VisualControl

0.1

Generated by Doxygen 1.7.2

Sat Feb 12 2011 17:15:03

Contents

Chapter 1

Todo List

Class cognition::EigenfaceRecognizer upgrade internals to the OpenCV C++ api (cv::Mat and cv::PCA)

Class cognition::Recognizer add method to add already loaded training images (cv::Mat)

Chapter 2

Directory Hierarchy

2.1 Directories

This directory hierarchy is sorted roughly, but not completely, alphabetically:

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 4

Class Index

4.1 Class Hierarchy

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

Chapter 5

Class Index

5.1 Class List

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

Chapter 6

File Index

6.1 File List

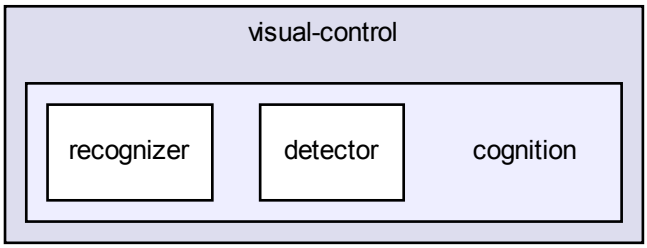
Here is a list of all files with brief descriptions:

Chapter 7

Directory Documentation

7.1 visual-control/cognition/ Directory Reference

Directory dependency graph for visual-control/cognition/:



Directories

- directory detector
- directory recognizer

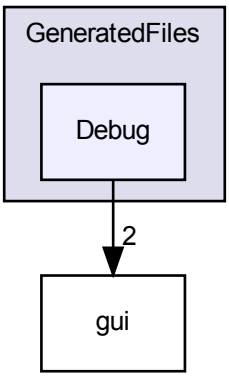
Files

- file Controller.h
- file detector.cpp

- file detector.h
- file framecapture.cpp
- file framecapture.h
- file framereceiver.h
- file recognizer.h
- file trainerimage.cpp
- file trainerimage.h
- file util.h

7.2 visual-control/GeneratedFiles/Debug/ Directory Reference

Directory dependency graph for visual-control/GeneratedFiles/Debug/:

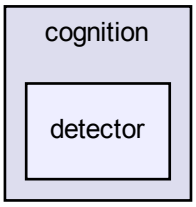


Files

- file moc_visualcontrol.cpp
- file moc_webcamwidget.cpp

7.3 visual-control/cognition/detector/ Directory Reference

Directory dependency graph for visual-control/cognition/detector/:

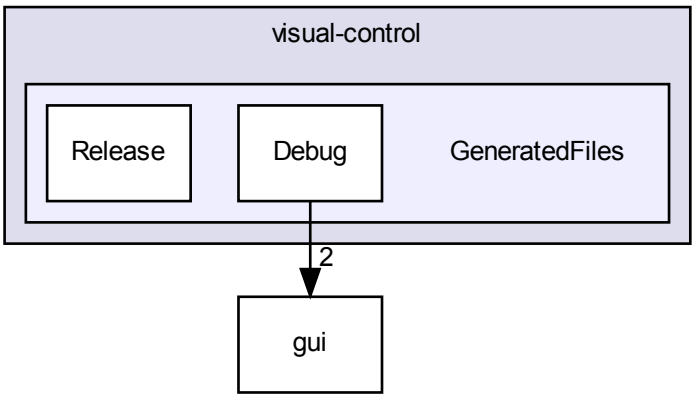


Files

- file detailedfacedetector.cpp
- file detailedfacedetector.h
- file facedetector.cpp
- file facedetector.h

7.4 visual-control/GeneratedFiles/ Directory Reference

Directory dependency graph for visual-control/GeneratedFiles/:



Directories

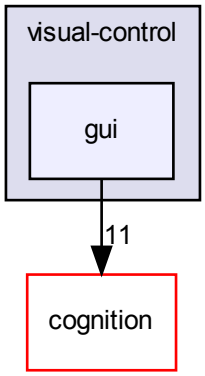
- directory Debug
- directory Release

Files

- file qrc_visualcontrol.cpp

7.5 visual-control/gui/ Directory Reference

Directory dependency graph for visual-control/gui/:

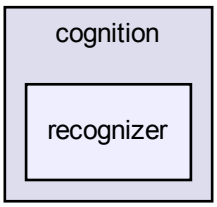


Files

- file visualcontrol.cpp
- file visualcontrol.h
- file webcamwidget.cpp
- file webcamwidget.h

7.6 visual-control/cognition/recognizer/ Directory Reference

Directory dependency graph for visual-control/cognition/recognizer/:

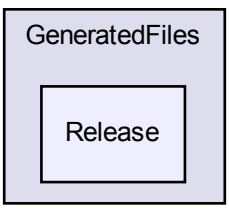


Files

- file eigenfacerecognizer.cpp
- file eigenfacerecognizer.h

7.7 visual-control/GeneratedFiles/Release/ Directory Reference

Directory dependency graph for visual-control/GeneratedFiles/Release/:

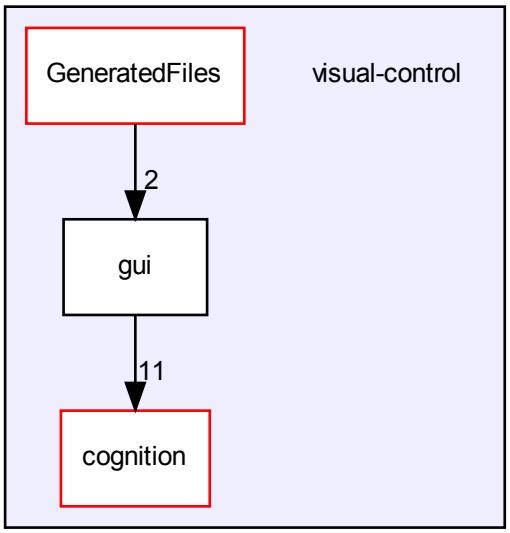


Files

- file moc_visualcontrol.cpp
- file moc_webcamwidget.cpp

7.8 visual-control/ Directory Reference

Directory dependency graph for visual-control/:



Directories

- directory cognition
- directory GeneratedFiles
- directory gui

Files

- file logger.cpp
- file logger.h
- file main.cpp

- file visualcontrol.cpp
- file visualcontrol.h
- file webcamwidget.cpp
- file webcamwidget.h

Chapter 8

Namespace Documentation

8.1 cognition Namespace Reference

Classes

- class Controller

  Interface for objects that want to receive status change notifications from detectors.

- class DetailedFaceDetector

  Detects faces and the corresponding eyes nose and mouth that are part of that face.

- class FaceDetector

  Basic detector that detects faces inside a frame.

- class Detector

  Base class for detectors. Provides threading system and thread safe methods for getting the most recent frame, notifying controllers and getting the detection result.

- class FrameCapture

  Captures frames from a capture device (cv::VideoCapture) on a certain framerate and sends the frames to all attached frame receivers.

- class FrameReceiver

  Interfaces for classes that receive frames.

- class EigenfaceRecognizer

  Implementation of the Eigenface face recognition algorithm it uses old style C functions of the OpenCV library, based on `http://www.cognotics.com/opencv/servo_2007_series/part_5/index.html` *by Robin Hewitt.*

- class Recognizer

  Base class for recognizers. Provides the interface for different kinds of recognizers.

- class TrainerImage

  Class used to prepare and store trainig images to be used in recognizer classes. This class basically resizes the image and transforms its color to grayscale. Input is assumed to be in BGR format. (captured from cv::VideoCapture).

8.2 cv Namespace Reference

8.3 gui Namespace Reference

Classes

- class VisualControl

  Main application window.

- class WebcamWidget

  Widget that displays the webcam feed, provided by a capture devices it also draws the rectangles it recieves from detectors!

Chapter 9

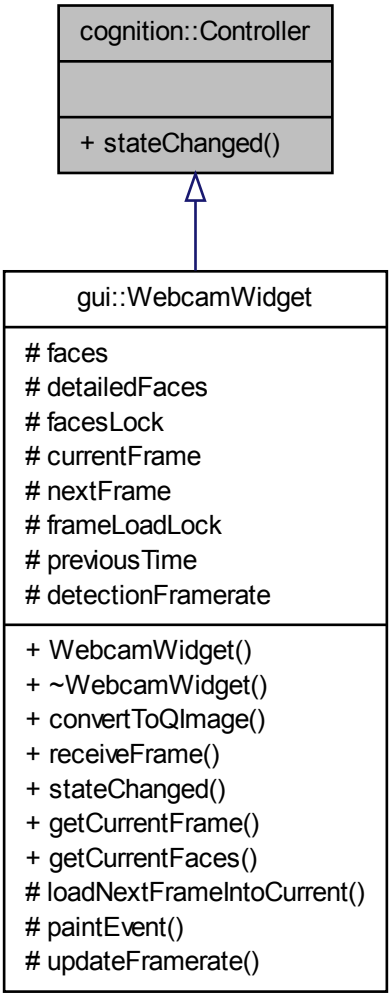
Class Documentation

9.1 cognition::Controller Class Reference

Interface for objects that want to receive status change notifications from detectors.

`#include <Controller.h>`

Inheritance diagram for cognition::Controller:



Public Member Functions

- virtual void stateChanged (Detector *detector)=0

  This method will be called when a detector has an update.

9.1.1 Detailed Description

Interface for objects that want to receive status change notifications from detectors. Subclass Controller and register the object to a detector. You will then receive updates form the detector as they are available.

Author

Christophe Hesters

Definition at line 17 of file Controller.h.

9.1.2 Member Function Documentation

9.1.2.1 virtual void cognition::Controller::stateChanged (Detector * *detector*) [pure virtual]

This method will be called when a detector has an update.

Parameters

detector	the detector that has an updated status

Implemented in gui::WebcamWidget.

