 3 => rgb

5.5.4.6 vtkSmartPointer<vtkImageViewer2> QVTKImageWidget::imageViewer [private]

the image viewer for display images

5.5.4.7 int QVTKImageWidget::imageWidth [private]

Width of the image

5.5.4.8 bool QVTKImageWidget::isImageStackLoaded

Flag to know if it's displayed an image stack.

5.5.4.9 bool QVTKImageWidget::isVolumeImageStackLoaded

Flag to know if it's displayed an volume image stack.

5.5.4.10 ImageType::Pointer QVTKImageWidget::itkImage [private]

The grayscale image displayed in this widget

5.5.4.11 size_t QVTKImageWidget::numDimensions [private]

The number of the image dimensions

5.5.4.12 int QVTKImageWidget::opacityPoint [private]

5.5.4.13 std::string QVTKImageWidget::pixelType [private]

The type of the image pixels

5.5.4.14 QVTKWidget∗ QVTKImageWidget::qvtkWidget [private]

The QVTKWidget for display and interact with the images

5.5.4.15 vtkSmartPointer<vtkRenderer> QVTKImageWidget::renderer [private]

The VTK renderer

5.5.4.16 vtkSmartPointer<vtkRenderWindow> QVTKImageWidget::renwin [private]

The VTK render window

5.5.4.17 RGBImageType::Pointer QVTKImageWidget::rgbItkImage [private]

The RGB image displayed for this widget

5.5.4.18 vnl_vector<double> QVTKImageWidget::scale [private]

Estimated scale of the images

5.5.4.19 std::vector< vnl_matrix<double> > QVTKImageWidget::transformStack [private]

A vtkImageData Vector for keep the volume image references when load an image stack Transformed.

5.5.4.20 vtkSmartPointer<vtkVolume> QVTKImageWidget::volume [private]

The displayed volume data

5.5.4.21 vtkSmartPointer<vtkImageData> QVTKImageWidget::volumeData [private]

The displayed volume data

5.5.4.22 std::vector<double> QVTKImageWidget::volumeDataCalibration [private]

The angles and translation estimated

5.5.4.23 vnl_matrix<double> QVTKImageWidget::volumeDataRotations [private]

a vnl_matrix to store the rotations of each image given by the tracker

5.5.4.24 vnl_matrix<double> QVTKImageWidget::volumeDataTranslations [private]

a vnl_matrix to store the translations of each image given by the tracker

5.5.4.25 std::vector< vtkSmartPointer<vtkImageActor> > QVTKImageWidget::volumeImageActorStack [private]

A vtkImageData Vector for keep the volume image actor references when load an image stack.

5.5.4.26 std::vector< vtkSmartPointer<vtkImageData> > QVTKImageWidget::volumeImageStack [private]

A vtkImageData Vector for keep the volume image references when load an image stack.

5.5.4.27 vtkSmartPointer<vtkVolumeProperty> QVTKImageWidget::volumeProperty [private]

The displayed volume properties

5.5.4.28 vtkSmartPointer<vtkPiecewiseFunction> QVTKImageWidget::volumeScalarOpacity [private]

The displayed volume opacity

5.5.4.29 vtkSmartPointer<vtkImageData> QVTKImageWidget::vtkImage [private]

The VTK image to display i this window

5.5.4.30 int QVTKImageWidget::xPicked [private]

The x coordinate of the picked position over the image

5.5.4.31 int QVTKImageWidget::xPosition [private]

current x coordinate of mouse position over the image

5.5.4.32 int QVTKImageWidget::yPicked `[private]`

current y coordinate of picked position over the image

5.5.4.33 int QVTKImageWidget::yPosition `[private]`

current y coordinate of mouse position over the image

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

- QVTKImageWidget.h
- QVTKImageWidget.cpp

5.6 QVTKImageWidgetCommand Class Reference

Interaction with mouse.

`#include <QVTKImageWidgetCommand.h>`

Public Member Functions

- QVTKImageWidgetCommand ()
- ∼QVTKImageWidgetCommand ()
- void SetPicker (vtkSmartPointer< vtkPropPicker > picker)
- void SetAnnotation (vtkSmartPointer< vtkCornerAnnotation > annotation)
- void SetImageWidget (QVTKImageWidget ∗imageWidget)

  Set the 2d image widget related to this 2d event manager.
- virtual void Execute (vtkObject ∗, unsigned long vtkNotUsed(event), void ∗)

Static Public Member Functions

- static QVTKImageWidgetCommand ∗ New ()

Private Attributes

- vtkSmartPointer< vtkPropPicker > Picker
- vtkSmartPointer
  < vtkCornerAnnotation > Annotation
- QVTKImageWidget ∗ ImageWidget

5.6.1 Detailed Description

Interaction with mouse.

