

Mister Spex
1.0

Generated by Doxygen 1.8.4

Wed Jan 22 2014 11:39:00

Contents

1	Documentation of Mister Spex	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Class Documentation	9
5.1	Buffer< T > Class Template Reference	9
5.1.1	Detailed Description	10
5.1.2	Constructor & Destructor Documentation	10
5.1.2.1	Buffer	10
5.1.3	Member Function Documentation	10
5.1.3.1	add	10
5.1.3.2	clear	10
5.1.3.3	get	10
5.1.3.4	isEmpty	11
5.1.3.5	isFull	11
5.1.3.6	maxSize	11
5.1.3.7	size	11
5.1.4	Member Data Documentation	11
5.1.4.1	bufferSize	11
5.1.4.2	clearBuffer_add	11
5.1.4.3	clearBuffer_get	11
5.1.4.4	freeSlots	12
5.1.4.5	queue	12
5.1.4.6	queueProtect	12
5.1.4.7	usedSlots	12
5.2	CameraConnectDialog Class Reference	12

5.2.1	Detailed Description	13
5.2.2	Constructor & Destructor Documentation	13
5.2.2.1	CameraConnectDialog	13
5.2.3	Member Function Documentation	13
5.2.3.1	getMode	13
5.2.4	Member Data Documentation	13
5.2.4.1	ui	13
5.3	FaceDetection Class Reference	13
5.3.1	Detailed Description	14
5.3.2	Member Function Documentation	14
5.3.2.1	cvCreateOverlay	14
5.3.2.2	cvCreateOverlayGlasses	14
5.3.2.3	detectObjects	14
5.4	ImageHandler Class Reference	15
5.4.1	Detailed Description	15
5.4.2	Member Function Documentation	15
5.4.2.1	applyChangesToImageProc	15
5.4.2.2	loadImageFromFile	16
5.4.2.3	saveImageToFile	16
5.4.3	Member Data Documentation	16
5.4.3.1	faceDetection	16
5.4.3.2	image	16
5.4.3.3	orig_image	17
5.5	ImageProcessingFlags Struct Reference	17
5.5.1	Detailed Description	17
5.5.2	Member Data Documentation	17
5.5.2.1	showDetectionOn	17
5.5.2.2	showOverlaysOn	17
5.6	ImageProcessingSettings Struct Reference	17
5.6.1	Detailed Description	18
5.6.2	Member Data Documentation	18
5.6.2.1	overlayParam1	18
5.6.2.2	overlayParam2	18
5.6.2.3	overlayParam3	18
5.7	ImageProcessingSettingsDialog Class Reference	18
5.7.1	Detailed Description	19
5.7.2	Constructor & Destructor Documentation	19
5.7.2.1	ImageProcessingSettingsDialog	19
5.7.3	Member Function Documentation	20
5.7.3.1	getImgProcSettingsForImg	20

5.7.3.2	ImgProcFlagsForImg	20
5.7.3.3	newImageProcessingSettings	20
5.7.3.4	resetOverlays	20
5.7.3.5	updateStoredSettingsFromDialog	20
5.7.4	Member Data Documentation	20
5.7.4.1	imageProcessingSettings	20
5.7.4.2	ui	21
5.8	MainWindow Class Reference	21
5.8.1	Detailed Description	23
5.8.2	Constructor & Destructor Documentation	23
5.8.2.1	MainWindow	23
5.8.3	Member Function Documentation	24
5.8.3.1	changeDetectionState	24
5.8.3.2	newImageProcessingFlags	25
5.8.3.3	setLiveViewSaveImgFlag	25
5.8.3.4	setRecordFlag	25
5.8.3.5	setROI	25
5.8.3.6	setSaveParams	25
5.8.3.7	showImage	26
5.8.3.8	startPlayback	26
5.8.3.9	updateFrame	26
5.8.4	Member Data Documentation	26
5.8.4.1	detection	26
5.8.4.2	deviceNumber	26
5.8.4.3	file	26
5.8.4.4	imageDetection	26
5.8.4.5	imageHandler	27
5.8.4.6	imageProcessingFlags	27
5.8.4.7	imageProcessingSettings	27
5.8.4.8	imageProcessingSettingsDialog	27
5.8.4.9	liveViewActive	27
5.8.4.10	mode	27
5.8.4.11	playbackActive	27
5.8.4.12	playbackThread	27
5.8.4.13	processingThread	27
5.8.4.14	rec_height	28
5.8.4.15	rec_width	28
5.8.4.16	recording	28
5.8.4.17	recordThread	28
5.8.4.18	saveDirectory	28

5.8.4.19	ui	28
5.8.4.20	videoImageBuffer	28
5.8.4.21	videoThread	28
5.9	PlaybackThread Class Reference	29
5.9.1	Detailed Description	30
5.9.2	Constructor & Destructor Documentation	30
5.9.2.1	PlaybackThread	30
5.9.3	Member Function Documentation	30
5.9.3.1	nextFrame	30
5.9.3.2	run	30
5.9.3.3	updateDetState	31
5.9.3.4	updateImageProcessingFlags	31
5.9.3.5	updateImageProcessingSettings	31
5.9.4	Member Data Documentation	31
5.9.4.1	cap	31
5.9.4.2	counter	31
5.9.4.3	detectionState	31
5.9.4.4	doStop	31
5.9.4.5	doStopMutex	31
5.9.4.6	faceDetection	32
5.9.4.7	fn	32
5.9.4.8	fps	32
5.9.4.9	frame	32
5.9.4.10	frames	32
5.9.4.11	grabbedFrame	32
5.9.4.12	imgProcFlags	32
5.9.4.13	imgProcSettings	32
5.9.4.14	processingMutex	32
5.10	ProcessingThread Class Reference	33
5.10.1	Detailed Description	35
5.10.2	Constructor & Destructor Documentation	35
5.10.2.1	ProcessingThread	35
5.10.3	Member Function Documentation	35
5.10.3.1	getCurrentROI	35
5.10.3.2	newFrame	35
5.10.3.3	recFrame	35
5.10.3.4	run	35
5.10.3.5	saveCurrentImage	35
5.10.3.6	setFrameProcessing	36
5.10.3.7	setRecFlag	36

5.10.3.8	setROI	36
5.10.3.9	setSaveImgFlag	36
5.10.3.10	setSaveParams	36
5.10.3.11	updateImageProcessingFlags	36
5.10.3.12	updateImageProcessingSettings	37
5.10.4	Member Data Documentation	37
5.10.4.1	currentFrame	37
5.10.4.2	currentFrameGrayscale	37
5.10.4.3	currentROI	37
5.10.4.4	deviceNumber	37
5.10.4.5	doStop	37
5.10.4.6	doStopMutex	37
5.10.4.7	enableFrameProcessing	37
5.10.4.8	faceDetection	38
5.10.4.9	frame	38
5.10.4.10	framePoint	38
5.10.4.11	frameSize	38
5.10.4.12	imgProcFlags	38
5.10.4.13	imgProcSettings	38
5.10.4.14	processingMutex	38
5.10.4.15	recDir	38
5.10.4.16	recFlag	38
5.10.4.17	recFn	39
5.10.4.18	savImage	39
5.10.4.19	videoImageBuffer	39
5.11	RecordVideoThread Class Reference	39
5.11.1	Detailed Description	40
5.11.2	Constructor & Destructor Documentation	40
5.11.2.1	RecordVideoThread	40
5.11.3	Member Function Documentation	40
5.11.3.1	addRecFrame	40
5.11.3.2	createVideoWriter	40
5.11.3.3	run	41
5.11.4	Member Data Documentation	41
5.11.4.1	doStop	41
5.11.4.2	doStopMutex	41
5.11.4.3	fn	41
5.11.4.4	rec_fps	41
5.11.4.5	rec_height	41
5.11.4.6	rec_width	41

5.11.4.7	recordingMutex	41
5.11.4.8	vid_writer	41
5.12	VidolImageBuffer Class Reference	42
5.12.1	Detailed Description	42
5.12.2	Member Function Documentation	42
5.12.2.1	add	42
5.12.2.2	containsImageBufferForDeviceNumber	43
5.12.2.3	getByDeviceNumber	43
5.12.2.4	removeByDeviceNumber	43
5.12.3	Member Data Documentation	43
5.12.3.1	imageBufferMap	43
5.13	VideoThread Class Reference	43
5.13.1	Detailed Description	44
5.13.2	Constructor & Destructor Documentation	45
5.13.2.1	VideoThread	45
5.13.3	Member Function Documentation	45
5.13.3.1	connectToCamera	45
5.13.3.2	disconnectCamera	45
5.13.3.3	getInputSourceHeight	45
5.13.3.4	getInputSourceWidth	45
5.13.3.5	isCameraConnected	45
5.13.3.6	run	46
5.13.4	Member Data Documentation	46
5.13.4.1	cap	46
5.13.4.2	deviceNumber	46
5.13.4.3	doStop	46
5.13.4.4	doStopMutex	46
5.13.4.5	dropFrameIfBufferFull	46
5.13.4.6	grabbedFrame	46
5.13.4.7	videolImageBuffer	46
6	File Documentation	47
6.1	Config.h File Reference	47
6.1.1	Detailed Description	47
6.1.2	Macro Definition Documentation	47
6.1.2.1	DEFAULT_IMAGE_BUFFER_SIZE	47
6.1.2.2	OVERLAY_PARAM_1	47
6.1.2.3	OVERLAY_PARAM_2	47
6.1.2.4	OVERLAY_PARAM_3	47
6.2	ImageConversion.h File Reference	48

6.2.1	Detailed Description	48
6.2.2	Function Documentation	48
6.2.2.1	cvMatToQImage	48
6.2.2.2	cvMatToQPixmap	48
6.2.2.3	QImageToCvMat	49
6.2.2.4	QPixmapToCvMat	50
6.3	Structures.h File Reference	50
6.3.1	Detailed Description	50

Index**51**

Chapter 1

Documentation of Mister Spex

This is the documentation of the software 'Mister Spex'. It was developed for the university course 'Praktische Aspekte der Informatik' in WS 13/14.