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

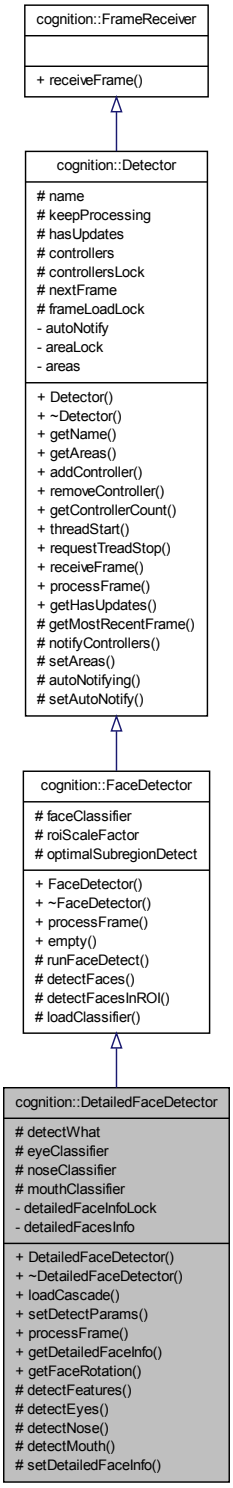
- visual-control/cognition/Controller.h

9.2 cognition::DetailedFaceDetector Class Reference

Detects faces and the corresponding eyes nose and mouth that are part of that face.

`#include <detailedfacedetector.h>`

Inheritance diagram for cognition::DetailedFaceDetector:



Classes

- struct less_cvrect

Public Types

- enum DetectParams {
  FACE = 0, LEFT_EYE = 1, RIGHT_EYE = 2, EYES = 3,
  NOSE = 4, EYES_AND_NOSE = 7, MOUTH = 8, EYES_AND_MOUTH = 11,
  NOSE_AND_MOUTH = 12, ALL = 15 }
- typedef std::map< DetectParams, cv::Rect > FaceDetails
- typedef std::map< cv::Rect, FaceDetails, less_cvrect > DetailedFaces

Public Member Functions

- DetailedFaceDetector (DetectParams detectWhat, const std::string &faceCascadePath, FrameCapture *captureDevice=0, bool optimalSubregionDetect=false, double roiScaleFactor=1.16, const std::string &name="detailed face detector")

  Initializes this object. If you wish to detect more features than just the face, load the approrpriate classifiers trough loadCascade before starting detection! otherwise you can expect a crash!

- virtual ~DetailedFaceDetector (void)
- bool loadCascade (DetectParams which, const std::string &cascadePath)

  Loads the specified internal classifier with the given cascade. this is required if these detection features are enabled. This has to be called before detection starts and it is NOT thread-safe!

- void setDetectParams (DetectParams featuresToDetect)

  configures this object to detect the specified features make sure all the corresponding classifiers are loaded before enabling this

- virtual void processFrame ()

  fetches the current frame and detects the selected features. this method is called by the client or the threading system.

- DetailedFaces getDetailedFaceInfo ()

  Returns most recent detection results (thread-safe)

Static Public Member Functions

- static float getFaceRotation (const FaceDetails &faceDetails)

  Searches for the eyes in the FaceDetails structure and returns the angle in degrees. (0 means the face is straight ahead) a negative value means a rotation left, positive means a rotation right.

Protected Member Functions

- void detectFeatures (const cv::Mat &frame)

  initiates detection of the selected features on all the detected faces

- void detectEyes (const cv::Mat &frame, const cv::Rect &faceRect, FaceDetails &results)

  Tries to detect eyes at the most probable location inside the frame and stores the result in a FaceDetails structure.

- void detectNose (const cv::Mat &frame, const cv::Rect &faceRect, FaceDetails &results)

  Tries to detect the nose at the most probable location inside the frame and stores the result in a FaceDetails structure.

- void detectMouth (const cv::Mat &frame, const cv::Rect &faceRect, FaceDetails &results)

  Tries to detect the mouth at the most probable location inside the frame and stores the result in a FaceDetails structure.

- void setDetailedFaceInfo (const DetailedFaces &facesDetails)

  Updates the latest detection result (thread-safe)

Protected Attributes

- DetectParams detectWhat
- cv::CascadeClassifier eyeClassifier
- cv::CascadeClassifier noseClassifier
- cv::CascadeClassifier mouthClassifier

Private Attributes

- boost::mutex detailedFaceInfoLock
- DetailedFaces detailedFacesInfo

9.2.1 Detailed Description

Detects faces and the corresponding eyes nose and mouth that are part of that face. Always tries to detect a face first. On succesfull detection it will try to detect the eyes, nose and mouth too. These options can turned on/off

Author

Christophe Hesters

Definition at line 19 of file detailedfacedetector.h.

9.2.2 Member Typedef Documentation

9.2.2.1 typedef std::map<cv::Rect, FaceDetails, less_cvrect> cognition::DetailedFaceDetector::DetailedFaces

Definition at line 49 of file detailedfacedetector.h.

9.2.2.2 typedef std::map<DetectParams, cv::Rect> cognition::DetailedFaceDetector::FaceDetails

Definition at line 46 of file detailedfacedetector.h.

9.2.3 Member Enumeration Documentation

9.2.3.1 enum cognition::DetailedFaceDetector::DetectParams

Enumerator:

FACE

LEFT_EYE

RIGHT_EYE

EYES

NOSE

EYES_AND_NOSE

MOUTH

EYES_AND_MOUTH

NOSE_AND_MOUTH

ALL

Definition at line 23 of file detailedfacedetector.h.

9.2.4 Constructor & Destructor Documentation

9.2.4.1 cognition::DetailedFaceDetector::DetailedFaceDetector (DetectParams *detectWhat,* const std::string & *faceCascadePath,* FrameCapture * *captureDevice* = `0`, bool *optimalSubregionDetect* = `false`, double *roiScaleFactor* = `1.16`, const std::string & *name* = `"detailed face detector"`)

Initializes this object. If you wish to detect more features than just the face, load the approrpriate classifiers trough loadCascade before starting detection! otherwise you can expect a crash!

Parameters

detectWhat	needed to specify what to detect, use loadCascade to load required classifiers
faceCascadePath	the path to the haar cascade xml file, loads the classifier!
captureDevice	if 0, nothing else it will register itself as a frame receiver
optimalSubregionDetect	if true, only scan area's where a face was detected
roiScaleFactor	how far beyond the former face rect should be scanned?
name	an name for this object

Definition at line 3 of file detailedfacedetector.cpp.

9.2.4.2 cognition::DetailedFaceDetector::~DetailedFaceDetector (void) [virtual]

Definition at line 19 of file detailedfacedetector.cpp.

9.2.5 Member Function Documentation

9.2.5.1 void cognition::DetailedFaceDetector::detectEyes (const cv::Mat & *frame,* const cv::Rect & *faceRect,* FaceDetails & *results*) [protected]

Tries to detect eyes at the most probable location inside the frame and stores the result in a FaceDetails structure.

Parameters

frame	the current complete frame(no subregion!)
faceRect	the rectangle of the current face
results	the structure to store the results in

Definition at line 86 of file detailedfacedetector.cpp.

9.2.5.2 void cognition::DetailedFaceDetector::detectFeatures (const cv::Mat & *frame*) [protected]

initiates detection of the selected features on all the detected faces

Parameters

frame	the current frame

Definition at line 57 of file detailedfacedetector.cpp.

9.2.5.3 void cognition::DetailedFaceDetector::detectMouth (const cv::Mat & *frame,* const cv::Rect & *faceRect,* FaceDetails & *results*) [protected]

Tries to detect the mouth at the most probable location inside the frame and stores the result in a FaceDetails structure.

Parameters

frame	the current complete frame(no subregion!)
faceRect	the rectangle of the current face
results	the structure to store the results in

Definition at line 181 of file detailedfacedetector.cpp.

9.2.5.4 void cognition::DetailedFaceDetector::detectNose (const cv::Mat & *frame,* const cv::Rect & *faceRect,* FaceDetails & *results*) [protected]

Tries to detect the nose at the most probable location inside the frame and stores the result in a FaceDetails structure.

Parameters

frame	the current complete frame(no subregion!)
faceRect	the rectangle of the current face
results	the structure to store the results in

Definition at line 135 of file detailedfacedetector.cpp.

9.2.5.5 DetailedFaceDetector::DetailedFaces cognition::DetailedFaceDetector::getDetailedFaceInfo ( )

Returns most recent detection results (thread-safe)

Returns

A structure which maps detected face rectangles to the other features of that face

See also

definition of DetailedFaces

Definition at line 234 of file detailedfacedetector.cpp.

9.2.5.6 float cognition::DetailedFaceDetector::getFaceRotation (const FaceDetails & *faceDetails*) [static]

Searches for the eyes in the FaceDetails structure and returns the angle in degrees. (0 means the face is straight ahead) a negative value means a rotation left, positive means a rotation right.

Parameters

faceDetails	The detailed features of a face

Returns

the angle that describes face rotation, negative is left, positive is right

Definition at line 241 of file detailedfacedetector.cpp.

9.2.5.7 bool cognition::DetailedFaceDetector::loadCascade (DetectParams *which,* const std::string & *cascadePath*)

Loads the specified internal classifier with the given cascade. this is required if these detection features are enabled. This has to be called before detection starts and it is NOT thread-safe!

Parameters

which	the classifier to load possibilities are EYES, NOSE and MOUTH
cascadePath	the path at which the xml haar cascade resides

Returns

true on success, false on failure

Definition at line 44 of file detailedfacedetector.cpp.

9.2.5.8 void cognition::DetailedFaceDetector::processFrame ( ) [virtual]

fetches the current frame and detects the selected features. this method is called by the client or the threading system.

Reimplemented from cognition::FaceDetector.

Definition at line 22 of file detailedfacedetector.cpp.

9.2.5.9 void cognition::DetailedFaceDetector::setDetailedFaceInfo (const DetailedFaces & *facesDetails*) [protected]

Updates the latest detection result (thread-safe)

Parameters

facesDetails	updates the latest detection result with facesDetails

Definition at line 227 of file detailedfacedetector.cpp.