The mouse motion callback, to pick the image and recover pixel values

5.6.2 Constructor & Destructor Documentation

5.6.2.1 QVTKImageWidgetCommand::QVTKImageWidgetCommand ( )

5.6.2.2 QVTKImageWidgetCommand::∼QVTKImageWidgetCommand ( )

5.6.3 Member Function Documentation

5.6.3.1 void QVTKImageWidgetCommand::Execute (vtkObject ∗ , unsigned long *vtkNotUsedevent*, void ∗) [virtual]

5.6.3.2 QVTKImageWidgetCommand ∗ QVTKImageWidgetCommand::New ( ) [static]

5.6.3.3 void QVTKImageWidgetCommand::SetAnnotation (vtkSmartPointer< vtkCornerAnnotation > *annotation*)

5.6.3.4 void QVTKImageWidgetCommand::SetImageWidget (QVTKImageWidget ∗ *imageWidget*)

Set the 2d image widget related to this 2d event manager.

Parameters

in	*viewer*	QVTKImageWidget target 2D image

5.6.3.5 void QVTKImageWidgetCommand::SetPicker (vtkSmartPointer< vtkPropPicker > *picker*)

5.6.4 Member Data Documentation

5.6.4.1 vtkSmartPointer<vtkCornerAnnotation> QVTKImageWidgetCommand::Annotation [private]

Pointer to the annotation

5.6.4.2 QVTKImageWidget∗ QVTKImageWidgetCommand::ImageWidget [private]

The widget related to the mouse events

5.6.4.3 vtkSmartPointer<vtkPropPicker> QVTKImageWidgetCommand::Picker [private]

Pointer to the picker

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

- QVTKImageWidgetCommand.h
- QVTKImageWidgetCommand.cpp

5.7 VolumeReconstruction Class Reference

Generate a new volume.

```
#include <VolumeReconstruction.h>
```

Public Member Functions

- void setVolumeSize (vnl_vector< double >)

  Set the size of the volume data.
- void setVolumeOrigin (vnl_vector< double >)

  Set the volume data orgin in the 3D scene.
- void setImageBoundsStack (std::vector< vnl_vector< double > >, std::vector< vnl_vector< double > >, std::vector< vnl_vector< double > >)

  Set the image bounds.

- void setVolumeImageStack (std::vector< vtkSmartPointer< vtkImageData > >)

  Set image data stack to generate the volume.
- void setTransformStack (std::vector< vnl_matrix< double > >)

  Set the transformation for each image used in the reconstruction.
- void setScale (vnl_vector< double >)

  Set the scale of the images.
- vtkSmartPointer< vtkImageData > generateVolume ()

  Returns the new volume data with the voxel based method.

Static Public Member Functions

- static VolumeReconstruction ∗ New ()

  Constructor.

Private Member Functions

- void calcImagePlane ()

  Compute the plane equation for each image.
- double calcMaxDistance ()

  Computes the maximun distance in the volume.
- double calcVoxelValue (std::vector< vnl_vector< double > >, vnl_vector< double >, vnl_vector< double >)

  Computes the voxel value using three lineal interpolation.

Private Attributes

- vnl_vector< double > volumeSize
- vnl_vector< double > volumeOrigin
- std::vector< vnl_vector< double > > imageBoundsXStack
- std::vector< vnl_vector< double > > imageBoundsYStack
- std::vector< vnl_vector< double > > imageBoundsZStack
- std::vector< vtkSmartPointer
  < vtkImageData > > volumeImageStack
- std::vector< vnl_matrix< double > > transformStack
- vnl_vector< double > scale
- std::vector< vtkSmartPointer
  < vtkPlane > > imagePlaneStack
- double maxDistance

5.7.1 Detailed Description

Generate a new volume.

This class generate a new volume data using a voxel based method with the previously loaded data. It requiers the images data, the tracker data and the estimated parameters from a calibration. The method implemented a nearest pixel interpolation.

5.7.2 Member Function Documentation

5.7.2.1 void VolumeReconstruction::calcImagePlane ( ) [private]

Compute the plane equation for each image.

5.7.2.2 double VolumeReconstruction::calcMaxDistance ( ) [private]

Computes the maximun distance in the volume.