The proposal was declared as following:

+++++ GERMAN ++++++

Student Sven Frank, 4442***

Projektname/working title: MisterSpecs

Kurze Zusammenfassung

Ein kleines Programm zur Erkennung von bestimmten Körperpartien/Gegenständen aus einem Video oder einer Live-Aufnahme. Anschließend können verschiedene Elemente auf die getrackten Objekte aufgesetzt werden. (Beispiel: digitales Aufsetzen einer Brille [richtige Position im Gesicht])

Detaillierte Beschreibung/Auflistung der Funktionalität

Materialinput:

- importieren von vorheraufgenommenen Videos (Standard: mp4-Format, h.264 Codec)
- Echtzeit-Videos über Webcam

Analysefunktionen:

- Gesichtserkennung bzw. Erkennung von verschiedenen Gesichtspartien, wie z.B. Nase

GUI

- Anzeige des Videos/Livebildes
- Auswahl für verschiedene Overlays

Optional:

- Tracking von weiteren Objekten abseits von Gesichtern
- Aufzeichnung von Webcamaufnahmen (inkl. Overlays) und Speichern in mp4-Format
- Unterstützung verschiedener Codes (Import/Export)
- Erstellen von eigenen Overlays
- passende Skalierung von Overlays
- passende GUI-Erweiterung für einzelne optionale Funktionen

Welche externen Bibliotheken/Tools werden benutzt und wofür?

OpenCV: Bildanalyse und -verarbeitung Bibliothek zur Endcodierung von Videodateien (Auswertung was sich am besten eignet)

Welche besonderen Anforderungen ergeben sich?

(z.B. komplexe Berechnungen in Echtzeit, effiziente Speicherverwaltung/Verarbeitung großer Datenmengen)

- verschiedene Funktionen
- Echtzeit-Tracking von Objekten
- Unterstützung von verschiedenen Dateitypen
- Verarbeitung der Datenmengen beim Aufzeichnen von Videos

+++++

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Buffer< T >	9
FaceDetection	13
ImageHandler	15
ImageProcessingFlags	17
ImageProcessingSettings	17
QDialog	
CameraConnectDialog	12
ImageProcessingSettingsDialog	18
QMainWindow	
MainWindow	21
QThread	
PlaybackThread	29
ProcessingThread	33
RecordVideoThread	39
VideoThread	43
VideolImageBuffer	42

Chapter 3

Class Index

3.1 Class List

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

Buffer< T >	
Buffer template class	9
CameraConnectDialog	
Camera connect dialog class	12
FaceDetection	
FaceDetection class	13
ImageHandler	
Image handler class	15
ImageProcessingFlags	
Structure which contains information about the flags. Determine what process shall be done	17
ImageProcessingSettings	
Structure which contains information about the overlays apply on the image	17
ImageProcessingSettingsDialog	
Image processing dialog class	18
MainWindow	
MainWindow class creates the main application window and builds the basis	21
PlaybackThread	
Playback class which creates a thread to play a video	29
ProcessingThread	
Processing class which does image detection in webcam mode	33
RecordVideoThread	
Record Video class which records the processed images from the webcam to a video file	39
VideoImageBuffer	
Video buffer class	42
VideoThread	
Video class which captures an image from a webcam or specific device	43

Chapter 4

File Index

4.1 File List

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

Buffer.h	??
CameraConnectDialog.h	??
Config.h	
Structure file. It contains the general stored settings which are loaded at the beginning	47
FaceDetection.h	??
ImageConversion.h	
Image conversion functions. Four function to convert images between the following three types: OpenCV Mat, QImage, QPixmap	48
ImageHandler.h	??
ImageProcessingSettingsDialog.h	??
MainWindow.h	??
PlaybackThread.h	??
ProcessingThread.h	??
RecordVideoThread.h	??
Structures.h	
Structures of the program. This file contains to structures to share settings and flags programm wide	50
VideoImageBuffer.h	??
VideoThread.h	??

Chapter 5

Class Documentation

5.1 Buffer< T > Class Template Reference

Buffer template class.

```
#include <Buffer.h>
```

Public Member Functions

- Buffer (int size)
Template function to initialize a new buffer of type T with a specific size. Compareable to a constructor.
- void add (const T &data, bool dropIfFull=false)
Template function add data of type T to the buffer. Data can be dropped if buffer is full.
- T get ()
Template function get data of type T from the buffer.
- int size ()
Template function to get actual buffer size.
- int maxSize ()
Template function to get max buffer size.
- bool clear ()
Template function to clear the actual buffer.
- bool isFull ()
Template function check if buffer is full.
- bool isEmpty ()
Template function check if buffer is empty.

Private Attributes

- QMutex queueProtect
QMutex.
- QQueue< T > queue
QQueue.
- QSemaphore * freeSlots
QSemaphore - free Slots.
- QSemaphore * usedSlots
QSemaphore - used Slots.
- QSemaphore * clearBuffer_add

- QSemaphore* - clear add *Buffer*.
- *QSemaphore* * *clearBuffer_get*
QSemaphore - clear get *Buffer*.
- int *bufferSize*
Buffer size.

5.1.1 Detailed Description

template<class T>class Buffer< T >

Buffer template class.

The buffer template class is the basis to save image temporarily to display them later.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 template<class T > Buffer< T >::Buffer (int size)

Template function to initialize a new buffer of type T with a specific size. Compareable to a constructor.

Parameters

<i>size</i>	an integer argument.
-------------	----------------------

5.1.3 Member Function Documentation

5.1.3.1 template<class T > void Buffer< T >::add (const T & data, bool droplfFull = false)

Template function add data of type T to the buffer. Data can be dropped if buffer is full.

Parameters

<i>data</i>	an argument of the specific type T.
<i>droplfFull</i>	an boolean argument to set if an image should be dropped when buffer is full.

5.1.3.2 template<class T > bool Buffer< T >::clear ()

Template function to clear the actual buffer.

Returns

True if buffer was released. In any other case the function return false.

5.1.3.3 template<class T > T Buffer< T >::get ()

Template function get data of type T from the buffer.

Returns

Data with type T

5.1.3.4 `template<class T> bool Buffer< T >::isEmpty ()`

Template function check if buffer is empty.

Returns

True if buffer is empty, else false.

5.1.3.5 `template<class T> bool Buffer< T >::isFull ()`

Template function check if buffer is full.

Returns

True if buffer is full, else false.

5.1.3.6 `template<class T> int Buffer< T >::maxSize ()`

Template function to get max buffer size.

Returns

Size of the buffer as an integer value.

5.1.3.7 `template<class T> int Buffer< T >::size ()`

Template function to get actual buffer size.

Returns

Actual size of the buffer as an integer value.

5.1.4 Member Data Documentation

5.1.4.1 `template<class T> int Buffer< T >::bufferSize [private]`

Buffer size.

Contains the size of the buffer (Integer value).

5.1.4.2 `template<class T> QSemaphore* Buffer< T >::clearBuffer_add [private]`

QSemaphore - clear add Buffer.

Temporarily buffer to add the image to the queue.

5.1.4.3 `template<class T> QSemaphore* Buffer< T >::clearBuffer_get [private]`

QSemaphore - clear get Buffer.

Temporarily buffer to get the image from the queue.

5.1.4.4 `template<class T> QSemaphore* Buffer< T >::freeSlots` [private]

QSemaphore - free Slots.

Checks for for free slots to save the image.

5.1.4.5 `template<class T> QQueue<T> Buffer< T >::queue` [private]

QQueue.

Place where the actual image will be saved before displaying.

5.1.4.6 `template<class T> QMutex Buffer< T >::queueProtect` [private]

QMutex.

Mutex to protect the image queue while adding the latest image.

5.1.4.7 `template<class T> QSemaphore* Buffer< T >::usedSlots` [private]

QSemaphore - used Slots.

Returns which slots are actually in use.

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

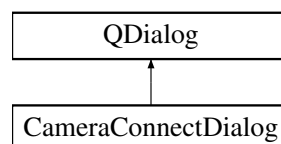
- Buffer.h

5.2 CameraConnectDialog Class Reference

Camera connect dialog class.

```
#include <CameraConnectDialog.h>
```

Inheritance diagram for CameraConnectDialog:



Public Member Functions

- [CameraConnectDialog](#) (QWidget *parent=0)
Constructor of the dialog to set up the UI from the Qt file.
- [~CameraConnectDialog](#) ()
Destructor of the dialog to delete the created UI.
- int [getMode](#) ()
Function which return the user selected mode.

Private Attributes

- Ui::CameraConnectDialog * [ui](#)
CameraConnectDialog UI pointer.

5.2.1 Detailed Description

Camera connect dialog class.

Dialog class to select the mode in which the software should run.

5.2.2 Constructor & Destructor Documentation

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

Constructor of the dialog to set up the UI from the Qt file.

Parameters

<i>parent</i>	an QWidget argument. If no parameter is give parent = 0.
---------------	--

5.2.3 Member Function Documentation

5.2.3.1 int CameraConnectDialog::getMode ()

Function which return the user selected mode.

Returns

An integer value which identifies the selected mode.

5.2.4 Member Data Documentation

5.2.4.1 Ui::CameraConnectDialog* CameraConnectDialog::ui [private]

[CameraConnectDialog](#) UI pointer.

Pointer the the user interface file of the dialog.

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

- CameraConnectDialog.h
- CameraConnectDialog.cpp

5.3 FaceDetection Class Reference

[FaceDetection](#) class.

```
#include <FaceDetection.h>
```

Public Member Functions

- [FaceDetection](#) ()
Class constructor which creates a new handler.
- cv::Mat [detectObjects](#) (cv::Mat &, [ImageProcessingSettings](#), [ImageProcessingFlags](#))
Function to detect objects in a given image. In this case face objects.
- void [cvCreateOverlay](#) (QString, const cv::Mat &, cv::Mat &, cv::Point2i, int)
Function to draw a specific overlay on the given image.
- void [cvCreateOverlayGlasses](#) (QString, const cv::Mat &, cv::Mat &, cv::Point2i, int)
Special draw function for the glasses overlay because the special anker point behaviour.

Private Attributes

- double **tem_crofactor**
- cv::Mat **temp_glass_overlay**

5.3.1 Detailed Description

[FaceDetection](#) class.