9.2.5.10 void cognition::DetailedFaceDetector::setDetectParams (DetectParams *featuresToDetect*) [inline]

configures this object to detect the specified features make sure all the corresponding classifiers are loaded before enabling this

Parameters

featuresToDetect	configures what this object detects

Definition at line 91 of file detailedfacedetector.h.

9.2.6 Member Data Documentation

9.2.6.1 boost::mutex cognition::DetailedFaceDetector::detailedFaceInfoLock [private]

Definition at line 171 of file detailedfacedetector.h.

9.2.6.2 DetailedFaces cognition::DetailedFaceDetector::detailedFacesInfo [private]

Definition at line 174 of file detailedfacedetector.h.

9.2.6.3 DetectParams cognition::DetailedFaceDetector::detectWhat [protected]

Definition at line 119 of file detailedfacedetector.h.

9.2.6.4 cv::CascadeClassifier cognition::DetailedFaceDetector::eyeClassifier [protected]

Definition at line 121 of file detailedfacedetector.h.

9.2.6.5 cv::CascadeClassifier cognition::DetailedFaceDetector::mouthClassifier [protected]

Definition at line 123 of file detailedfacedetector.h.

9.2.6.6 cv::CascadeClassifier cognition::DetailedFaceDetector::noseClassifier [protected]

Definition at line 122 of file detailedfacedetector.h.

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

- visual-control/cognition/detector/detailedfacedetector.h

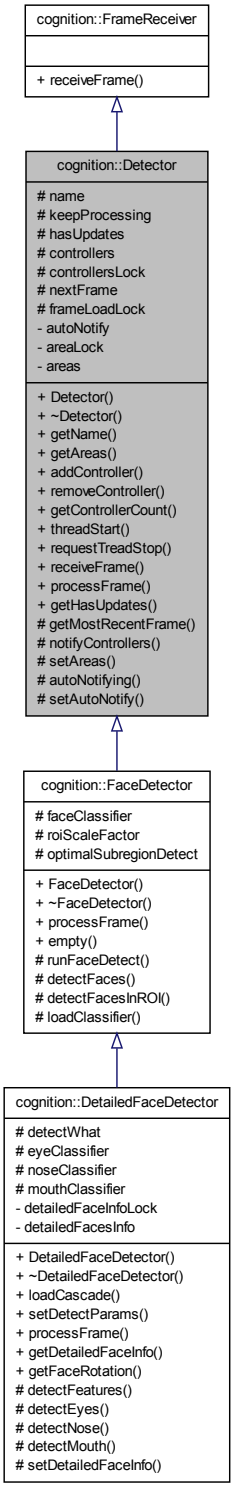
- visual-control/cognition/detector/detailedfacedetector.cpp

9.3 cognition::Detector Class Reference

Base class for detectors. Provides threading system and thread safe methods for getting the most recent frame, notifying controllers and getting the detection result.

`#include <detector.h>`

Inheritance diagram for cognition::Detector:



Public Types

- typedef std::vector< cv::Rect > RectVector
- typedef RectVector::iterator RectVectorItr
- typedef FrameCapture * FrameCapturePtr
- typedef Controller * ControllerPtr

Public Member Functions

- Detector (const std::string &name, FrameCapturePtr captureDevice=0)

  Constructs this object with a name and the selected capture device.

- virtual ~Detector (void)
- std::string getName () const

  Returns the name of the detector.

- RectVector getAreas ()

  Returns the latest detection result (thread-safe)

- bool addController (ControllerPtr controller)

  Adds the controller to the internal set. It will receive updated status notifications (thread-safe)

- void removeController (ControllerPtr controller)

  Removes the controller to the internal set. (thread-safe)

- unsigned getControllerCount ()

  Gets the internal controller count.

- void threadStart ()

  Starts the internal processing thread. Make sure this method is called from a new thread as it will be trapped inside a loop until requestThreadStop is called or an internal error causes it to stop. The internal loop calls processFrame!

- void requestTreadStop ()

  Signals the internal processing thread to stop. It sets the internal keepProcessing flag to false, which should terminate the internal processing loop in a clean way.

- virtual void receiveFrame (const cv::Mat &frame)

  Updates the internal latest frame for processing.(thread-safe)

- virtual void processFrame ()=0

  Template method for implementors. They should call getMostRecentFrame , check if the returned frame it is not empty and process it.

- bool getHasUpdates ()

  Returns true if there are updates.

Protected Types

- typedef std::set< ControllerPtr > ControllerSet
- typedef ControllerSet::iterator ControllerSetItr

Protected Member Functions

- cv::Mat getMostRecentFrame ()

  Retrieves the most recent frame.

- void notifyControllers ()

  Notifies all the attached controllers about updates.

- bool setAreas (const RectVector &areas)

  Updates the internal detection areas, and checks if they are really different.

- bool autoNotifying ()

  Checks if the detector automatically notifies on setAreas.

- void setAutoNotify (bool doAutoNotify)

  Sets an internal flag that causes setAreas to send attached controllers notifications automatically.

Protected Attributes

- std::string name
- bool keepProcessing
- volatile bool hasUpdates
- ControllerSet controllers
- boost::mutex controllersLock
- cv::Mat nextFrame
- boost::mutex frameLoadLock

Private Attributes

- bool autoNotify
- boost::mutex areaLock
- RectVector areas

9.3.1 Detailed Description

Base class for detectors. Provides threading system and thread safe methods for getting the most recent frame, notifying controllers and getting the detection result.

Author

Christophe Hesters

Definition at line 24 of file detector.h.

9.3.2 Member Typedef Documentation

9.3.2.1 typedef Controller* cognition::Detector::ControllerPtr

Definition at line 35 of file detector.h.

9.3.2.2 typedef std::set< ControllerPtr > cognition::Detector::ControllerSet [protected]

Definition at line 127 of file detector.h.

9.3.2.3 typedef ControllerSet::iterator cognition::Detector::ControllerSetItr [protected]

Definition at line 128 of file detector.h.

9.3.2.4 typedef FrameCapture* cognition::Detector::FrameCapturePtr

Definition at line 32 of file detector.h.

9.3.2.5 typedef std::vector<cv::Rect> cognition::Detector::RectVector

Definition at line 28 of file detector.h.

9.3.2.6 typedef RectVector::iterator cognition::Detector::RectVectorItr

Definition at line 29 of file detector.h.

9.3.3 Constructor & Destructor Documentation

9.3.3.1 cognition::Detector::Detector (const std::string & *name,* FrameCapturePtr *captureDevice* = 0)

Constructs this object with a name and the selected capture device.

Parameters

name	Description of parameter name.
captureDevice	Description of parameter captureDevice.

Definition at line 7 of file detector.cpp.

9.3.3.2 cognition::Detector::~Detector (void) [virtual]

Definition at line 16 of file detector.cpp.

9.3.4 Member Function Documentation

9.3.4.1 bool cognition::Detector::addController (ControllerPtr *controller*)

Adds the controller to the internal set. It will receive updated status notifications (thread-safe)

Parameters

controller	pointer to the controller to receive notifications of this object

Returns

true if added, false on failure or if the controller was already in the set

Definition at line 21 of file detector.cpp.

9.3.4.2 bool cognition::Detector::autoNotifying ( ) [inline, protected]

Checks if the detector automatically notifies on setAreas.

Returns

true if it automatically notifies on setAreas, false otherwise

Definition at line 163 of file detector.h.

9.3.4.3 Detector::RectVector cognition::Detector::getAreas ( )

Returns the latest detection result (thread-safe)

Returns

the rectangles wherin the detected objects reside

Definition at line 70 of file detector.cpp.

9.3.4.4 unsigned cognition::Detector::getControllerCount ( ) [inline]

Gets the internal controller count.

Returns

the number of controllers registered

Definition at line 80 of file detector.h.

9.3.4.5 bool cognition::Detector::getHasUpdates ( ) [inline]

Returns true if there are updates.

Returns

true if the detector had updates

Definition at line 115 of file detector.h.

9.3.4.6 cv::Mat cognition::Detector::getMostRecentFrame ( ) [protected]

Retrieves the most recent frame.

Returns

a copy of the most recent frame

Definition at line 107 of file detector.cpp.

9.3.4.7 std::string cognition::Detector::getName ( ) const [inline]

Returns the name of the detector.

Returns

the name

Definition at line 51 of file detector.h.

9.3.4.8 void cognition::Detector::notifyControllers ( ) [protected]

Notifies all the attached controllers about updates.

Definition at line 43 of file detector.cpp.

9.3.4.9 virtual void cognition::Detector::processFrame ( ) [pure virtual]

Template method for implementors. They should call getMostRecentFrame , check if the returned frame it is not empty and process it.

Implemented in cognition::DetailedFaceDetector, and cognition::FaceDetector.

9.3.4.10 void cognition::Detector::receiveFrame (const cv::Mat & *frame*) [virtual]

Updates the internal latest frame for processing.(thread-safe)

Parameters

frame	the latest frame captured.

Implements cognition::FrameReceiver.

Definition at line 98 of file detector.cpp.

9.3.4.11 void cognition::Detector::removeController (ControllerPtr *controller*)

Removes the controller to the internal set. (thread-safe)

Parameters

controller	pointer to the controller that needs to be removed

Definition at line 27 of file detector.cpp.

9.3.4.12 void cognition::Detector::requestTreadStop ( )

Signals the internal processing thread to stop. It sets the internal keepProcessing flag to false, which should terminate the internal processing loop in a clean way.

Definition at line 65 of file detector.cpp.

9.3.4.13 bool cognition::Detector::setAreas (const RectVector & *areas*) [protected]

Updates the internal detection areas, and checks if they are really different.

Parameters

areas	the new rectangle vector where objects are detected

Returns

true if areas was indeed different than the previous state, false if there wasn't a real change

Definition at line 78 of file detector.cpp.

9.3.4.14 void cognition::Detector::setAutoNotify (bool *doAutoNotify*) [inline, protected]

Sets an internal flag that causes setAreas to send attached controllers notifications automatically.

Parameters

doAutoNotify	true if it needs to send updates automatically, false if it wants to this automatically

Definition at line 172 of file detector.h.