5.7.2.3 double VolumeReconstruction::calcVoxelValue (std::vector< vnl_vector< double > > *crossPoints*, vnl_vector< double > *distancePlane*, vnl_vector< double > *distance*) [private]

Computes the voxel value using three lineal interpolation.

5.7.2.4 vtkSmartPointer< vtkImageData > VolumeReconstruction::generateVolume ( )

Returns the new volume data with the voxel based method.

5.7.2.5 static VolumeReconstruction* VolumeReconstruction::New ( ) [inline, static]

Constructor.

5.7.2.6 void VolumeReconstruction::setImageBoundsStack (std::vector< vnl_vector< double > > *imageBoundsXStack*, std::vector< vnl_vector< double > > *imageBoundsYStack*, std::vector< vnl_vector< double > > *imageBoundsZStack*)

Set the image bounds.

5.7.2.7 void VolumeReconstruction::setScale (vnl_vector< double > *scale*)

Set the scale of the images.

5.7.2.8 void VolumeReconstruction::setTransformStack (std::vector< vnl_matrix< double > > *transformStack*)

Set the transformation for each image used in the reconstruction.

5.7.2.9 void VolumeReconstruction::setVolumeImageStack (std::vector< vtkSmartPointer< vtkImageData > > *volumeImageStack*)

Set image data stack to generate the volume.

5.7.2.10 void VolumeReconstruction::setVolumeOrigin (vnl_vector< double > *volumeOrigin*)

Set the volume data orgin in the 3D scene.

5.7.2.11 void VolumeReconstruction::setVolumeSize (vnl_vector< double > *volumeSize*)

Set the size of the volume data.

5.7.3 Member Data Documentation

5.7.3.1 std::vector< vnl_vector<double> > VolumeReconstruction::imageBoundsXStack [private]

Stacks for the image Bounds in x

5.7.3.2 std::vector< vnl_vector<double> > VolumeReconstruction::imageBoundsYStack [private]

Stacks for the image Bounds in Y

5.7.3.3 std::vector< vnl_vector<double> > VolumeReconstruction::imageBoundsZStack [private]

Stacks for the image Bounds in Z

5.7.3.4 std::vector< vtkSmartPointer<vtkPlane> > VolumeReconstruction::imagePlaneStack [private]

The plane equation for each image

5.7.3.5 double VolumeReconstruction::maxDistance [private]

the maximun distance found in the volume

5.7.3.6 vnl_vector<double> VolumeReconstruction::scale [private]

scale of the images

5.7.3.7 std::vector< vnl_matrix<double> > VolumeReconstruction::transformStack [private]

Contains the transformation for each image

5.7.3.8 std::vector< vtkSmartPointer< vtkImageData> > VolumeReconstruction::volumeImageStack [private]

The stack of images data

5.7.3.9 vnl_vector<double> VolumeReconstruction::volumeOrigin [private]

Where the volume data begins in the 3D scene

5.7.3.10 vnl_vector<double> VolumeReconstruction::volumeSize [private]

Size of the volume

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

- VolumeReconstruction.h
- VolumeReconstruction.cpp

5.8 VolumeReconstructionWidget Class Reference

Has the interaction methods for the user to generate a new volue.

```
#include <VolumeReconstructionWidget.h>
```

Public Member Functions

- VolumeReconstructionWidget (QWidget ∗parent=0)
- ∼VolumeReconstructionWidget ()
- void setMainWindow (MainWindow ∗mainwindow)
- void setTransformStack (std::vector< vnl_matrix< double > >)
- void setVolumeImageStack (std::vector< vtkSmartPointer< vtkImageData > >)

Private Slots

- void save ()

  Saves the volume in a .mhd and .raw file.
- void generate ()

  Calls VolumeReconstruction.h to generate the new volume data.

Private Member Functions

- void calcImageCoords ()

  Computes every pixel coord of each image in the 3D space.
- void calcImageBounds ()

  Computes the coords of the images bounds int he 3D space.
- void calcVolumeSize (bool)

  Computes the volume size.
- void setVolumeOpacity ()

  Set the volume opacity.
- void setVolumeColorMap ()

  Set the volume colo transfer function.
- void setDisplayProperties (vtkSmartPointer< vtkImageData >)

  Set the display properties of the volume and the rendering method.
- void displayVolume ()

  Display the volume in the 3D scene.