Class has functions for object detection and overlay composing.

5.3.2 Member Function Documentation

5.3.2.1 void FaceDetection::cvCreateOverlay (QString *fn*, const cv::Mat & *src*, cv::Mat & *output*, cv::Point2i *location*, int *scale*)

Function to draw a specific overlay on the given image.

Parameters

<i>fn</i>	an QString name of the overlay file to load.
<i>src</i>	an Mat image (OpenCV type) file.
<i>output</i>	an Mat image (OpenCV type) file which also contains the overlay.
<i>location</i>	an Point2i (OpenCV type for single point with to integers coordinates) which is the anker point of the overlay.
<i>scale</i>	an integer value for scaling the overlay according to the image size.

5.3.2.2 void FaceDetection::cvCreateOverlayGlasses (QString *fn*, const cv::Mat & *src*, cv::Mat & *output*, cv::Point2i *location*, int *scale*)

Special draw function for the glasses overlay because the special anker point behaviour.

Parameters

<i>fn</i>	an QString name of the overlay file to load.
<i>src</i>	an Mat image (OpenCV type) file.
<i>output</i>	an Mat image (OpenCV type) file which also contains the overlay.
<i>location</i>	an Point2i (OpenCV type for single point with to integers coordinates) which is the anker point of the overlay.
<i>scale</i>	an integer value for scaling the overlay according to the image size.

5.3.2.3 cv::Mat FaceDetection::detectObjects (cv::Mat & *inMat*, ImageProcessingSettings *settings*, ImageProcessingFlags *drawFlags*)

Function to detect objects in a given image. In this case face objects.

Parameters

<i>inMat</i>	an Mat image (OpenCV type) file.
<i>settings</i>	an ImageProcessingSettings struct which contains the settings for the area which should be detected

<i>drawFlags</i>	an ImageProcessingFlags struct which contains bool values about overlay showing and detection drawing
------------------	---

Returns

Image in Mat type with in image drawn detection and/or overlays

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

- FaceDetection.h
- FaceDetection.cpp

5.4 ImageHandler Class Reference

Image handler class.

```
#include <ImageHandler.h>
```

Public Member Functions

- [ImageHandler](#) ()
Class constructor which creates a new handler.
- [~ImageHandler](#) ()
Class destructor.
- QPixmap [loadImageFromFile](#) (QString, struct [ImageProcessingSettings](#), struct [ImageProcessingFlags](#), bool)
Function to load a single image from a file.
- QPixmap [applyChangesToImageProc](#) (struct [ImageProcessingSettings](#), struct [ImageProcessingFlags](#))
Function to apply changes on the selected image.
- bool [saveImageToFile](#) (QString, QString)
Function to save the display image to file.

Private Attributes

- [FaceDetection](#) * [faceDetection](#)
FaceDetection.
- Mat [image](#)
Mat (OpenCV image format).
- Mat [orig_image](#)
Mat (OpenCV image format).

5.4.1 Detailed Description

Image handler class.

Class to handle static images for the software.

5.4.2 Member Function Documentation

5.4.2.1 QPixmap ImageHandler::applyChangesToImageProc (struct ImageProcessingSettings *imgProcSettings*, struct ImageProcessingFlags *drawFlags*)

Function to apply changes on the selected image.

Parameters

<i>imgProcSettings</i>	an ImageProcessingSettings struct which contains the settings for the area which should be detected
<i>drawFlags</i>	an ImageProcessingFlags struct which contains bool values about overlay showing and detection drawing

Returns

The function returns the selected image converted to QPixmap format with applied detection

5.4.2.2 QPixmap ImageHandler::loadImageFromFile (QString *fn*, struct [ImageProcessingSettings](#) *imgProcSettings*, struct [ImageProcessingFlags](#) *drawFlags*, bool *imageDetection*)

Function to load a single image from a file.

Parameters

<i>fn</i>	a QString argument which contains the name of the file to load.
<i>imgProcSettings</i>	an ImageProcessingSettings struct which contains the settings for the area which should be detected
<i>drawFlags</i>	an ImageProcessingFlags struct which contains bool values about overlay showing and detection drawing
<i>imageDetection</i>	a boolean value to say if image detection should be done.

Returns

The function returns the selected image converted to QPixmap format with applied detection

5.4.2.3 bool ImageHandler::saveImageToFile (QString *fn*, QString *dir*)

Function to save the display image to file.

Parameters

<i>fn</i>	a QString argument which contains the name to save the image.
<i>dir</i>	a QString argument which contains the directory name where the file should be saved.

Returns

Boolean value which says if the process was successful or not.

5.4.3 Member Data Documentation

5.4.3.1 FaceDetection* ImageHandler::faceDetection [private]

[FaceDetection](#).

Pointer to instance of [FaceDetection](#) class.

5.4.3.2 Mat ImageHandler::image [private]

Mat (OpenCV image format).

Image which will be display in the main window after conversion.

5.4.3.3 Mat ImageHandler::orig_image [private]

Mat (OpenCV image format).

Original image.

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

- ImageHandler.h
- ImageHandler.cpp

5.5 ImageProcessingFlags Struct Reference

Structure which contains information about the flags. Determine what process shall be done.

```
#include <Structures.h>
```

Public Attributes

- bool [showDetectionOn](#)
Bool.
- bool [showOverlaysOn](#)
Bool.

5.5.1 Detailed Description

Structure which contains information about the flags. Determine what process shall be done.

Structure with two boolean values.

5.5.2 Member Data Documentation

5.5.2.1 bool ImageProcessingFlags::showDetectionOn

Bool.

Boolean value if detection shall be drawn.

5.5.2.2 bool ImageProcessingFlags::showOverlaysOn

Bool.

Boolean value if overlays shall be drawn.

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

- [Structures.h](#)

5.6 ImageProcessingSettings Struct Reference

Structure which contains information about the overlays apply on the image.

```
#include <Structures.h>
```

Public Attributes

- bool [overlayParam1](#)
Bool.
- bool [overlayParam2](#)
Bool.
- bool [overlayParam3](#)
Bool.

5.6.1 Detailed Description

Structure which contains information about the overlays apply on the image.
Structure with three boolean values.

5.6.2 Member Data Documentation

5.6.2.1 bool ImageProcessingSettings::overlayParam1

Bool.

Boolean value if glasses shall be displayed.

5.6.2.2 bool ImageProcessingSettings::overlayParam2

Bool.

Boolean value if funny nose shall be displayed.

5.6.2.3 bool ImageProcessingSettings::overlayParam3

Bool.

Boolean value if red lips shall be displayed.

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

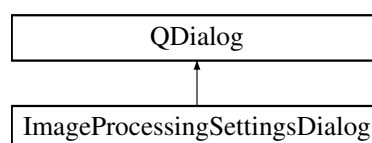
- [Structures.h](#)

5.7 ImageProcessingSettingsDialog Class Reference

Image processing dialog class.

```
#include <ImageProcessingSettingsDialog.h>
```

Inheritance diagram for ImageProcessingSettingsDialog:



Public Slots

- void [updateStoredSettingsFromDialog](#) ()
Public slot to updated the stored settings from the dialog ones.

Signals

- void [newImageProcessingSettings](#) (struct [ImageProcessingSettings](#) p_settings)
Signal: New [ImageProcessingSettings](#) are set.

Public Member Functions

- [ImageProcessingSettingsDialog](#) (QWidget *parent=0)
Constructor of the dialog to set up the UI from the Qt file.
- [~ImageProcessingSettingsDialog](#) ()
Destructor of the dialog to delete the created UI.
- void [updateDialogSettingsFromStored](#) ()
Function which updates the dialog settings from the stored ones.
- [ImageProcessingSettings](#) [getImgProcSettingsForImg](#) ()
Function to get actual [ImageProcessingSettings](#) struct.
- [ImageProcessingFlags](#) [ImgProcFlagsForImg](#) ()
Function to get actual [ImageProcessingFlags](#) struct.

Private Slots

- void [resetOverlays](#) ()
Private slot to reset the display settings.

Private Attributes

- Ui::ImageProcessingSettingsDialog * [ui](#)
[ImageProcessingSettingsDialog](#) UI pointer.
- [ImageProcessingSettings](#) [imageProcessingSettings](#)
[ImageProcessingSettings](#) struct.

5.7.1 Detailed Description

Image processing dialog class.

Dialog to set specfic parameters which will be applied on the displayed image.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 [ImageProcessingSettingsDialog::ImageProcessingSettingsDialog](#) ([QWidget](#) * *parent* = 0) [explicit]

Constructor of the dialog to set up the UI from the Qt file.

Parameters

<i>parent</i>	an QWidget argument. If no parameter is give parent = 0.
---------------	--

5.7.3 Member Function Documentation

5.7.3.1 ImageProcessingSettings ImageProcessingSettingsDialog::getImgProcSettingsForImg ()

Function to get actual [ImageProcessingSettings](#) struct.

Returns

[ImageProcessingSettings](#) struct which contains the acutal settings.

5.7.3.2 ImageProcessingFlags ImageProcessingSettingsDialog::ImgProcFlagsForImg ()

Function to get actual [ImageProcessingFlags](#) struct.

Returns

[ImageProcessingFlags](#) struct which contains the acutal flags.

5.7.3.3 void ImageProcessingSettingsDialog::newImageProcessingSettings (struct ImageProcessingSettings p_settings) [signal]

Signal: New [ImageProcessingSettings](#) are set.

Parameters

<i>p_settings</i>	ImageProcessingSettings struct as argument.
-------------------	---

5.7.3.4 void ImageProcessingSettingsDialog::resetOverlays () [private],[slot]

Private slot to reset the display settings.

Private slot to call function to reset settings. Value from [Config.h](#) are used.

5.7.3.5 void ImageProcessingSettingsDialog::updateStoredSettingsFromDialog () [slot]

Public slot to update the stored settings from the dialog ones.

Public slot to connect with a specific action in program that should call this function

5.7.4 Member Data Documentation

5.7.4.1 ImageProcessingSettings ImageProcessingSettingsDialog::imageProcessingSettings [private]

[ImageProcessingSettings](#) struct.

Structure of several settings.