9.3.4.15 void cognition::Detector::threadStart ( )

Starts the internal processing thread. Make sure this method is called from a new thread as it will be trapped inside a loop until requestThreadStop is called or an internal error causes it to stop. The internal loop calls processFrame!

Definition at line 51 of file detector.cpp.

9.3.5 Member Data Documentation

9.3.5.1 boost::mutex cognition::Detector::areaLock [private]

Definition at line 178 of file detector.h.

9.3.5.2 RectVector cognition::Detector::areas [private]

Definition at line 182 of file detector.h.

9.3.5.3 bool cognition::Detector::autoNotify [private]

Definition at line 175 of file detector.h.

9.3.5.4 ControllerSet cognition::Detector::controllers [protected]

Definition at line 130 of file detector.h.

9.3.5.5 boost::mutex cognition::Detector::controllersLock [protected]

Definition at line 131 of file detector.h.

9.3.5.6 boost::mutex cognition::Detector::frameLoadLock [protected]

Definition at line 135 of file detector.h.

9.3.5.7 volatile bool cognition::Detector::hasUpdates [protected]

Definition at line 124 of file detector.h.

9.3.5.8 bool cognition::Detector::keepProcessing [protected]

Definition at line 121 of file detector.h.

9.3.5.9 std::string cognition::Detector::name [protected]

Definition at line 118 of file detector.h.

9.3.5.10 cv::Mat cognition::Detector::nextFrame [protected]

Definition at line 134 of file detector.h.

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

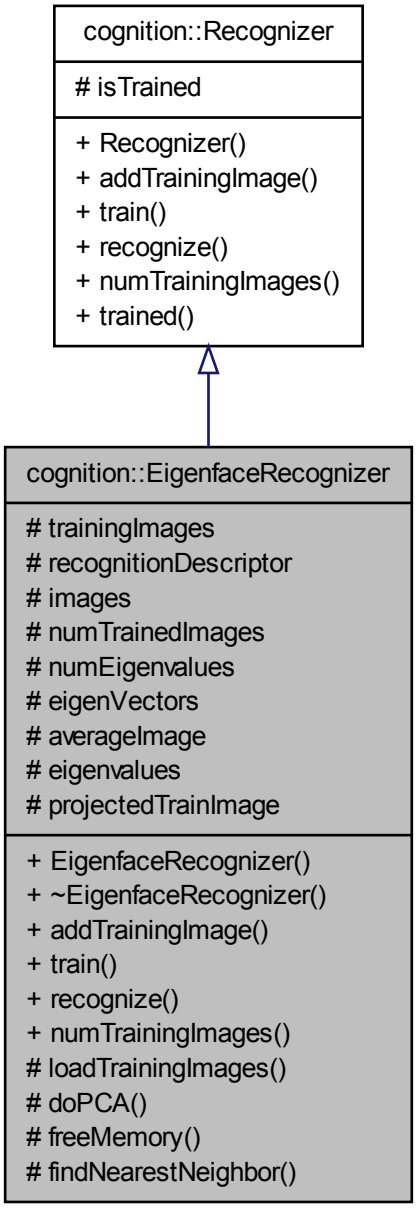
- visual-control/cognition/detector.h
- visual-control/cognition/detector.cpp

9.4 cognition::EigenfaceRecognizer Class Reference

Implementation of the Eigenface face recognition algorithm it uses old style C functions of the OpenCV library, based on http://www.cognotics.com/opencv/servo_-2007_series/part_5/index.html by Robin Hewitt.

`#include <eigenfacerecognizer.h>`

Inheritance diagram for cognition::EigenfaceRecognizer:



Public Member Functions

- EigenfaceRecognizer (void)
- virtual ~EigenfaceRecognizer (void)
- bool addTrainingImage (const std::string &filename, const std::string &name)

  Adds a training image path to the training set of known images after you have added 2 or more images, call train to learn and set yourself up for recognition. All images must be the same size!

- bool train ()

  starts the learning process on all the known images that are added trough addTrainingImage. You can add more training images after training, but you have to call train again. While training you cannot use recognize()!

- std::string recognize (cv::Mat &face)

  does recognition on the face, and returns the most likely match. This face must grayscale and be exactly the same size as the training images.

- std::size_t numTrainingImages ()

  does recognition on the face, and returns the most likely match. This face must grayscale and be exactly the same size as the training images.

Protected Types

- typedef std::map< std::string, std::string > StringMap
- typedef std::pair< std::string, std::string > StringPair

Protected Member Functions

- bool loadTrainingImages ()
- void doPCA ()
- void freeMemory (int fromIndex=-1)
- int findNearestNeighbor (float *projectedTestImage)

Protected Attributes

- StringMap trainingImages
- std::vector< StringPair > recognitionDescriptor
- IplImage ** images
- int numTrainedImages
- int numEigenvalues
- IplImage ** eigenVectors
- IplImage * averageImage
- CvMat * eigenvalues
- CvMat * projectedTrainImage

9.4.1 Detailed Description

Implementation of the Eigenface face recognition algorithm it uses old style C functions of the OpenCV library, based on http://www.cognotics.com/opencv/servo_2007_series/part_5/index.html by Robin Hewitt.

Todo

upgrade internals to the OpenCV C++ api (cv::Mat and cv::PCA)

Author

Christophe Hesters 29-1-2011

Definition at line 27 of file eigenfacerecognizer.h.

9.4.2 Member Typedef Documentation

9.4.2.1 typedef std::map<std::string, std::string> cognition::EigenfaceRecognizer::StringMap [protected]

Definition at line 86 of file eigenfacerecognizer.h.

9.4.2.2 typedef std::pair<std::string, std::string> cognition::EigenfaceRecognizer::StringPair [protected]

Definition at line 92 of file eigenfacerecognizer.h.

9.4.3 Constructor & Destructor Documentation

9.4.3.1 cognition::EigenfaceRecognizer::EigenfaceRecognizer (void)

Definition at line 4 of file eigenfacerecognizer.cpp.

9.4.3.2 cognition::EigenfaceRecognizer::~EigenfaceRecognizer (void) [virtual]

Definition at line 16 of file eigenfacerecognizer.cpp.

9.4.4 Member Function Documentation

9.4.4.1 bool cognition::EigenfaceRecognizer::addTrainingImage (const std::string & *filename*, const std::string & *name*) [virtual]

Adds a training image path to the training set of known images after you have added 2 or more images, call train to learn and set yourself up for recognition. All images must be the same size!

Note

warning! method does not check if the filename exists and the application can crash if it does not exist! this check should be added (in train?)!

Parameters

filename	the path where to find the image (all of the same size!)
name	the name you want to attach to the image

Returns

bool true if the path is added succesfully

Implements cognition::Recognizer.

Definition at line 21 of file eigenfacerecognizer.cpp.

9.4.4.2 void cognition::EigenfaceRecognizer::doPCA ( ) [protected]

Definition at line 115 of file eigenfacerecognizer.cpp.

9.4.4.3 int cognition::EigenfaceRecognizer::findNearestNeighbor (float * *projectedTestImage*) [protected]

Definition at line 185 of file eigenfacerecognizer.cpp.

9.4.4.4 void cognition::EigenfaceRecognizer::freeMemory (int *fromIndex* = -1) [protected]

Definition at line 27 of file eigenfacerecognizer.cpp.

9.4.4.5 bool cognition::EigenfaceRecognizer::loadTrainingImages ( ) [protected]

Definition at line 71 of file eigenfacerecognizer.cpp.

9.4.4.6 std::size_t cognition::EigenfaceRecognizer::numTrainingImages ( ) [inline, virtual]

does recognition on the face, and returns the most likely match. This face must grayscale and be exactly the same size as the training images.

Returns

size_t the number of registered training images

Implements cognition::Recognizer.

Definition at line 76 of file eigenfacerecognizer.h.

9.4.4.7 std::string cognition::EigenfaceRecognizer::recognize (cv::Mat & *face*) [virtual]

does recognition on the face, and returns the most likely match. This face must grayscale and be exactly the same size as the training images.

Parameters

face	the matrix containing the face

Returns

string name of closest match in the set of training images

Implements cognition::Recognizer.

Definition at line 158 of file eigenfacerecognizer.cpp.

9.4.4.8 bool cognition::EigenfaceRecognizer::train ( ) [virtual]

starts the learning process on all the known images that are added trough addTrainingImage. You can add more training images after training, but you have to call train again. While training you cannot use recognize()!

Note

make sure all image paths exist

Returns

bool true if trained, false otherwise

Implements cognition::Recognizer.

Definition at line 37 of file eigenfacerecognizer.cpp.

9.4.5 Member Data Documentation

9.4.5.1 IplImage* cognition::EigenfaceRecognizer::averageImage [protected]

Definition at line 99 of file eigenfacerecognizer.h.

9.4.5.2 CvMat* cognition::EigenfaceRecognizer::eigenvalues [protected]

Definition at line 100 of file eigenfacerecognizer.h.

9.4.5.3 IplImage** cognition::EigenfaceRecognizer::eigenVectors [protected]

Definition at line 98 of file eigenfacerecognizer.h.

9.4.5.4 IplImage** cognition::EigenfaceRecognizer::images [protected]

Definition at line 95 of file eigenfacerecognizer.h.

9.4.5.5 int cognition::EigenfaceRecognizer::numEigenvalues [protected]

Definition at line 97 of file eigenfacerecognizer.h.

9.4.5.6 int cognition::EigenfaceRecognizer::numTrainedImages [protected]

Definition at line 96 of file eigenfacerecognizer.h.

9.4.5.7 CvMat* cognition::EigenfaceRecognizer::projectedTrainImage [protected]

Definition at line 101 of file eigenfacerecognizer.h.

9.4.5.8 std::vector<StringPair> cognition::EigenfaceRecognizer::recognitionDescriptor [protected]

Definition at line 93 of file eigenfacerecognizer.h.

9.4.5.9 StringMap cognition::EigenfaceRecognizer::trainingImages [protected]

Definition at line 89 of file eigenfacerecognizer.h.