Private Attributes

- Ui::VolumeReconstructionWidget ∗ ui
- MainWindow ∗ mainWindow
- std::vector< vtkSmartPointer
  < vtkImageData > > volumeImageStack
- std::vector< vnl_matrix< double > > transformStack
- std::vector< vnl_matrix< double > > imageCoordsXStack
- std::vector< vnl_matrix< double > > imageCoordsYStack
- std::vector< vnl_matrix< double > > imageCoordsZStack
- std::vector< vnl_vector< double > > imageBoundsXStack
- std::vector< vnl_vector< double > > imageBoundsYStack
- std::vector< vnl_vector< double > > imageBoundsZStack
- vtkSmartPointer< vtkVolume > volume
- vtkSmartPointer< vtkImageData > volumeData
- vtkSmartPointer
  < vtkVolumeProperty > volumeProperty
- vnl_vector< double > volumeOrigin
- vnl_vector< double > volumeFinal
- vnl_vector< double > volumeSize
- vnl_vector< double > scale

5.8.1 Detailed Description

Has the interaction methods for the user to generate a new volue.

This class allows the user to choose between a voxel based method or a pixel based method to recontruct a volume and set the main volume properties. It allows to change the opacity of the generated volume and to changethe colors. This class also allows to the user to save the volume in a .mhd and .raw files

5.8.2 Constructor & Destructor Documentation

5.8.2.1 VolumeReconstructionWidget::VolumeReconstructionWidget (QWidget ∗ *parent* = 0) [explicit]

5.8.2.2 VolumeReconstructionWidget::∼VolumeReconstructionWidget ( )

5.8.3 Member Function Documentation

5.8.3.1 void VolumeReconstructionWidget::calcImageBounds ( ) [private]

Computes the coords of the images bounds int he 3D space.

5.8.3.2 void VolumeReconstructionWidget::calcImageCoords ( ) [private]

Computes every pixel coord of each image in the 3D space.

5.8.3.3 void VolumeReconstructionWidget::calcVolumeSize (bool *usePixelMethod*) [private]

Computes the volume size.

Parameters

in	*if*	bool is true it computes it with the imagecoords, else it uses the image bounds

5.8.3.4 void VolumeReconstructionWidget::displayVolume ( ) [private]

Display the volume in the 3D scene.

5.8.3.5 void VolumeReconstructionWidget::generate ( ) [private, slot]

Calls VolumeReconstruction.h to generate the new volume data.

5.8.3.6 void VolumeReconstructionWidget::save ( ) [private, slot]

Saves the volume in a .mhd and .raw file.

5.8.3.7 void VolumeReconstructionWidget::setDisplayProperties (vtkSmartPointer< vtkImageData >) [private]

Set the display properties of the volume and the rendering method.

5.8.3.8 void VolumeReconstructionWidget::setMainWindow (MainWindow ∗ *mainwindow*)

Set the window to display the volume

5.8.3.9 void VolumeReconstructionWidget::setTransformStack (std::vector< vnl_matrix< double > > *transformStack*)

Set the transformation stack for the volume image

5.8.3.10 void VolumeReconstructionWidget::setVolumeColorMap ( ) [private]

Set the volume colo transfer function.

5.8.3.11 void VolumeReconstructionWidget::setVolumeImageStack (std::vector< vtkSmartPointer< vtkImageData > > *volumeImageStack*)

Set the image data stack

5.8.3.12 void VolumeReconstructionWidget::setVolumeOpacity ( ) [private]

Set the volume opacity.

5.8.4 Member Data Documentation

5.8.4.1 std::vector< vnl_vector<double> > VolumeReconstructionWidget::imageBoundsXStack [private]

Contains the transformed bounds in x of each image pixel

5.8.4.2 std::vector< vnl_vector<double> > VolumeReconstructionWidget::imageBoundsYStack [private]

Contains the transformed bounds in y of each image pixel

5.8.4.3 std::vector< vnl_vector<double> > VolumeReconstructionWidget::imageBoundsZStack [private]

Contains the transformed bounds in z of each image pixel

5.8.4.4 std::vector< vnl_matrix<double> > VolumeReconstructionWidget::imageCoordsXStack [private]

Contains the transformed coords in x of each image pixel

5.8.4.5 std::vector< vnl_matrix<double> > VolumeReconstructionWidget::imageCoordsYStack [private]

Contains the transformed coords in y of each image pixel

5.8.4.6 std::vector< vnl_matrix<double> > VolumeReconstructionWidget::imageCoordsZStack [private]