5.7.4.2 Ui::ImageProcessingSettingsDialog* ImageProcessingSettingsDialog::ui [private]

[ImageProcessingSettingsDialog](#) UI pointer.

Pointer the the user interface file of the dialog.

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

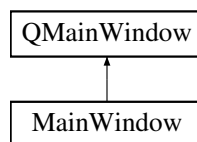
- [ImageProcessingSettingsDialog.h](#)
- [ImageProcessingSettingsDialog.cpp](#)

5.8 MainWindow Class Reference

[MainWindow](#) class creates the main application window and builds the basis.

```
#include <MainWindow.h>
```

Inheritance diagram for MainWindow:



Public Slots

- void [selectMode](#) ()
Public slot to select mode.
- void [connectToCamera](#) ()
Public slot to connect to camera.
- void [disconnectCamera](#) ()
Public slot to disconnect camera.
- void [showAboutDialog](#) ()
Public slot to show about dialog.
- void [setDetection](#) (bool)
Public slot to set detection flag.
- void [setImageProcessingSettings](#) ()
Public slot to set [ImageProcessingSettings](#).
- void [setRecDir](#) ()
Public slot to set recording directory.
- void [takeSnapshot](#) ()
Public slot to take a snapshot of the actual displayed image.
- void [recordVideo](#) ()
Public slot to record a video in LiveView mode.

Signals

- void [newImageProcessingFlags](#) (struct [ImageProcessingFlags](#) imageProcessingFlags)
Signal: Send new [ImageProcessingFlags](#).
- void [changeDetectionState](#) (bool detection)
Signal: Change detection state flag.
- void [setROI](#) (QRect roi)

Signal: Set Region of interest.

- void [setSaveParams](#) (QString dir, QString fn)

Signal: Set save parameter.

- void [setLiveViewSaveImgFlag](#) (bool sFlag)

Signal: Set LiveView snapshot flag.

- void [setRecordFlag](#) (bool rFlag)

Signal: Set record flag.

Public Member Functions

- [MainWindow](#) (QWidget *parent=0)

Constructor of the main window to set up the UI from the Qt file.

- [~MainWindow](#) ()

Destructor of the dialog to delete the created UI.

Private Slots

- void [updateFrame](#) (const QPixmap &frame)

Private slot to update the shown image in [MainWindow](#).

Private Member Functions

- void [showImage](#) (QString)

Function to show a static image from an image file.

- void [applyChangesOnSingleImage](#) ()

Function to apply settings and flags on a loaded image.

- void [stopVideoThread](#) ()

Function to stop the [VideoThread](#).

- void [stopProcessingThread](#) ()

Function to stop the [ProcessingThread](#).

- void [startPlayback](#) (QString)

Function to start displaying a video from a file.

- void [stopPlayback](#) ()

Function to stop the Playback.

- void [stopPlaybackThread](#) ()

Function to stop the [PlaybackThread](#).

Private Attributes

- Ui::MainWindow * [ui](#)

[MainWindow](#) UI pointer.

- [ImageProcessingSettingsDialog](#) * [imageProcessingSettingsDialog](#)

[ImageProcessingSettingsDialog](#) pointer.

- [VideoThread](#) * [videoThread](#)

[VideoThread](#) pointer.

- [VideoImageBuffer](#) * [videoImageBuffer](#)

[VideoImageBuffer](#) pointer.

- [ProcessingThread](#) * [processingThread](#)

[ProcessingThread](#) pointer.

- [PlaybackThread](#) * [playbackThread](#)

- *PlaybackThread* pointer.
- `RecordVideoThread * recordThread`
RecordVideoThread pointer.
- `ImageHandler * imageHandler`
ImageHandler pointer.
- `ImageProcessingFlags imageProcessingFlags`
ImageProcessingFlags struct.
- `ImageProcessingSettings imageProcessingSettings`
ImageProcessingSettings struct.
- `bool imageDetection`
Bool.
- `bool detection`
Bool.
- `QString saveDirectory`
QString.
- `bool recording`
Bool.
- `bool liveViewActive`
Bool.
- `bool playbackActive`
Bool.
- `int mode`
Integer.
- `int deviceNumber`
Integer.
- `QString file`
QString.
- `int rec_width`
Integer.
- `int rec_height`
Integer.

5.8.1 Detailed Description

`MainWindow` class creates the main application window and builds the basis.

Main windows of the application which does most of the work, like connection functions and buttons as well as the initialization of the different mode. Also function for recording and saving an image are provided here.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 `MainWindow::MainWindow (QWidget * parent = 0) [explicit]`

Constructor of the main window to set up the UI from the Qt file.

Parameters

<i>parent</i>	an QWidget argument. If no parameter is give parent = 0.
---------------	--

5.8.3 Member Function Documentation

5.8.3.1 void MainWindow::changeDetectionState (bool *detection*) [signal]

Signal: Change detection state flag.

This signal emits the actual detection flag to the connected slot.

Parameters

<i>detection</i>	a bool argument.
------------------	------------------

5.8.3.2 void MainWindow::newImageProcessingFlags (struct ImageProcessingFlags *imageProcessingFlags*)
[signal]

Signal: Send new [ImageProcessingFlags](#).

This signal emits the actual [ImageProcessingFlags](#) to the connected slot.

Parameters

<i>image-ProcessingFlags</i>	a ImageProcessingFlags structure.
------------------------------	---

5.8.3.3 void MainWindow::setLiveViewSaveImgFlag (bool *sFlag*) [signal]

Signal: Set LiveView snapshot flag.

This signal emits save sflag to the connected slot in [ProcessingThread](#).

Parameters

<i>sFlag</i>	a bool argument.
--------------	------------------

5.8.3.4 void MainWindow::setRecordFlag (bool *rFlag*) [signal]

Signal: Set record flag.

This signal emits record rflag to the connected slot in [ProcessingThread](#).

Parameters

<i>rFlag</i>	a bool argument.
--------------	------------------

5.8.3.5 void MainWindow::setROI (QRect *roi*) [signal]

Signal: Set Region of interest.

This signal emits the desired ROI to the connected slot.

Parameters

<i>roi</i>	a QRect argument.
------------	-------------------

5.8.3.6 void MainWindow::setSaveParams (QString *dir*, QString *fn*) [signal]

Signal: Set save parameter.

This signal emits the save parameters to the connected slot.

Parameters

<i>dir</i>	a QString argument with Directory name.
<i>fn</i>	a QString argument with filename.

5.8.3.7 void MainWindow::showImage (QString *fn*) [private]

Function to show a static image from an image file.

Parameters

<i>fn</i>	a QString with the filename to load.
-----------	--------------------------------------

5.8.3.8 void MainWindow::startPlayback (QString *filename*) [private]

Function to start displaying a video from a file.

Parameters

<i>filename</i>	a QString with the filename to load.
-----------------	--------------------------------------

5.8.3.9 void MainWindow::updateFrame (const QPixmap & *frame*) [private], [slot]

Private slot to update the shown image in [MainWindow](#).

Parameters

<i>frame</i>	a QPixmap to load in specific label.
--------------	--------------------------------------

5.8.4 Member Data Documentation

5.8.4.1 bool MainWindow::detection [private]

Bool.

Boolean value if detection (in video or liveview shall be done).

5.8.4.2 int MainWindow::deviceNumber [private]

Integer.

Integer value for device number.

5.8.4.3 QString MainWindow::file [private]

QString.

Name of a file.

5.8.4.4 bool MainWindow::imageDetection [private]

Bool.

Boolean value if image detection shall be done.

5.8.4.5 ImageHandler* MainWindow::imageHandler [private]

[ImageHandler](#) pointer.

Pointer to a imageHandler.

5.8.4.6 ImageProcessingFlags MainWindow::imageProcessingFlags [private]

[ImageProcessingFlags](#) struct.

Structure of several flags.

5.8.4.7 ImageProcessingSettings MainWindow::imageProcessingSettings [private]

[ImageProcessingSettings](#) struct.

Structure of several settings.

5.8.4.8 ImageProcessingSettingsDialog* MainWindow::imageProcessingSettingsDialog [private]

[ImageProcessingSettingsDialog](#) pointer.

Pointer to a imageProcessingSettingsDialog.

5.8.4.9 bool MainWindow::liveViewActive [private]

Bool.

Boolean value to determine if liveview mode is on.

5.8.4.10 int MainWindow::mode [private]

Integer.

Integer value to determine the mode of the program.

5.8.4.11 bool MainWindow::playbackActive [private]

Bool.

Boolean value to determine if playback mode is on.

5.8.4.12 PlaybackThread* MainWindow::playbackThread [private]

[PlaybackThread](#) pointer.

Pointer to a playbackThread.

5.8.4.13 ProcessingThread* MainWindow::processingThread [private]

[ProcessingThread](#) pointer.

Pointer to a processingThread.

5.8.4.14 `int MainWindow::rec_height` [private]

Integer.

Integer value with recording height.

5.8.4.15 `int MainWindow::rec_width` [private]

Integer.

Integer value with recording width.

5.8.4.16 `bool MainWindow::recording` [private]

Bool.

Boolean value to determine if recording is on.

5.8.4.17 `RecordVideoThread* MainWindow::recordThread` [private]

[RecordVideoThread](#) pointer.

Pointer to a recordThread.

5.8.4.18 `QString MainWindow::saveDirectory` [private]

QString.

Directory to record a video.

5.8.4.19 `Ui::MainWindow* MainWindow::ui` [private]

[MainWindow](#) UI pointer.

Pointer the the user interface file of the dialog.

5.8.4.20 `VideoImageBuffer* MainWindow::videolImageBuffer` [private]

[VideoImageBuffer](#) pointer.

Pointer to a videolImageBuffer.

5.8.4.21 `VideoThread* MainWindow::videoThread` [private]

[VideoThread](#) pointer.

Pointer to a videoThread.

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

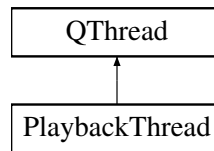
- MainWindow.h
- MainWindow.cpp

5.9 PlaybackThread Class Reference

Playback class which creates a thread to play a video.

```
#include <PlaybackThread.h>
```

Inheritance diagram for PlaybackThread:



Signals

- void [nextFrame](#) (const QPixmap &[frame](#))
Signal: Next frame.