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

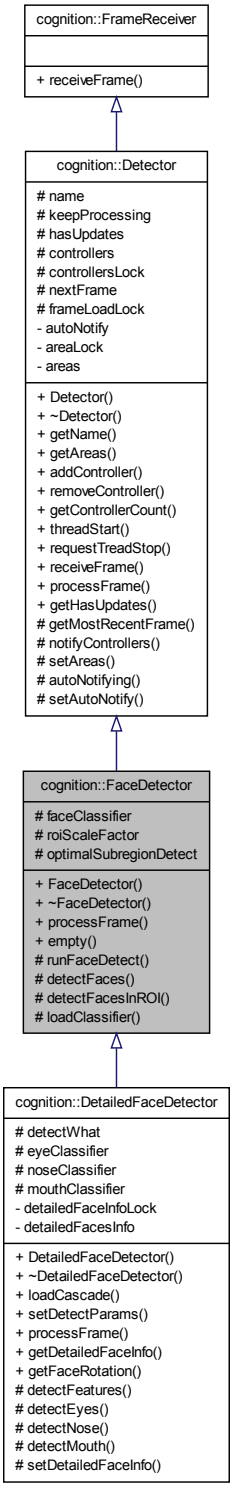
- visual-control/cognition/recognizer/eigenfacerecognizer.h
- visual-control/cognition/recognizer/eigenfacerecognizer.cpp

9.5 cognition::FaceDetector Class Reference

Basic detector that detects faces inside a frame.

`#include <facedetector.h>`

Inheritance diagram for cognition::FaceDetector:



Public Member Functions

- FaceDetector (const std::string &faceCascadePath, FrameCapture *captureDevice=0, bool optimalSubregionDetect=false, double roiScaleFactor=1.16, const std::string &name="face detector")

  Constructs the Face Detector.

- virtual ~FaceDetector (void)
- virtual void processFrame ()

  Starts the object detection on the latest frame.

- bool empty () const

  Checks if the internal classifier is empty.

Protected Member Functions

- void runFaceDetect (const cv::Mat &frame, RectVector &results)

  Runs the face detection classifier.

- void detectFaces (const cv::Mat &frame)

  Detects faces inside the frame. If there were previous results and optimalSubregion-Detect is on, it will automatically call detectFacesInRoi for more optimal subregion detection.

- void detectFacesInROI (RectVector &lastRects, const cv::Mat &frame)

  Detects faces inside subregions of a frame using the previous results.

- bool loadClassifier (const std::string &faceCascadePath)

  loads the internal classifier. Now protected because it is not thread-safe

Protected Attributes

- cv::CascadeClassifier faceClassifier
- double roiScaleFactor
- bool optimalSubregionDetect

9.5.1 Detailed Description

Basic detector that detects faces inside a frame.

Author

Christophe Hesters

Definition at line 14 of file facedetector.h.

9.5.2 Constructor & Destructor Documentation

9.5.2.1 cognition::FaceDetector::FaceDetector (const std::string & *faceCascadePath*, FrameCapture * *captureDevice* = 0, bool *optimalSubregionDetect* = false, double *roiScaleFactor* = 1.16, const std::string & *name* = "face detector")

Constructs the Face Detector.

Parameters

faceCascadePath	the path to the haar cascade xml file, loads the classifier!
captureDevice	if 0, nothing else it will register itself as a frame receiver
optimalSubregionDetect	if true, only scan area's where a face was detected
roiScaleFactor	how far beyond the former face rect should be scanned?
name	an name for this object

Author

Christophe hesters

Definition at line 8 of file facedetector.cpp.

9.5.2.2 cognition::FaceDetector::~FaceDetector (void) [virtual]

Definition at line 20 of file facedetector.cpp.

9.5.3 Member Function Documentation

9.5.3.1 void cognition::FaceDetector::detectFaces (const cv::Mat & *frame*) [protected]

Detects faces inside the frame. If there were previous results and optimalSubregionDetect is on, it will automatically call detectFacesInRoi for more optimal subregion detection.

Parameters

frame	the frame to detect faces in

Definition at line 34 of file facedetector.cpp.

9.5.3.2 void cognition::FaceDetector::detectFacesInROI (RectVector & *lastRects,* const cv::Mat & *frame*) [protected]

Detects faces inside subregions of a frame using the previous results.

Parameters

lastRects	the last detection results
frame	the current frame

Definition at line 75 of file facedetector.cpp.

9.5.3.3 bool cognition::FaceDetector::empty ( ) const [inline]

Checks if the internal classifier is empty.

Returns

true if the internal faceclassifier is empty.

Definition at line 45 of file facedetector.h.

9.5.3.4 bool cognition::FaceDetector::loadClassifier (const std::string & *faceCascadePath*) [inline, protected]

loads the internal classifier. Now protected because it is not thread-safe

Parameters

faceCascadePath	the path to the face cascade xml file

Returns

true on success, false otherwise

Definition at line 87 of file facedetector.h.

9.5.3.5 void cognition::FaceDetector::processFrame ( ) [virtual]

Starts the object detection on the latest frame.

Implements cognition::Detector.

Reimplemented in cognition::DetailedFaceDetector.

Definition at line 24 of file facedetector.cpp.

9.5.3.6 void cognition::FaceDetector::runFaceDetect (const cv::Mat & *frame,* RectVector & *results*) [inline, protected]

Runs the face detection classifier.

Parameters

frame	the frame to detect faces in
results	the vector to store the result rectangles in

Definition at line 121 of file facedetector.cpp.

9.5.4 Member Data Documentation

9.5.4.1 cv::CascadeClassifier cognition::FaceDetector::faceClassifier [protected]

Definition at line 50 of file facedetector.h.

9.5.4.2 bool cognition::FaceDetector::optimalSubregionDetect [protected]

Definition at line 53 of file facedetector.h.

9.5.4.3 double cognition::FaceDetector::roiScaleFactor [protected]

Definition at line 52 of file facedetector.h.

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

- visual-control/cognition/detector/facedetector.h
- visual-control/cognition/detector/facedetector.cpp

9.6 cognition::FrameCapture Class Reference

Captures frames from a capture device (cv::VideoCapture) on a certain framerate and sends the frames to all attached frame receivers.

`#include <framecapture.h>`

Public Types

- typedef FrameReceiver * FrameReceiverPtr

Public Member Functions

- FrameCapture (float framerate=24.0f)

Constructs the object, automatically creates a captureDevice.

- FrameCapture (CaptureDevicePtr existingCaptureDevice, float framerate=24.0f)

  Constructs the object with an existing captureDevices.

- ~FrameCapture ()

  Closes the captureDevice, even if it the device already existed!

- CaptureDevicePtr getCaptureDevice ()

  Returns a pointer to the internal capture device.

- void startCapturing ()

  Starts capturing frames. This will be a loop that captures frames and sends them to frame receivers. This method should be called by a thread that is free and designed for this purpose.

- void stopCapturing ()

  Signals the capturing loop to stop in a clean way.

- bool addFrameReceiver (FrameReceiverPtr receiver)

  Adds an object that wants to receive frames (thread-safe)

- void removeFrameReceiver (FrameReceiverPtr receiver)

  Removes an object that received frames (thread-safe)

- float getFramerate ()

  Returns the number of frames per second this object sends to its receivers.

- void setFramerate (float framerate)

  Sets the framerate at which this object sends frames to its receivers.

Protected Types

- typedef std::set< FrameReceiverPtr > ReceiverSet
- typedef ReceiverSet::iterator ReceiverSetItr

Protected Member Functions

- void notifyReceivers (cv::Mat &frame)

  Notifies the frame receivers with the new frame (thread-safe)

- void captureLoop ()

  Captures frames and notifies receivers.

Protected Attributes

- CaptureDevicePtr captureDevice
- unsigned long framerateWaitMs
- bool keepCapturing
- ReceiverSet receivers
- boost::mutex receiversLock

Private Types

- typedef cv::VideoCapture CaptureDevice
- typedef CaptureDevice * CaptureDevicePtr

9.6.1 Detailed Description

Captures frames from a capture device (cv::VideoCapture) on a certain framerate and sends the frames to all attached frame receivers.

Author

Christophe hesters

Definition at line 21 of file framecapture.h.

9.6.2 Member Typedef Documentation

9.6.2.1 typedef cv::VideoCapture cognition::FrameCapture::CaptureDevice [private]

Definition at line 23 of file framecapture.h.

9.6.2.2 typedef CaptureDevice* cognition::FrameCapture::CaptureDevicePtr [private]

Definition at line 25 of file framecapture.h.

9.6.2.3 typedef FrameReceiver* cognition::FrameCapture::FrameReceiverPtr

Definition at line 30 of file framecapture.h.

9.6.2.4 typedef std::set<FrameReceiverPtr > cognition::FrameCapture::ReceiverSet [protected]

Definition at line 106 of file framecapture.h.

9.6.2.5 typedef ReceiverSet::iterator cognition::FrameCapture::ReceiverSetItr [protected]

Definition at line 107 of file framecapture.h.

9.6.3 Constructor & Destructor Documentation

9.6.3.1 cognition::FrameCapture::FrameCapture (float *framerate* = 24.0f)

Constructs the object, automatically creates a captureDevice.

Parameters

framerate	the framerate at which it sends updates to the receivers

Definition at line 9 of file framecapture.cpp.

9.6.3.2 cognition::FrameCapture::FrameCapture (CaptureDevicePtr *existingCaptureDevice,* float *framerate* = 24.0f)

Constructs the object with an existing captureDevices.

Parameters

existingCaptureDevice	a valid CaptureDevice (warning: destructor will release the feed!)
framerate	the framerate at which it sends updates to the receivers

Definition at line 15 of file framecapture.cpp.

9.6.3.3 cognition::FrameCapture::~FrameCapture ( )

Closes the captureDevice, even if it the device already existed!

Definition at line 24 of file framecapture.cpp.

9.6.4 Member Function Documentation

9.6.4.1 bool cognition::FrameCapture::addFrameReceiver (FrameReceiverPtr *receiver*)

Adds an object that wants to receive frames (thread-safe)

Parameters

receiver	an object that will get frames from this object

Returns

true if it was added. false if it already was registered or if it failed

Definition at line 63 of file framecapture.cpp.