Contains the transformed coords in z of each image pixel

5.8.4.7 MainWindow* VolumeReconstructionWidget::mainWindow [private]

the main window to call it

5.8.4.8 vnl_vector<double> VolumeReconstructionWidget::scale [private]

Scale of the images

5.8.4.9 std::vector< vnl_matrix<double> > VolumeReconstructionWidget::transformStack [private]

Contains the transformation of each image

5.8.4.10 Ui::VolumeReconstructionWidget* VolumeReconstructionWidget::ui [private]

5.8.4.11 vtkSmartPointer<vtkVolume> VolumeReconstructionWidget::volume [private]

the volume to display

5.8.4.12 vtkSmartPointer<vtkImageData> VolumeReconstructionWidget::volumeData [private]

Data of the volume

5.8.4.13 vnl_vector<double> VolumeReconstructionWidget::volumeFinal [private]

End of the volume data in the 3D space

5.8.4.14 std::vector< vtkSmartPointer<vtkImageData> > VolumeReconstructionWidget::volumeImageStack [private]

The data of each image

5.8.4.15 vnl_vector<double> VolumeReconstructionWidget::volumeOrigin [private]

Start of the volume data in the 3D space

5.8.4.16 vtkSmartPointer<vtkVolumeProperty> VolumeReconstructionWidget::volumeProperty [private]

Main volume properties

5.8.4.17 vnl_vector<double> VolumeReconstructionWidget::volumeSize [private]

Size of the volume data

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

- VolumeReconstructionWidget.h
- VolumeReconstructionWidget.cpp

Chapter 6

File Documentation

6.1 Calibration.cpp File Reference

```
#include "Calibration.h"
```

6.2 Calibration.h File Reference

```
#include "SinglePointTargetUSCalibrationParametersEstimator.h"
#include "RANSAC.h"
```

Classes

- class Calibration

  Implements LSQRRecepies methods.

6.3 CropImagesWidget.cpp File Reference

```
#include "CropImagesWidget.h"
#include "vtkExtractVOI.h"
#include "vtkBMPWriter.h"
```

6.4 CropImagesWidget.h File Reference

```
#include "ui_CropImagesWidget.h"
#include "mainwindow.h"
#include <QWidget>
#include <vtkSmartPointer.h>
#include <vtkImageData.h>
```

Classes

- class CropImagesWidget

Crop images.

Namespaces

- namespace Ui

6.5 main.cpp File Reference

```
#include <QtGui/QApplication>
#include "mainwindow.h"
```

Functions

- int main (int argc, char ∗argv[])

6.5.1 Function Documentation

6.5.1.1 int main (int *argc,* char ∗ *argv[]*)

6.6 mainwindow.cpp File Reference

```
#include "ui_mainwindow.h"
#include "mainwindow.h"
#include "ProbeCalibrationWidget.h"
#include "VolumeReconstructionWidget.h"
#include "CropImagesWidget.h"
#include <QVBoxLayout>
#include <vtkEventQtSlotConnect.h>
```

6.7 mainwindow.h File Reference

```
#include <QMainWindow>
#include <QtGui>
#include "QVTKImageWidget.h"
#include <vtkEventQtSlotConnect.h>
```

Classes

- class MainWindow

  Main window for user interaction.

Namespaces

- namespace Ui

6.8 ProbeCalibrationWidget.cpp File Reference

```
#include "ProbeCalibrationWidget.h"
#include "Calibration.h"
#include <QErrorMessage>
#include <QString>
#include <QFile>
#include <QTextStream>
#include <vtkExtractVOI.h>
#include <vnl/vnl_quaternion.h>
#include <vnl/vnl_vector_fixed.h>
#include <vnl/algo/vnl_levenberg_marquardt.h>
#include <vnl/vnl_double_2.h>
```

Variables

- bool setCoordsSize = false

6.8.1 Variable Documentation

6.8.1.1 bool **setCoordsSize** = false

6.9 ProbeCalibrationWidget.h File Reference

```
#include "ui_ProbeCalibrationWidget.h"
#include "mainwindow.h"
#include <QWidget>
#include <vtkSmartPointer.h>
#include <vtkImageData.h>
#include <vnl/vnl_matrix.h>
#include <string>
#include <fstream>
#include <stdio.h>
```

Classes

- class ProbeCalibrationWidget

  Obtain data for calibration process.