Public Member Functions

- [PlaybackThread](#) (QString filename)
Thread constructor class.
- void [stop](#) ()
Set stop flag for thread worker.
- bool [loadPlayback](#) ()
Function to load video file to VideoCapture.

Protected Member Functions

- void [run](#) ()
Thread function which runs till doStop is set to true.

Private Slots

- void [updateImageProcessingFlags](#) (struct [ImageProcessingFlags](#))
Private slot to update the [ImageProcessingFlags](#).
- void [updateImageProcessingSettings](#) (struct [ImageProcessingSettings](#))
Private slot to update the [ImageProcessingSettings](#).
- void [updateDetState](#) (bool)
Private slot to update the detection state.

Private Attributes

- [FaceDetection](#) * [faceDetection](#)
FaceDetection.
- QMutex [doStopMutex](#)
QMutex.
- QMutex [processingMutex](#)
QMutex.
- volatile bool [doStop](#)

- Bool.*
- VideoCapture [cap](#)
- VideoCapture.*
- Mat [grabbedFrame](#)
- Mat.*
- QPixmap [frame](#)
- QPixmap.*
- QString [fn](#)
- QString.*
- int [fps](#)
- Integer.*
- int [frames](#)
- Integer.*
- int [counter](#)
- Integer.*
- bool [detectionState](#)
- Boolean.*
- struct [ImageProcessingFlags](#) [imgProcFlags](#)
- ImageProcessingFlags struct.*
- struct [ImageProcessingSettings](#) [imgProcSettings](#)
- ImageProcessingSettings struct.*

5.9.1 Detailed Description

Playback class which creates a thread to play a video.

Thread creating class to playback a video from file.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 PlaybackThread::PlaybackThread (QString *filename*)

Thread constructor class.

Parameters

<i>filename</i>	a QString argument which contains the video filename.
-----------------	---

5.9.3 Member Function Documentation

5.9.3.1 void PlaybackThread::nextFrame (const QPixmap & *frame*) [signal]

Signal: Next frame.

This signal emits the next frame to the main window to display it.

Parameters

<i>frame</i>	a QPixmap image.
--------------	------------------

5.9.3.2 void PlaybackThread::run () [protected]

Thread function which runs till doStop is set to true.

To function is the worker of the thread. It does all the calls to detection functions and emits new images.

5.9.3.3 void PlaybackThread::updateDetState (bool *flag*) [private],[slot]

Private slot to update the detection state.

Parameters

<i>flag</i>	a bool value.
-------------	---------------

5.9.3.4 void PlaybackThread::updateImageProcessingFlags (struct ImageProcessingFlags *imgProcFlags*) [private],[slot]

Private slot to update the [ImageProcessingFlags](#).

Parameters

<i>imgProcFlags</i>	a ImageProcessingFlags struct.
---------------------	--

5.9.3.5 void PlaybackThread::updateImageProcessingSettings (struct ImageProcessingSettings *imgProcSettings*) [private],[slot]

Private slot to update the [ImageProcessingSettings](#).

Parameters

<i>imgProcSettings</i>	a ImageProcessingSettings struct.
------------------------	---

5.9.4 Member Data Documentation

5.9.4.1 VideoCapture PlaybackThread::cap [private]

VideoCapture.

VideoCapture reads in the images from the given file.

5.9.4.2 int PlaybackThread::counter [private]

Integer.

Integer counter value.

5.9.4.3 bool PlaybackThread::detectionState [private]

Boolean.

Boolean value to make detection or not.

5.9.4.4 volatile bool PlaybackThread::doStop [private]

Bool.

Boolean value to stop the thread.

5.9.4.5 QMutex PlaybackThread::doStopMutex [private]

QMutex.

Mutex to stop the thread.

5.9.4.6 FaceDetection* PlaybackThread::faceDetection [private]

[FaceDetection](#).

Pointer to instance of [FaceDetection](#) class.

5.9.4.7 QString PlaybackThread::fn [private]

QString.

Filename of the video file.

5.9.4.8 int PlaybackThread::fps [private]

Integer.

Integer value which contains the framerate.

5.9.4.9 QPixmap PlaybackThread::frame [private]

QPixmap.

Converted image in QPixmap format which will be emitted to main window to display

5.9.4.10 int PlaybackThread::frames [private]

Integer.

Integer value which contains the number of grabbed frames.

5.9.4.11 Mat PlaybackThread::grabbedFrame [private]

Mat.

Grabbed frame from the video file.

5.9.4.12 struct ImageProcessingFlags PlaybackThread::imgProcFlags [private]

[ImageProcessingFlags](#) struct.

Structure which contains information about the flags.

5.9.4.13 struct ImageProcessingSettings PlaybackThread::imgProcSettings [private]

[ImageProcessingSettings](#) struct.

Structure which contains information about the settings.

5.9.4.14 QMutex PlaybackThread::processingMutex [private]

QMutex.

Mutex which is set while processing the image.

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

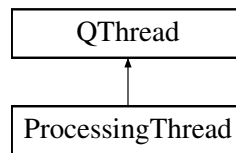
- PlaybackThread.h
- PlaybackThread.cpp

5.10 ProcessingThread Class Reference

Processing class which does image detection in webcam mode.

```
#include <ProcessingThread.h>
```

Inheritance diagram for ProcessingThread:



Signals

- void [newFrame](#) (const QPixmap &frame)
Signal: New frame.
- void [recFrame](#) (const Mat &img)
Signal: Record frame.

Public Member Functions

- [ProcessingThread](#) (VideolImageBuffer *videolImageBuffer, int deviceNumber)
Thread constructor class.
- QRect [getCurrentROI](#) ()
Function to get the current region of interest.
- void [stop](#) ()
Set stop flag for thread worker.

Protected Member Functions

- void [run](#) ()
Thread function which runs till doStop is set to true.

Private Slots

- void [updateImageProcessingFlags](#) (struct ImageProcessingFlags)
Private slot to update the ImageProcessingFlags.
- void [updateImageProcessingSettings](#) (struct ImageProcessingSettings)
Private slot to update the ImageProcessingSettings.
- void [setROI](#) (QRect roi)
Private slot to set region of interest.
- void [setSaveParams](#) (QString dir, QString fn)
Private slot to set save parameters.
- void [setSaveImgFlag](#) (bool sFlag)
Private slot to set save image sFlag.
- void [setFrameProcessing](#) (bool flag)
Private slot to set frame processing flag.
- void [setRecFlag](#) (bool rFlag)
Private slot to set recording rFlag.

Private Member Functions

- void [setROI](#) ()
Function to set ROI.
- void [resetROI](#) ()
Function to reset ROI.
- bool [saveCurrentImage](#) (Mat)
Function to save current image.

Private Attributes

- [FaceDetection](#) * [faceDetection](#)
FaceDetection.
- [VideoImageBuffer](#) * [videoImageBuffer](#)
VideoImageBuffer.
- Mat [currentFrame](#)
Mat image.
- Mat [currentFrameGrayscale](#)
Mat image.
- QPixmap [frame](#)
QPixmap image.
- QMutex [doStopMutex](#)
QMutex.
- QMutex [processingMutex](#)
QMutex.
- Size [frameSize](#)
Size.
- Point [framePoint](#)
Point.
- Rect [currentROI](#)
Rect.
- struct [ImageProcessingFlags](#) [imgProcFlags](#)
ImageProcessingFlags struct.
- struct [ImageProcessingSettings](#) [imgProcSettings](#)
ImageProcessingSettings struct.
- volatile bool [doStop](#)
Bool.
- int [deviceNumber](#)
Integer.
- bool [enableFrameProcessing](#)
Bool.
- bool [saveImage](#)
Bool.
- QString [recDir](#)
QString.
- QString [recFn](#)
QString.
- bool [recFlag](#)
Bool.

5.10.1 Detailed Description

Processing class which does image detection in webcam mode.

Thread processing captured images.

5.10.2 Constructor & Destructor Documentation

5.10.2.1 ProcessingThread::ProcessingThread (**VideolImageBuffer** * *videolImageBuffer*, int *deviceNumber*)

Thread constructor class.

Parameters

<i>videolImage-Buffer</i>	pointer to a buffer which contains the captured images.
<i>deviceNumber</i>	an integer argument.

5.10.3 Member Function Documentation

5.10.3.1 QRect ProcessingThread::getCurrentROI ()

Function to get the current region of interest.

Returns

QRect - The current ROI

5.10.3.2 void ProcessingThread::newFrame (const QPixmap & *frame*) [signal]

Signal: New frame.

This signale emits the next frame to the main window to display it.

Parameters

<i>frame</i>	a QPixmap image.
--------------	------------------

5.10.3.3 void ProcessingThread::recFrame (const Mat & *img*) [signal]

Signal: Record frame.

This signale emits the next frame for recording.

Parameters

<i>img</i>	a Mat image.
------------	--------------

5.10.3.4 void ProcessingThread::run () [protected]

Thread function which runs till doStop is set to true.

To function is the worker of the thread. It does all the calls to detection functions and emits new images.

5.10.3.5 bool ProcessingThread::saveCurrentImage (Mat *img*) [private]

Function to save current image.

Parameters

<i>img</i>	a Mat image argument.
------------	-----------------------

Returns

Boolean value if process was successful.

5.10.3.6 void ProcessingThread::setFrameProcessing (bool *flag*) [private],[slot]

Private slot to set frame processing flag.

Parameters

<i>flag</i>	a bool argument.
-------------	------------------

5.10.3.7 void ProcessingThread::setRecFlag (bool *rFlag*) [private],[slot]

Private slot to set recording rFlag.

Parameters

<i>rFlag</i>	a bool argument.
--------------	------------------

5.10.3.8 void ProcessingThread::setROI (QRect *roi*) [private],[slot]

Private slot to set region of interest.

Parameters

<i>roi</i>	a QRect argument.
------------	-------------------

5.10.3.9 void ProcessingThread::setSavelmgFlag (bool *sFlag*) [private],[slot]

Private slot to set save image sFlag.

Parameters

<i>sFlag</i>	a bool argument.
--------------	------------------

5.10.3.10 void ProcessingThread::setSaveParams (QString *dir*, QString *fn*) [private],[slot]

Private slot to set save parameters.