9.6.4.2 void cognition::FrameCapture::captureLoop ( ) [protected]

Captures frames and notifies receivers.

Definition at line 42 of file framecapture.cpp.

9.6.4.3 CaptureDevicePtr cognition::FrameCapture::getCaptureDevice ( ) [inline]

Returns a pointer to the internal capture device.

Returns

pointer to the interal capture devices

Definition at line 56 of file framecapture.h.

9.6.4.4 float cognition::FrameCapture::getFramerate ( ) [inline]

Returns the number of frames per second this object sends to its receivers.

Returns

number of frames per second

Definition at line 91 of file framecapture.h.

9.6.4.5 void cognition::FrameCapture::notifyReceivers (cv::Mat & *frame*) [protected]

Notifies the frame receivers with the new frame (thread-safe)

Parameters

frame	the frame to send to the frame receivers

Definition at line 55 of file framecapture.cpp.

9.6.4.6 void cognition::FrameCapture::removeFrameReceiver (FrameReceiverPtr *receiver*)

Removes an object that received frames (thread-safe)

Parameters

receiver	the object to remove

Definition at line 70 of file framecapture.cpp.

9.6.4.7 void cognition::FrameCapture::setFramerate (float *framerate*) [inline]

Sets the framerate at which this object sends frames to its receivers.

Parameters

framerate	the number of frames per second

Definition at line 98 of file framecapture.h.

9.6.4.8 void cognition::FrameCapture::startCapturing ( )

Starts capturing frames. This will be a loop that captures frames and sends them to frame receivers. This method should be called by a thread that is free and designed for this purpose.

Definition at line 30 of file framecapture.cpp.

9.6.4.9 void cognition::FrameCapture::stopCapturing ( )

Signals the capturing loop to stop in a clean way.

Definition at line 37 of file framecapture.cpp.

9.6.5 Member Data Documentation

9.6.5.1 CaptureDevicePtr cognition::FrameCapture::captureDevice [protected]

Definition at line 102 of file framecapture.h.

9.6.5.2 unsigned long cognition::FrameCapture::framerateWaitMs [protected]

Definition at line 103 of file framecapture.h.

9.6.5.3 bool cognition::FrameCapture::keepCapturing [protected]

Definition at line 104 of file framecapture.h.

9.6.5.4 ReceiverSet cognition::FrameCapture::receivers [protected]

Definition at line 109 of file framecapture.h.

9.6.5.5 boost::mutex cognition::FrameCapture::receiversLock [protected]

Definition at line 110 of file framecapture.h.

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

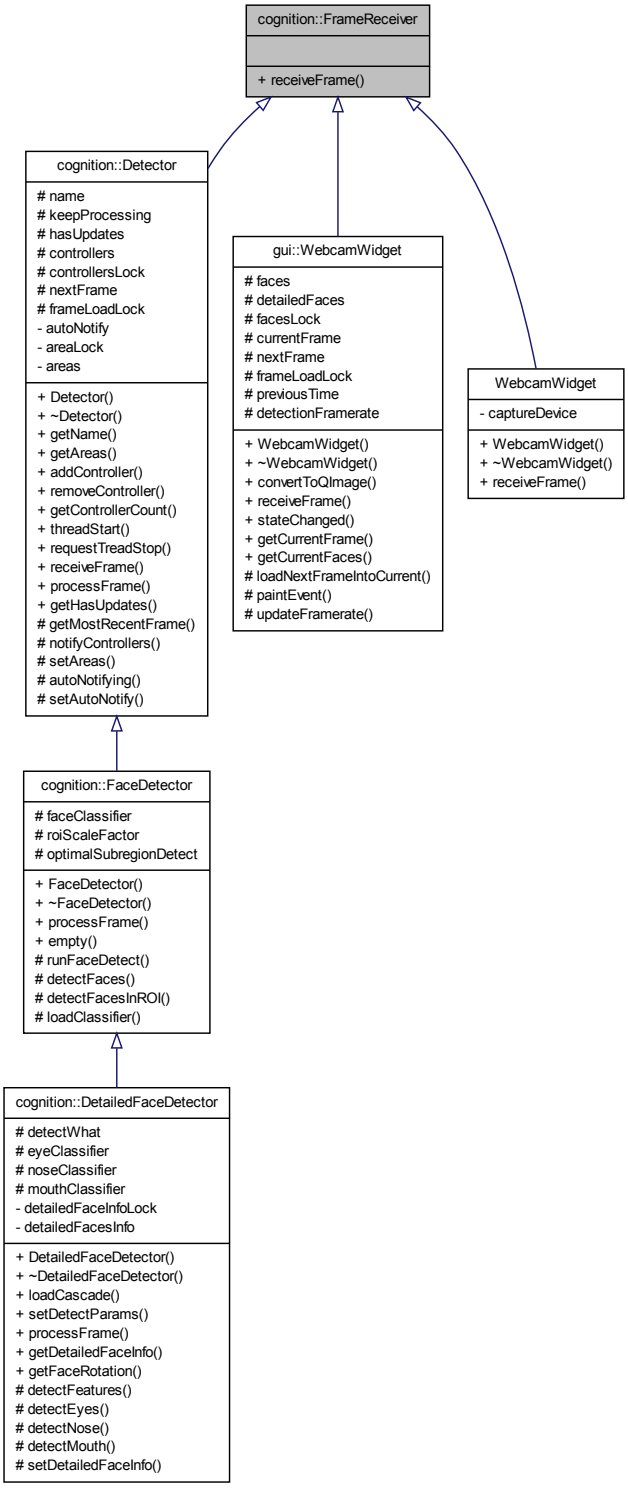
- visual-control/cognition/framecapture.h
- visual-control/cognition/framecapture.cpp

9.7 cognition::FrameReceiver Class Reference

Interfaces for classes that receive frames.

`#include <framereceiver.h>`

Inheritance diagram for cognition::FrameReceiver:



Public Member Functions

- virtual void receiveFrame (const cv::Mat &frame)=0

  implementors receive frames trough this method from FrameCapture

9.7.1 Detailed Description

Interfaces for classes that receive frames.

Author

Christophe Hesters

Definition at line 14 of file framereceiver.h.

9.7.2 Member Function Documentation

9.7.2.1 virtual void cognition::FrameReceiver::receiveFrame (const cv::Mat & *frame*) [pure virtual]

implementors receive frames trough this method from FrameCapture

Parameters

frame	the new frame

Implemented in cognition::Detector, gui::WebcamWidget, and WebcamWidget.

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

- visual-control/cognition/framereceiver.h

9.8 cognition::DetailedFaceDetector::less_cvrect Struct Reference

`#include <detailedfacedetector.h>`

Public Member Functions

- bool operator() (const cv::Rect &x, const cv::Rect &y) const

9.8.1 Detailed Description

Definition at line 38 of file detailedfacedetector.h.

9.8.2 Member Function Documentation

9.8.2.1 bool cognition::DetailedFaceDetector::less_cvrect::operator() (const cv::Rect & *x*, const cv::Rect & *y*) const [inline]

Definition at line 39 of file detailedfacedetector.h.

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

- visual-control/cognition/detector/detailedfacedetector.h

9.9 Logger Class Reference

`#include <logger.h>`

Public Member Functions

- void log (const std::string &msg)
- void setLogWidget (QListWidget *logWidget)

Static Public Member Functions

- static Logger & getInstance ()

Protected Member Functions

- Logger ()

Private Attributes

- QListWidget * logWidget

Static Private Attributes

- static Logger instance

9.9.1 Detailed Description

Definition at line 7 of file logger.h.

9.9.2 Constructor & Destructor Documentation

9.9.2.1 Logger::Logger (void) [protected]

Definition at line 6 of file logger.cpp.

9.9.3 Member Function Documentation

9.9.3.1 Logger & Logger::getInstance ( ) [static]

Definition at line 18 of file logger.cpp.

9.9.3.2 void Logger::log (const std::string & *msg*)

Definition at line 12 of file logger.cpp.

9.9.3.3 void Logger::setLogWidget (QListWidget * *logWidget*)

Definition at line 23 of file logger.cpp.

9.9.4 Member Data Documentation

9.9.4.1 Logger Logger::instance [static, private]

Definition at line 23 of file logger.h.

9.9.4.2 QListWidget* Logger::logWidget [private]

Definition at line 22 of file logger.h.

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

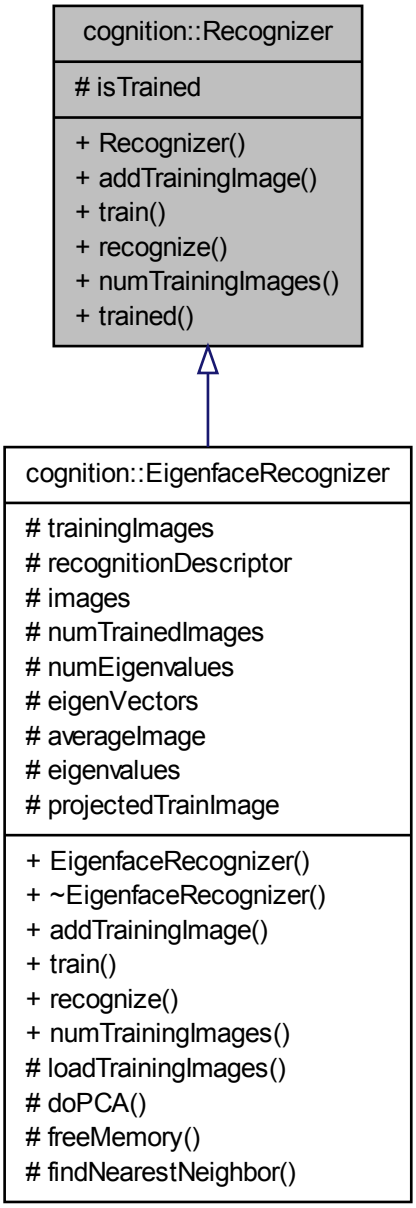
- visual-control/logger.h
- visual-control/logger.cpp

9.10 cognition::Recognizer Class Reference

Base class for recognizers. Provides the interface for different kinds of recognizers.