6.10 QVTKImageWidget.cpp File Reference

```
#include "QVTKImageWidget.h"
```

```
#include "QVTKImageWidgetCommand.h"
#include <QSize.h>
#include <QBoxLayout>
#include <QString>
#include <itkImage.h>
#include <itkImageFileReader.h>
#include <vtkImageReader2.h>
#include <vtkImageReader2Factory.h>
#include <vtkCornerAnnotation.h>
#include <vtkPropPicker.h>
#include <vtkTextProperty.h>
#include <vtkImageActor.h>
#include <vtkInteractorStyleImage.h>
#include <vtkImageFlip.h>
#include <vtkVolumeRayCastMapper.h>
#include <vtkVolumeRayCastCompositeFunction.h>
#include <vtkColorTransferFunction.h>
#include <vtkMath.h>
#include <vtkMetaImageReader.h>
```

6.11 QVTKImageWidget.h File Reference

```
#include <QtGui>
#include <QWidget>
#include <QVTKWidget.h>
#include <itkImage.h>
#include <itkRGBPixel.h>
#include <vtkSmartPointer.h>
#include <vtkImageData.h>
#include <vtkRenderWindow.h>
#include <vtkRenderer.h>
#include <vtkCamera.h>
#include <vtkImageActor.h>
#include <vtkCommand.h>
#include <vtkImageViewer2.h>
#include <vtkCornerAnnotation.h>
#include <vtkVolume.h>
#include <vtkVolumeProperty.h>
#include <vtkPiecewiseFunction.h>
#include <vtkTransform.h>
#include <vnl/vnl_quaternion.h>
#include <vnl/vnl_matrix.h>
#include <vnl/vnl_vector.h>
```

Classes

- class QVTKImageWidget

  Display VTK images.

Typedefs

- typedef itk::RGBPixel
  < unsigned char > RGBPixelType

- typedef itk::Image< unsigned char > ImageType
- typedef itk::Image
  < RGBPixelType, 2 > RGBImageType

6.11.1 Typedef Documentation

6.11.1.1 typedef itk::Image< unsigned char > ImageType

6.11.1.2 typedef itk::Image< RGBPixelType, 2> RGBImageType

6.11.1.3 typedef itk::RGBPixel< unsigned char > RGBPixelType

6.12 QVTKImageWidgetCommand.cpp File Reference

```
#include "QVTKImageWidgetCommand.h"
#include <vtkImageActor.h>
#include <vtkImageData.h>
#include <vtkInteractorStyleImage.h>
#include <vtkRenderWindow.h>
#include <vtkRenderWindowInteractor.h>
#include <vtkVariant.h>
#include <vtkMath.h>
#include <vtkCommand.h>
#include <vtkImageViewer2.h>
```

6.13 QVTKImageWidgetCommand.h File Reference

```
#include "QVTKImageWidget.h"
#include <vtkCommand.h>
#include <vtkPropPicker.h>
#include <vtkCornerAnnotation.h>
#include <vtkSmartPointer.h>
```

Classes

- class QVTKImageWidgetCommand

  Interaction with mouse.

6.14 VolumeReconstruction.cpp File Reference

```
#include "VolumeReconstruction.h"
#include <vtkMath.h>
#include <vtkMetaImageWriter.h>
#include <vnl/vnl_inverse.h>
#include <exception>
#include <time.h>
```

6.15 VolumeReconstruction.h File Reference

```
#include <vtkSmartPointer.h>
#include <vtkImageData.h>
#include <vtkPlane.h>
#include <vnl/vnl_matrix.h>
#include <vnl/vnl_vector.h>
#include <math.h>
#include <vector>
```

Classes

- class VolumeReconstruction

  Generate a new volume.

6.16 VolumeReconstructionWidget.cpp File Reference

```
#include "VolumeReconstructionWidget.h"
#include "ui_VolumeReconstructionWidget.h"
#include "VolumeReconstruction.h"
#include "vtkMetaImageWriter.h"
#include <QString>
```

6.17 VolumeReconstructionWidget.h File Reference

```
#include <QWidget>
#include "mainwindow.h"
#include <vtkSmartPointer.h>
#include <vtkImageData.h>
#include <vtkVolume.h>
#include <vtkVolumeRayCastMapper.h>
#include <vtkVolumeRayCastCompositeFunction.h>
#include <vtkVolumeProperty.h>
#include <vtkColorTransferFunction.h>
#include <vtkPiecewiseFunction.h>
#include <vtkMath.h>
#include <vnl/vnl_matrix.h>
#include <vnl/vnl_vector.h>
```

Classes

- class VolumeReconstructionWidget

  Has the interaction methods for the user to generate a new volue.

Namespaces

- namespace Ui

Index