Parameters

<i>dir</i>	a QString argument which contains the Directory name.
<i>fn</i>	a QString argument which contains the file name.

5.10.3.11 void ProcessingThread::updateImageProcessingFlags (struct ImageProcessingFlags *imgProcFlags*) [private],[slot]

Private slot to update the [ImageProcessingFlags](#).

Parameters

<i>imgProcFlags</i>	a ImageProcessingFlags struct.
---------------------	--

5.10.3.12 void ProcessingThread::updateImageProcessingSettings (struct ImageProcessingSettings *imgProcSettings*)
[private], [slot]

Private slot to update the [ImageProcessingSettings](#).

Parameters

<i>imgProcSettings</i>	a ImageProcessingSettings struct.
------------------------	---

5.10.4 Member Data Documentation

5.10.4.1 Mat ProcessingThread::currentFrame [private]

Mat image.

Current image.

5.10.4.2 Mat ProcessingThread::currentFrameGrayscale [private]

Mat image.

Current image in grayscale.

5.10.4.3 Rect ProcessingThread::currentROI [private]

Rect.

Current region of interest.

5.10.4.4 int ProcessingThread::deviceNumber [private]

Integer.

Integer value for device number.

5.10.4.5 volatile bool ProcessingThread::doStop [private]

Bool.

Boolean value to stop the thread.

5.10.4.6 QMutex ProcessingThread::doStopMutex [private]

QMutex.

Mutex to stop the thread.

5.10.4.7 bool ProcessingThread::enableFrameProcessing [private]

Bool.

Boolean value if frame processing is enabled.

5.10.4.8 **FaceDetection*** ProcessingThread::faceDetection [private]

[FaceDetection](#).

Pointer to class instance for [FaceDetection](#).

5.10.4.9 **QPixmap** ProcessingThread::frame [private]

QPixmap image.

Current image converted to QPixmap format.

5.10.4.10 **Point** ProcessingThread::framePoint [private]

Point.

Frame point.

5.10.4.11 **Size** ProcessingThread::frameSize [private]

Size.

Size of the frame.

5.10.4.12 **struct ImageProcessingFlags** ProcessingThread::imgProcFlags [private]

[ImageProcessingFlags](#) struct.

Structure which contains information about the flags.

5.10.4.13 **struct ImageProcessingSettings** ProcessingThread::imgProcSettings [private]

[ImageProcessingSettings](#) struct.

Structure which contains information about the settings.

5.10.4.14 **QMutex** ProcessingThread::processingMutex [private]

QMutex.

Mutex which is set while processing the image.

5.10.4.15 **QString** ProcessingThread::recDir [private]

QString.

Directory to record a video.

5.10.4.16 **bool** ProcessingThread::recFlag [private]

Bool.

State if video shall be recorded.

5.10.4.17 QString ProcessingThread::recFn [private]

QString.

Filename of the recorded video.

5.10.4.18 bool ProcessingThread::saveImage [private]

Bool.

Boolean value if image shall be saved.

5.10.4.19 QImage* ProcessingThread::videoImageBuffer [private]

[QImage](#).

Pointer to buffer which contains the images.

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

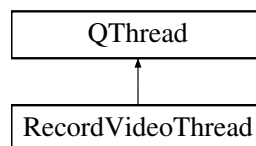
- ProcessingThread.h
- ProcessingThread.cpp

5.11 RecordVideoThread Class Reference

Record Video class which records the processed images from the webcam to a video file.

```
#include <RecordVideoThread.h>
```

Inheritance diagram for RecordVideoThread:



Public Member Functions

- [RecordVideoThread](#) (QString filename, int height, int width)
Thread constructor class.
- void [stop](#) ()
Set stop flag for thread worker.
- bool [createVideoWriter](#) ()
Function to create a new video writer.

Protected Member Functions

- void [run](#) ()
Thread function which runs till doStop is set to true.

Private Slots

- void [addRecFrame](#) (const Mat &frame)
Private slot to add next image to video file.

Private Attributes

- QMutex [doStopMutex](#)
QMutex.
- QMutex [recordingMutex](#)
QMutex.
- volatile bool [doStop](#)
Bool.
- VideoWriter [vid_writer](#)
VideoWriter.
- QString [fn](#)
QString.
- int [rec_width](#)
Integer.
- int [rec_height](#)
Integer.
- int [rec_fps](#)
Integer.

5.11.1 Detailed Description

Record Video class which records the processed images from the webcam to a video file.

Thread to record processed images.

5.11.2 Constructor & Destructor Documentation

5.11.2.1 RecordVideoThread::RecordVideoThread (QString *filename*, int *height*, int *width*)

Thread constructor class.

Parameters

<i>filename</i>	a QString argument - Filename for recording.
<i>height</i>	an integer argument with the height of the video.
<i>width</i>	an integer argument with the width of the video.

5.11.3 Member Function Documentation

5.11.3.1 void RecordVideoThread::addRecFrame (const Mat & *frame*) [private], [slot]

Private slot to add next image to video file.

Parameters

<i>frame</i>	a Mat image to add to recorded vide file.
--------------	---

5.11.3.2 bool RecordVideoThread::createVideoWriter ()

Function to create a new video writer.

Returns

bool - True if successfull.

5.11.3.3 void RecordVideoThread::run () [protected]

Thread function which runs till doStop is set to true.

To function is the worker of the thread. It does all the calls to detection functions and emits new images.

5.11.4 Member Data Documentation

5.11.4.1 volatile bool RecordVideoThread::doStop [private]

Bool.

Boolean value to stop the thread.

5.11.4.2 QMutex RecordVideoThread::doStopMutex [private]

QMutex.

Mutex to stop the thread.

5.11.4.3 QString RecordVideoThread::fn [private]

QString.

Filename of the recorded video.

5.11.4.4 int RecordVideoThread::rec_fps [private]

Integer.

Integer value with recording framerate.

5.11.4.5 int RecordVideoThread::rec_height [private]

Integer.

Integer value with recording height.

5.11.4.6 int RecordVideoThread::rec_width [private]

Integer.

Integer value with recording width.

5.11.4.7 QMutex RecordVideoThread::recordingMutex [private]

QMutex.

Mutex which is set while recording the image.

5.11.4.8 VideoWriter RecordVideoThread::vid_writer [private]

VideoWriter.

VideoWriter from OpenCV to record video to file.

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

- RecordVideoThread.h
- RecordVideoThread.cpp

5.12 VideoImageBuffer Class Reference

Video buffer class.

```
#include <VideoImageBuffer.h>
```

Public Member Functions

- [VideoImageBuffer](#) ()
Constructor of [VideoImageBuffer](#).
- void [add](#) (int deviceNumber, [Buffer](#)< Mat > *imageBuffer)
Function to add a Image buffer.
- [Buffer](#)< Mat > * [getByDeviceNumber](#) (int deviceNumber)
Get the buffer for specific device number.
- void [removeByDeviceNumber](#) (int deviceNumber)
Remove a buffer for a specific device number.
- void [wakeAll](#) ()
Wake all QWaitConditions.
- bool [containsImageBufferForDeviceNumber](#) (int deviceNumber)
Function to check if the Video image buffer contains a buffer for a device number.

Private Attributes

- QHash< int, [Buffer](#)< Mat > * > [imageBufferMap](#)
QHash<int, Buffer<Mat>>*
- QSet< int > [syncSet](#)
QSet<int>
- QWaitCondition [wc](#)
QWaitCondition.
- QMutex [mutex](#)
QMutex.

5.12.1 Detailed Description

Video buffer class.

Video buffer class to share images between two or more threads.

5.12.2 Member Function Documentation

5.12.2.1 void VideoImageBuffer::add (int deviceNumber, [Buffer](#)< Mat > * imageBuffer)

Function to add a Image buffer.

Parameters

<i>deviceNumber</i>	an integer argument with the device number.
<i>imageBuffer</i>	a pointer to the image buffer to add (Type Buffer<Mat>).

5.12.2.2 bool VideoImageBuffer::containsImageBufferForDeviceNumber (int *deviceNumber*)

Function to check if the Video image buffer contains a buffer for a device number.

Parameters

<i>deviceNumber</i>	an integer argument with the device number.
---------------------	---

Returns

True if [VideoImageBuffer](#) contains buffer for specific device number

5.12.2.3 Buffer< Mat > * VideoImageBuffer::getByDeviceNumber (int *deviceNumber*)

Get the buffer for specific device number.

Parameters

<i>deviceNumber</i>	an integer argument with the device number.
---------------------	---

Returns

Pointer the requested buffer of type Mat.

5.12.2.4 void VideoImageBuffer::removeByDeviceNumber (int *deviceNumber*)

Remove a buffer for a specific device number.

Parameters

<i>deviceNumber</i>	an integer argument with the device number.
---------------------	---

5.12.3 Member Data Documentation

5.12.3.1 QHash<int, Buffer<Mat>*> VideoImageBuffer::imageBufferMap [private]

QHash<int, Buffer<Mat>*>

A map which contains all pointers to the added buffers

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

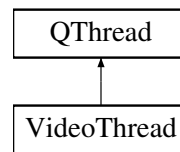
- VideoImageBuffer.h
- VideoImageBuffer.cpp

5.13 VideoThread Class Reference

Video class which captures an image from a webcam or specific device.

```
#include <VideoThread.h>
```

Inheritance diagram for VideoThread:



Public Member Functions

- [VideoThread](#) ([VideoImageBuffer](#) *[videolImageBuffer](#), int [deviceNumber](#), bool [dropFrameIfBufferFull](#))
Thread constructor class.
- void [stop](#) ()
Set stop flag for thread worker.
- bool [connectToCamera](#) ()
Function to create a camera connection.
- bool [disconnectCamera](#) ()
Function to disconnect a camera connection.
- bool [isCameraConnected](#) ()
Function to check if a camera is connected.
- int [getInputSourceWidth](#) ()
Function to get the camera input width of an image.
- int [getInputSourceHeight](#) ()
Function to get the camera input height of an image.

Protected Member Functions

- void [run](#) ()
Thread function which runs till doStop is set to true.