`#include <recognizer.h>`

Inheritance diagram for cognition::Recognizer:



Public Member Functions

- Recognizer ()
- virtual bool addTrainingImage (const std::string &filename, const std::string &name)=0

  Adds a training image path to the training set of known images after you have added 2 or more images, call train to learn and set yourself up for recognition. All images must be the same size!

- virtual bool train ()=0

  starts the learning process on all the known images that are added trough addTrainingImage. You can add more training images after training, but you have to call train again. While training you cannot use recognize()!

- virtual std::string recognize (cv::Mat &face)=0

  does recognition on the face, and returns the most likely match. This face must grayscale and be exactly the same size as the training images.

- virtual std::size_t numTrainingImages ()=0

  gets the number of training images available

- bool trained ()

Protected Attributes

- bool isTrained

9.10.1 Detailed Description

Base class for recognizers. Provides the interface for different kinds of recognizers.

Todo

add method to add already loaded training images (cv::Mat)

Author

Christophe Hesters 29-1-2011

Definition at line 20 of file recognizer.h.

9.10.2 Constructor & Destructor Documentation

9.10.2.1 cognition::Recognizer::Recognizer ( ) [inline]

Definition at line 24 of file recognizer.h.

9.10.3 Member Function Documentation

9.10.3.1 virtual bool cognition::Recognizer::addTrainingImage (const std::string & *filename*, const std::string & *name*) [pure virtual]

Adds a training image path to the training set of known images after you have added 2 or more images, call train to learn and set yourself up for recognition. All images must be the same size!

Parameters

filename	the path where to find the image (all of the same size!)
name	the name you want to attach to the image

Returns

bool true if the path is added succesfully

Implemented in cognition::EigenfaceRecognizer.

9.10.3.2 virtual std::size_t cognition::Recognizer::numTrainingImages ( ) [pure virtual]

gets the number of training images available

Returns

size_t the number of registered training images

Implemented in cognition::EigenfaceRecognizer.

9.10.3.3 virtual std::string cognition::Recognizer::recognize (cv::Mat & *face*) [pure virtual]

does recognition on the face, and returns the most likely match. This face must grayscale and be exactly the same size as the training images.

Parameters

face	the matrix containing the face

Returns

string name of closest match in the set of training images

Implemented in cognition::EigenfaceRecognizer.

9.10.3.4 virtual bool cognition::Recognizer::train ( ) [pure virtual]

starts the learning process on all the known images that are added trough addTrainingImage. You can add more training images after training, but you have to call train again. While training you cannot use recognize()!

Returns

bool true if trained and ready, false otherwise

Implemented in cognition::EigenfaceRecognizer.

9.10.3.5 bool cognition::Recognizer::trained ( ) [inline]

Definition at line 65 of file recognizer.h.

9.10.4 Member Data Documentation

9.10.4.1 bool cognition::Recognizer::isTrained [protected]

Definition at line 68 of file recognizer.h.

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

- visual-control/cognition/recognizer.h

9.11 cognition::TrainerImage Class Reference

Class used to prepare and store trainig images to be used in recognizer classes. This class basically resizes the image and transforms its color to grayscale. Input is assumed to be in BGR format. (captured from cv::VideoCapture).

`#include <trainerimage.h>`

Public Member Functions

- TrainerImage (cv::Size goalSize=cv::Size(140, 200), bool keepAspectRatio=true, const std::string &directoryPath="")

  Constructs this class to resize(either with or without keeping the aspect ratio), grayscale and optionally store images.

- cv::Mat processImage (cv::Mat &image)

  processes the image according to the configuration of this object and returns the processed image.

- bool processAndSaveImage (cv::Mat &image, const std::string &name)

processes the image according to the configuration of this object and stores the processed image in the directoryPath, with the given name.

Protected Attributes

- std::string directoryPath
- cv::Size goalSize
- bool keepAspectRatio

9.11.1 Detailed Description

Class used to prepare and store trainig images to be used in recognizer classes. This class basically resizes the image and transforms its color to grayscale. Input is assumed to be in BGR format. (captured from cv::VideoCapture).

Author

Christophe Hesters 31-1-2011

Definition at line 18 of file trainerimage.h.

9.11.2 Constructor & Destructor Documentation

9.11.2.1 cognition::TrainerImage::TrainerImage (cv::Size *goalSize* = `cv::Size(140,200)`, bool *keepAspectRatio* = `true`, const std::string & *directoryPath* = `""`) `[inline]`

Constructs this class to resize(either with or without keeping the aspect ratio), grayscale and optionally store images.

Parameters

goalSize	the size to which images are to be scaled
keepAspectRatio	true if the image should be proportionally scaled
directoryPath	the directory where to store the images (default = current working dir.)

Definition at line 29 of file trainerimage.h.

9.11.3 Member Function Documentation

9.11.3.1 bool cognition::TrainerImage::processAndSaveImage (cv::Mat & *image,* const std::string & *name*)

processes the image according to the configuration of this object and stores the processed image in the directoryPath, with the given name.

Parameters

image	the image to be processed
name	the name to save this image under

Returns

true if succesfully saved, false otherwise

Definition at line 53 of file trainerimage.cpp.

9.11.3.2 cv::Mat cognition::TrainerImage::processImage (cv::Mat & *image*)

processes the image according to the configuration of this object and returns the processed image.

Parameters

image	the image to processed

Returns

the processed matrix

Definition at line 3 of file trainerimage.cpp.

9.11.4 Member Data Documentation

9.11.4.1 std::string cognition::TrainerImage::directoryPath [protected]

Definition at line 59 of file trainerimage.h.

9.11.4.2 cv::Size cognition::TrainerImage::goalSize [protected]

Definition at line 60 of file trainerimage.h.

9.11.4.3 bool cognition::TrainerImage::keepAspectRatio [protected]

Definition at line 61 of file trainerimage.h.

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

- visual-control/cognition/trainerimage.h
- visual-control/cognition/trainerimage.cpp

9.12 VisualControl Class Reference

The main window.

`#include <visualcontrol.h>`

Public Member Functions

- VisualControl (QWidget *parent=0, Qt::WFlags flags=0)
- ~VisualControl ()

9.12.1 Detailed Description

The main window.

Author

Christophe Hesters

Note

put this in the gui namespace!

Definition at line 12 of file visualcontrol.h.

9.12.2 Constructor & Destructor Documentation

9.12.2.1 VisualControl::VisualControl (QWidget * *parent* = 0, Qt::WFlags *flags* = 0)

Definition at line 3 of file visualcontrol.cpp.

9.12.2.2 VisualControl::~VisualControl ( )

Definition at line 9 of file visualcontrol.cpp.

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

- visual-control/visualcontrol.h
- visual-control/visualcontrol.cpp

9.13 gui::VisualControl Class Reference

Main application window.

`#include <visualcontrol.h>`

Public Slots

- void captureTrainingImage ()
- void trainRecognizer ()
- void recognizeFaces ()

Public Member Functions

- VisualControl (QWidget *parent=0, Qt::WFlags flags=0)
- ~VisualControl ()

Protected Member Functions

- void closeEvent (QCloseEvent *event)
- void setupFramework ()
- void setupGUI ()
- void captureFrameAndFaces (cognition::Detector::RectVector &rects, cv::Mat &frame)

Protected Attributes

- WebcamWidget * webcamWidget
- QListWidget * logWidget
- QPushButton * captureTrainingImageButton
- QPushButton * trainRecognizerButton
- QPushButton * recognizeFaceButton
- boost::thread frameCaptureThread
- boost::thread faceDetectorThread
- boost::shared_ptr< cv::VideoCapture > videoCapture
- boost::shared_ptr< cognition::FrameCapture > frameCapture
- boost::shared_ptr< cognition::DetailedFaceDetector > faceDetector
- boost::shared_ptr< cognition::EigenfaceRecognizer > recognizer

Static Protected Attributes

- static const cv::Size testImageSize

9.13.1 Detailed Description

Main application window.

Author

Christophe hesters

Definition at line 24 of file visualcontrol.h.

9.13.2 Constructor & Destructor Documentation

9.13.2.1 VisualControl::VisualControl (QWidget * *parent* = 0, Qt::WFlags *flags* = 0)

Definition at line 24 of file visualcontrol.cpp.

9.13.2.2 VisualControl::~VisualControl ( )

Definition at line 41 of file visualcontrol.cpp.

9.13.3 Member Function Documentation

9.13.3.1 void VisualControl::captureFrameAndFaces (cognition::Detector::RectVector & *rects*, cv::Mat & *frame*) [protected]

Definition at line 213 of file visualcontrol.cpp.

9.13.3.2 void VisualControl::captureTrainingImage ( ) [slot]

Definition at line 130 of file visualcontrol.cpp.

9.13.3.3 void VisualControl::closeEvent (QCloseEvent * *event*) [protected]

Definition at line 69 of file visualcontrol.cpp.

9.13.3.4 void VisualControl::recognizeFaces ( ) [slot]

Definition at line 180 of file visualcontrol.cpp.

9.13.3.5 void VisualControl::setupFramework ( ) [protected]

Definition at line 45 of file visualcontrol.cpp.

9.13.3.6 void VisualControl::setupGUI ( ) [protected]

Definition at line 94 of file visualcontrol.cpp.

9.13.3.7 void VisualControl::trainRecognizer ( ) [slot]

Definition at line 161 of file visualcontrol.cpp.

9.13.4 Member Data Documentation

9.13.4.1 QPushButton* gui::VisualControl::captureTrainingImageButton [protected]

Definition at line 44 of file visualcontrol.h.

9.13.4.2 boost::shared_ptr<cognition::DetailedFaceDetector> gui::VisualControl::faceDetector [protected]

Definition at line 54 of file visualcontrol.h.

9.13.4.3 boost::thread gui::VisualControl::faceDetectorThread [protected]

Definition at line 49 of file visualcontrol.h.