Private Attributes

- [VideoImageBuffer](#) * [videolImageBuffer](#)
VideoImageBuffer.
- VideoCapture [cap](#)
VideoCapture.
- Mat [grabbedFrame](#)
Mat image.
- QMutex [doStopMutex](#)
QMutex.
- volatile bool [doStop](#)
Bool.
- int [deviceNumber](#)
Integer.
- bool [dropFrameIfBufferFull](#)
Bool.

5.13.1 Detailed Description

Video class which captures an image from a webcam or specific device.

Thread to capture images.

5.13.2 Constructor & Destructor Documentation

5.13.2.1 VideoThread::VideoThread (VideoImageBuffer * *videoImageBuffer*, int *deviceNumber*, bool *dropFrameIfBufferFull*)

Thread constructor class.

Parameters

<i>videoImage-Buffer</i>	pointer to a buffer which contains the captured images.
<i>deviceNumber</i>	an integer argument.
<i>dropFrameIf-BufferFull</i>	a bool value to set if images shall be dropped when the buffer is full.

5.13.3 Member Function Documentation

5.13.3.1 bool VideoThread::connectToCamera ()

Function to create a camera connection.

Returns

Returns true when attempt was successful.

5.13.3.2 bool VideoThread::disconnectCamera ()

Function to disconnect a camera connection.

Returns

Returns true when attempt was successful.

5.13.3.3 int VideoThread::getInputSourceHeight ()

Function to get the camera input height of an image.

Returns

Returns the height as an integer value.

5.13.3.4 int VideoThread::getInputSourceWidth ()

Function to get the camera input width of an image.

Returns

Returns the width as an integer value.

5.13.3.5 bool VideoThread::isCameraConnected ()

Function to check if a camera is connected.

Returns

Returns true when a camera is connected.

5.13.3.6 void VideoThread::run () [protected]

Thread function which runs till doStop is set to true.

To function is the worker of the thread. It does all the calls to detection functions and emits new images.

5.13.4 Member Data Documentation

5.13.4.1 VideoCapture VideoThread::cap [private]

VideoCapture.

Variable to store the camera connection. From OpenCV lib.

5.13.4.2 int VideoThread::deviceNumber [private]

Integer.

Integer value for device number.

5.13.4.3 volatile bool VideoThread::doStop [private]

Bool.

Boolean value to stop the thread.

5.13.4.4 QMutex VideoThread::doStopMutex [private]

QMutex.

Mutex to stop the thread.

5.13.4.5 bool VideoThread::dropFrameIfBufferFull [private]

Bool.

Boolean value if a frame shall be dropped when the buffer is full.

5.13.4.6 Mat VideoThread::grabbedFrame [private]

Mat image.

Grabbed frame from input source.

5.13.4.7 QImage* VideoThread::videoImageBuffer [private]

[VideoImageBuffer](#).

Pointer to buffer which contains the images.

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

- VideoThread.h
- VideoThread.cpp

Chapter 6

File Documentation

6.1 Config.h File Reference

Structure file. It contains the general stored settings which are loaded at the beginning.

Macros

- `#define DEFAULT_IMAGE_BUFFER_SIZE 1`
- `#define OVERLAY_PARAM_1 0`
- `#define OVERLAY_PARAM_2 0`
- `#define OVERLAY_PARAM_3 0`

6.1.1 Detailed Description

Structure file. It contains the general stored settings which are loaded at the beginning.

6.1.2 Macro Definition Documentation

6.1.2.1 `#define DEFAULT_IMAGE_BUFFER_SIZE 1`

Default image buffer size for the software

6.1.2.2 `#define OVERLAY_PARAM_1 0`

Standard value for reset functions. This values defines the glasses as overlay

6.1.2.3 `#define OVERLAY_PARAM_2 0`

Standard value for reset functions. This values defines the red nose as overlay

6.1.2.4 `#define OVERLAY_PARAM_3 0`

Standard value for reset functions. This values defines the red lips as overlay

6.2 ImageConversion.h File Reference

Image conversion functions. Four function to convert images between the following three types: OpenCV Mat, QImage, QPixmap.

```
#include <iostream>
#include <QImage>
#include <QPixmap>
#include <opencv2/imgproc/imgproc.hpp>
#include <opencv2/imgproc/types_c.h>
```

Functions

- QImage [cvMatToQImage](#) (const cv::Mat &inMat)
Image conversion function which converts an OpenCV Mat image to an Qt QImage.
- cv::Mat [QImageToCvMat](#) (const QImage &inImage, bool inCloneImageData)
Image conversion function which converts an OpenCV Mat image to an Qt QImage.
- QPixmap [cvMatToQPixmap](#) (const cv::Mat &inMat)
Inline function for direct conversion from Mat to QPixmap format.
- cv::Mat [QPixmapToCvMat](#) (const QPixmap &inPixmap, bool inCloneImageData=true)
Inline function for direct conversion from QPixmap to Mat format.

6.2.1 Detailed Description

Image conversion functions. Four function to convert images between the following three types: OpenCV Mat, QImage, QPixmap.

6.2.2 Function Documentation

6.2.2.1 QImage cvMatToQImage (const cv::Mat & *inMat*)

Image conversion function which converts an OpenCV Mat image to an Qt QImage.

Parameters

<i>inMat</i>	a Mat image to convert.
--------------	-------------------------

Returns

The converted image as QImage (Qt format).

6.2.2.2 QPixmap cvMatToQPixmap (const cv::Mat & *inMat*) [inline]

Inline function for direct conversion from Mat to QPixmap format.

Parameters

<i>inMat</i>	a Mat image to convert.
--------------	-------------------------

Returns

The converted image as QPixmap (Qt format).

6.2.2.3 cv::Mat QImageToCvMat (const QImage & *inImage*, bool *inCloneImageData*)

Image conversion function which converts an OpenCV Mat image to an Qt QImage.

Parameters

<i>inImage</i>	a QImage image to convert.
<i>inCloneImageData</i>	a boolean value if image should be cloned for conversion.

Returns

The converted image as Mat (OpenCV format).

6.2.2.4 `cv::Mat QPixmapToCvMat (const QPixmap & inPixmap, bool inCloneImageData = true) [inline]`

Inline function for direct conversion from QPixmap to Mat format.

Parameters

<i>inPixmap</i>	a QPixmap image to convert.
<i>inCloneImageData</i>	a boolean value if image should be cloned for conversion.

Returns

The converted image as Mat (OpenCV format).

6.3 Structures.h File Reference

Structures of the program. This file contains to structures to share settings and flags programm wide.

Classes

- struct [ImageProcessingSettings](#)
Structure which contains information about the overlays apply on the image.
- struct [ImageProcessingFlags](#)
Structure which contains information about the flags. Determine what process shall be done.

6.3.1 Detailed Description

Structures of the program. This file contains to structures to share settings and flags programm wide.

Index

- add
 - Buffer, 10
 - VideoImageBuffer, 42
- addRecFrame
 - RecordVideoThread, 40
- applyChangesToImageProc
 - ImageHandler, 15
- Buffer
 - add, 10
 - Buffer, 10
 - bufferSize, 11
 - clear, 10
 - clearBuffer_add, 11
 - clearBuffer_get, 11
 - freeSlots, 11
 - get, 10
 - isEmpty, 10
 - isFull, 11
 - maxSize, 11
 - queue, 12
 - queueProtect, 12
 - size, 11
 - usedSlots, 12
- Buffer< T >, 9
- bufferSize
 - Buffer, 11
- CameraConnectDialog, 12
 - CameraConnectDialog, 13
 - CameraConnectDialog, 13
 - getMode, 13
 - ui, 13
- cap
 - PlaybackThread, 31
 - VideoThread, 46
- changeDetectionState
 - MainWindow, 24
- clear
 - Buffer, 10
- clearBuffer_add
 - Buffer, 11
- clearBuffer_get
 - Buffer, 11
- Config.h, 47
 - OVERLAY_PARAM_1, 47
 - OVERLAY_PARAM_2, 47
 - OVERLAY_PARAM_3, 47
- connectToCamera
 - VideoThread, 45
- containsImageBufferForDeviceNumber
 - VideoImageBuffer, 43
- counter
 - PlaybackThread, 31
- createVideoWriter
 - RecordVideoThread, 40
- currentFrame
 - ProcessingThread, 37
- currentFrameGrayscale
 - ProcessingThread, 37
- currentROI
 - ProcessingThread, 37
- cvCreateOverlay
 - FaceDetection, 14
- cvCreateOverlayGlasses
 - FaceDetection, 14
- cvMatToQImage
 - ImageConversion.h, 48
- cvMatToQPixmap
 - ImageConversion.h, 48
- detectObjects
 - FaceDetection, 14
- detection
 - MainWindow, 26
- detectionState
 - PlaybackThread, 31
- deviceNumber
 - MainWindow, 26
 - ProcessingThread, 37
 - VideoThread, 46
- disconnectCamera
 - VideoThread, 45
- doStop
 - PlaybackThread, 31
 - ProcessingThread, 37
 - RecordVideoThread, 41
 - VideoThread, 46
- doStopMutex
 - PlaybackThread, 31
 - ProcessingThread, 37
 - RecordVideoThread, 41
 - VideoThread, 46
- dropFrameIfBufferFull
 - VideoThread, 46
- enableFrameProcessing
 - ProcessingThread, 37
- FaceDetection, 13

- cvCreateOverlay, 14
 - cvCreateOverlayGlasses, 14
 - detectObjects, 14
- faceDetection
 - ImageHandler, 16
 - PlaybackThread, 31
 - ProcessingThread, 37
- file
 - MainWindow, 26
- fn
 - PlaybackThread, 32
 - RecordVideoThread, 41
- fps
 - PlaybackThread, 32
- frame
 - PlaybackThread, 32
 - ProcessingThread, 38
- framePoint
 - ProcessingThread, 38
- frameSize
 - ProcessingThread, 38
- frames
 - PlaybackThread, 32
- freeSlots
 - Buffer, 11
- get
 - Buffer, 10
- getByDeviceNumber
 - VideoImageBuffer, 43
- getCurrentROI
 - ProcessingThread, 35
- getImgProcSettingsForImg
 - ImageProcessingSettingsDialog, 20
- getInputSourceHeight
 - VideoThread, 45
- getInputSourceWidth
 - VideoThread, 45
- getMode
 - CameraConnectDialog, 13
- grabbedFrame
 - PlaybackThread, 32
 - VideoThread, 46
- image
 - ImageHandler, 16
- imageBufferMap
 - VideoImageBuffer, 43
- ImageConversion.h, 48
 - cvMatToQImage, 48
 - cvMatToQPixmap, 48
 - QImageToCvMat, 48
 - QPixmapToCvMat, 50
- imageDetection
 - MainWindow, 26
- ImageHandler, 15
 - applyChangesToImageProc, 15
 - faceDetection, 16
 - image, 16
 - loadImageFromFile, 16
 - orig_image, 16
 - saveImageToFile, 16
- imageHandler
 - MainWindow, 26
- ImageProcessingFlags, 17
 - showDetectionOn, 17
 - showOverlaysOn, 17
- imageProcessingFlags
 - MainWindow, 27
- ImageProcessingSettings, 17
 - overlayParam1, 18
 - overlayParam2, 18
 - overlayParam3, 18
- imageProcessingSettings
 - ImageProcessingSettingsDialog, 20
 - MainWindow, 27
- ImageProcessingSettingsDialog, 18
 - getImgProcSettingsForImg, 20
 - imageProcessingSettings, 20
 - ImageProcessingSettingsDialog, 19
 - ImageProcessingSettingsDialog, 19
 - ImgProcFlagsForImg, 20
 - newImageProcessingSettings, 20
 - resetOverlays, 20
 - ui, 20
 - updateStoredSettingsFromDialog, 20
- imageProcessingSettingsDialog
 - MainWindow, 27
- imgProcFlags
 - PlaybackThread, 32
 - ProcessingThread, 38
- ImgProcFlagsForImg
 - ImageProcessingSettingsDialog, 20
- imgProcSettings
 - PlaybackThread, 32
 - ProcessingThread, 38
- isCameraConnected
 - VideoThread, 45
- isEmpty
 - Buffer, 10
- isFull
 - Buffer, 11
- liveViewActive
 - MainWindow, 27
- loadImageFromFile
 - ImageHandler, 16
- MainWindow, 21
 - changeDetectionState, 24
 - detection, 26
 - deviceNumber, 26
 - file, 26
 - imageDetection, 26
 - imageHandler, 26
 - imageProcessingFlags, 27
 - imageProcessingSettings, 27
 - imageProcessingSettingsDialog, 27

- liveViewActive, [27](#)
- MainWindow, [23](#)
- MainWindow, [23](#)
- mode, [27](#)
- newImageProcessingFlags, [25](#)
- playbackActive, [27](#)
- playbackThread, [27](#)
- processingThread, [27](#)
- rec_height, [27](#)
- rec_width, [28](#)
- recordThread, [28](#)
- recording, [28](#)
- saveDirectory, [28](#)
- setLiveViewSavelmgFlag, [25](#)
- setROI, [25](#)
- setRecordFlag, [25](#)
- setSaveParams, [25](#)
- showImage, [26](#)
- startPlayback, [26](#)
- ui, [28](#)
- updateFrame, [26](#)
- videoImageBuffer, [28](#)
- videoThread, [28](#)
- maxSize
 - Buffer, [11](#)
- mode
 - MainWindow, [27](#)
- newFrame
 - ProcessingThread, [35](#)
- newImageProcessingFlags
 - MainWindow, [25](#)
- newImageProcessingSettings
 - ImageProcessingSettingsDialog, [20](#)
- nextFrame
 - PlaybackThread, [30](#)
- OVERLAY_PARAM_1
 - Config.h, [47](#)
- OVERLAY_PARAM_2
 - Config.h, [47](#)
- OVERLAY_PARAM_3
 - Config.h, [47](#)
- orig_image
 - ImageHandler, [16](#)
- overlayParam1
 - ImageProcessingSettings, [18](#)
- overlayParam2
 - ImageProcessingSettings, [18](#)
- overlayParam3
 - ImageProcessingSettings, [18](#)
- playbackActive
 - MainWindow, [27](#)
- PlaybackThread, [29](#)
 - cap, [31](#)
 - counter, [31](#)
 - detectionState, [31](#)
 - doStop, [31](#)
 - doStopMutex, [31](#)
 - faceDetection, [31](#)
 - fn, [32](#)
 - fps, [32](#)
 - frame, [32](#)
 - frames, [32](#)
 - grabbedFrame, [32](#)
 - imgProcFlags, [32](#)
 - imgProcSettings, [32](#)
 - nextFrame, [30](#)
 - PlaybackThread, [30](#)
 - PlaybackThread, [30](#)
 - processingMutex, [32](#)
 - run, [30](#)
 - updateDetState, [30](#)
 - updateImageProcessingFlags, [31](#)
 - updateImageProcessingSettings, [31](#)
- playbackThread
 - MainWindow, [27](#)
- processingMutex
 - PlaybackThread, [32](#)
 - ProcessingThread, [38](#)
- ProcessingThread, [33](#)
 - currentFrame, [37](#)
 - currentFrameGrayscale, [37](#)
 - currentROI, [37](#)
 - deviceNumber, [37](#)
 - doStop, [37](#)
 - doStopMutex, [37](#)
 - enableFrameProcessing, [37](#)
 - faceDetection, [37](#)
 - frame, [38](#)
 - framePoint, [38](#)
 - frameSize, [38](#)
 - getCurrentROI, [35](#)
 - imgProcFlags, [38](#)
 - imgProcSettings, [38](#)
 - newFrame, [35](#)
 - processingMutex, [38](#)
 - ProcessingThread, [35](#)
 - ProcessingThread, [35](#)
 - recDir, [38](#)
 - recFlag, [38](#)
 - recFn, [38](#)
 - recFrame, [35](#)
 - run, [35](#)
 - saveCurrentImage, [35](#)
 - savelImage, [39](#)
 - setFrameProcessing, [36](#)
 - setROI, [36](#)
 - setRecFlag, [36](#)
 - setSavelmgFlag, [36](#)
 - setSaveParams, [36](#)
 - updateImageProcessingFlags, [36](#)
 - updateImageProcessingSettings, [37](#)
 - videoImageBuffer, [39](#)
- processingThread
 - MainWindow, [27](#)

- QImageToCvMat
 - ImageConversion.h, 48
- QPixmapToCvMat
 - ImageConversion.h, 50
- queue
 - Buffer, 12
- queueProtect
 - Buffer, 12
- rec_fps
 - RecordVideoThread, 41
- rec_height
 - MainWindow, 27
 - RecordVideoThread, 41
- rec_width
 - MainWindow, 28
 - RecordVideoThread, 41
- recDir
 - ProcessingThread, 38
- recFlag
 - ProcessingThread, 38
- recFn
 - ProcessingThread, 38
- recFrame
 - ProcessingThread, 35
- recordThread
 - MainWindow, 28
- RecordVideoThread, 39
 - addRecFrame, 40
 - createVideoWriter, 40
 - doStop, 41
 - doStopMutex, 41
 - fn, 41
 - rec_fps, 41
 - rec_height, 41
 - rec_width, 41
 - RecordVideoThread, 40
 - recordingMutex, 41
 - RecordVideoThread, 40
 - run, 40
 - vid_writer, 41
- recording
 - MainWindow, 28
- recordingMutex
 - RecordVideoThread, 41
- removeByDeviceNumber
 - VideoImageBuffer, 43
- resetOverlays
 - ImageProcessingSettingsDialog, 20
- run
 - PlaybackThread, 30
 - ProcessingThread, 35
 - RecordVideoThread, 40
 - VideoThread, 45
- saveCurrentImage
 - ProcessingThread, 35
- saveDirectory
 - MainWindow, 28
- saveImage
 - ProcessingThread, 39
- saveImageToFile
 - ImageHandler, 16
- setFrameProcessing
 - ProcessingThread, 36
- setLiveViewSaveImgFlag
 - MainWindow, 25
- setROI
 - MainWindow, 25
 - ProcessingThread, 36
- setRecFlag
 - ProcessingThread, 36
- setRecordFlag
 - MainWindow, 25
- setSaveImgFlag
 - ProcessingThread, 36
- setSaveParams
 - MainWindow, 25
 - ProcessingThread, 36
- showDetectionOn
 - ImageProcessingFlags, 17
- showImage
 - MainWindow, 26
- showOverlaysOn
 - ImageProcessingFlags, 17
- size
 - Buffer, 11
- startPlayback
 - MainWindow, 26
- Structures.h, 50
- ui
 - CameraConnectDialog, 13
 - ImageProcessingSettingsDialog, 20
 - MainWindow, 28
- updateDetState
 - PlaybackThread, 30
- updateFrame
 - MainWindow, 26
- updateImageProcessingFlags
 - PlaybackThread, 31
 - ProcessingThread, 36
- updateImageProcessingSettings
 - PlaybackThread, 31
 - ProcessingThread, 37
- updateStoredSettingsFromDialog
 - ImageProcessingSettingsDialog, 20
- usedSlots
 - Buffer, 12
- vid_writer
 - RecordVideoThread, 41
- VideoImageBuffer, 42
 - add, 42
 - containsImageBufferForDeviceNumber, 43
 - getByDeviceNumber, 43
 - imageBufferMap, 43
 - removeByDeviceNumber, 43

- videoImageBuffer
 - MainWindow, [28](#)
 - ProcessingThread, [39](#)
 - VideoThread, [46](#)
- VideoThread, [43](#)
 - cap, [46](#)
 - connectToCamera, [45](#)
 - deviceNumber, [46](#)
 - disconnectCamera, [45](#)
 - doStop, [46](#)
 - doStopMutex, [46](#)
 - dropFrameIfBufferFull, [46](#)
 - getInputSourceHeight, [45](#)
 - getInputSourceWidth, [45](#)
 - grabbedFrame, [46](#)
 - isCameraConnected, [45](#)
 - run, [45](#)
 - videoImageBuffer, [46](#)
 - VideoThread, [45](#)
 - VideoThread, [45](#)
- videoThread
 - MainWindow, [28](#)