9.13.4.4 boost::shared_ptr<cognition::FrameCapture> gui::VisualControl::frameCapture [protected]

Definition at line 53 of file visualcontrol.h.

9.13.4.5 boost::thread gui::VisualControl::frameCaptureThread [protected]

Definition at line 48 of file visualcontrol.h.

9.13.4.6 QListWidget* gui::VisualControl::logWidget [protected]

Definition at line 42 of file visualcontrol.h.

9.13.4.7 QPushButton* gui::VisualControl::recognizeFaceButton [protected]

Definition at line 46 of file visualcontrol.h.

9.13.4.8 boost::shared_ptr<cognition::EigenfaceRecognizer> gui::VisualControl::recognizer [protected]

Definition at line 55 of file visualcontrol.h.

9.13.4.9 const cv::Size VisualControl::testImageSize [static, protected]

Definition at line 39 of file visualcontrol.h.

9.13.4.10 QPushButton* gui::VisualControl::trainRecognizerButton [protected]

Definition at line 45 of file visualcontrol.h.

9.13.4.11 boost::shared_ptr<cv::VideoCapture> gui::VisualControl::videoCapture [protected]

Definition at line 52 of file visualcontrol.h.

9.13.4.12 WebcamWidget* gui::VisualControl::webcamWidget [protected]

Definition at line 41 of file visualcontrol.h.

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

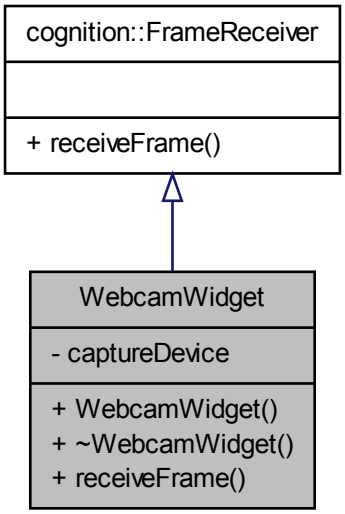
- visual-control/gui/visualcontrol.h
- visual-control/gui/visualcontrol.cpp

9.14 WebcamWidget Class Reference

A widget that displays feeds that it receives from a FrameCapture device and displays the information of the detector.

`#include <webcamwidget.h>`

Inheritance diagram for WebcamWidget:



Public Types

- typedef boost::shared_ptr< cognition::FrameCapture > CaptureDevicePtr

Public Member Functions

- WebcamWidget (QWidget *parent, CaptureDevicePtr existingCaptureDevice)
- virtual ~WebcamWidget ()
- virtual void receiveFrame (const cv::Mat &frame)

  implementors receive frames trough this method from FrameCapture

Private Attributes

- CaptureDevicePtr captureDevice

9.14.1 Detailed Description

A widget that displays feeds that it receives from a FrameCapture device and displays the information of the detector.

Author

Christophe Hesters

Note

put this in the gui namespace!

Definition at line 16 of file webcamwidget.h.

9.14.2 Member Typedef Documentation

9.14.2.1 typedef boost::shared_ptr<cognition::FrameCapture> WebcamWidget::CaptureDevicePtr

Definition at line 21 of file webcamwidget.h.

9.14.3 Constructor & Destructor Documentation

9.14.3.1 WebcamWidget::WebcamWidget (QWidget * *parent*, CaptureDevicePtr *existingCaptureDevice*)

Definition at line 3 of file webcamwidget.cpp.

9.14.3.2 WebcamWidget::~WebcamWidget ( ) [virtual]

Definition at line 10 of file webcamwidget.cpp.

9.14.4 Member Function Documentation

9.14.4.1 void WebcamWidget::receiveFrame (const cv::Mat & *frame*) [virtual]

implementors receive frames trough this method from FrameCapture

Parameters

frame	the new frame

Implements cognition::FrameReceiver.

Definition at line 16 of file webcamwidget.cpp.

9.14.5 Member Data Documentation

9.14.5.1 CaptureDevicePtr WebcamWidget::captureDevice [private]

Definition at line 40 of file webcamwidget.h.

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

- visual-control/webcamwidget.h

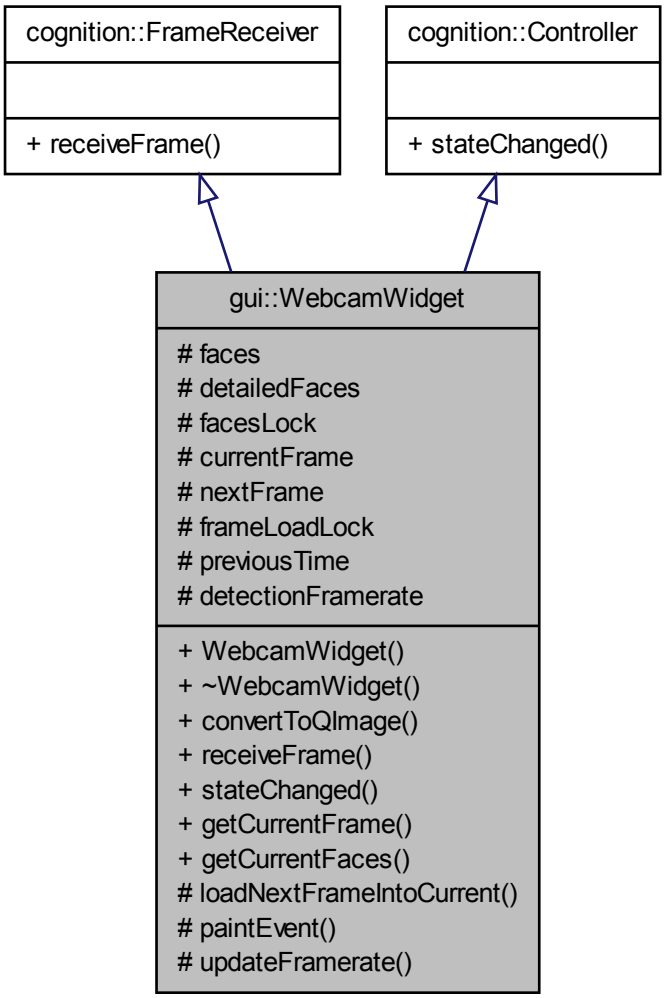
- visual-control/webcamwidget.cpp

9.15 gui::WebcamWidget Class Reference

Widget that displays the webcam feed, provided by a capture devices it also draws the rectangles it recieves from detectors!

```
#include <webcamwidget.h>
```

Inheritance diagram for gui::WebcamWidget:



Public Types

- typedef cognition::FrameCapture * FrameCapturePtr

Public Member Functions

- WebcamWidget (QWidget *parent, FrameCapturePtr existingCaptureDevice)

- virtual ~WebcamWidget ()
- QImage convertToQImage (const cv::Mat &mat) const
- void receiveFrame (const cv::Mat &frame)

  implementors receive frames trough this method from FrameCapture

- void stateChanged (cognition::Detector *detector)

  This method will be called when a detector has an update.

- cv::Mat getCurrentFrame ()
- cognition::Detector::RectVector getCurrentFaces ()

Protected Member Functions

- virtual void loadNextFrameIntoCurrent ()
- void paintEvent (QPaintEvent *event)
- void updateFramerate ()

Protected Attributes

- cognition::Detector::RectVector faces
- cognition::DetailedFaceDetector::DetailedFaces detailedFaces
- boost::mutex facesLock
- cv::Mat currentFrame
- cv::Mat nextFrame
- boost::mutex frameLoadLock
- boost::posix_time::ptime previousTime
- volatile int detectionFramerate

9.15.1 Detailed Description

Widget that displays the webcam feed, provided by a capture devices it also draws the rectangles it recieves from detectors!

Author

Christophe hesters

Definition at line 26 of file webcamwidget.h.

9.15.2 Member Typedef Documentation

9.15.2.1 typedef cognition::FrameCapture* gui::WebcamWidget::FrameCapturePtr

Definition at line 34 of file webcamwidget.h.

9.15.3 Constructor & Destructor Documentation

9.15.3.1 WebcamWidget::WebcamWidget (QWidget * *parent,* FrameCapturePtr *existingCaptureDevice*)

Definition at line 15 of file webcamwidget.cpp.

9.15.3.2 WebcamWidget::~WebcamWidget ( ) [virtual]

Definition at line 30 of file webcamwidget.cpp.

9.15.4 Member Function Documentation

9.15.4.1 QImage WebcamWidget::convertToQImage (const cv::Mat & *mat*) const

Definition at line 35 of file webcamwidget.cpp.

9.15.4.2 cognition::Detector::RectVector WebcamWidget::getCurrentFaces ( )

Definition at line 157 of file webcamwidget.cpp.

9.15.4.3 cv::Mat WebcamWidget::getCurrentFrame ( )

Definition at line 151 of file webcamwidget.cpp.

9.15.4.4 void WebcamWidget::loadNextFrameIntoCurrent ( ) [protected, virtual]

Definition at line 163 of file webcamwidget.cpp.

9.15.4.5 void WebcamWidget::paintEvent (QPaintEvent * *event*) [protected]

Definition at line 41 of file webcamwidget.cpp.

9.15.4.6 void WebcamWidget::receiveFrame (const cv::Mat & *frame*) [virtual]

implementors receive frames trough this method from FrameCapture

Parameters

frame	the new frame

Implements cognition::FrameReceiver.

Definition at line 138 of file webcamwidget.cpp.

9.15.4.7 void WebcamWidget::stateChanged (cognition::Detector * *detector*) [virtual]

This method will be called when a detector has an update.

Parameters

detector	the detector that has an updated status

Implements cognition::Controller.

Definition at line 116 of file webcamwidget.cpp.

9.15.4.8 void WebcamWidget::updateFramerate ( ) [inline, protected]

Definition at line 127 of file webcamwidget.cpp.

9.15.5 Member Data Documentation

9.15.5.1 cv::Mat gui::WebcamWidget::currentFrame [protected]

Definition at line 56 of file webcamwidget.h.

9.15.5.2 cognition::DetailedFaceDetector::DetailedFaces gui::WebcamWidget::detailedFaces [protected]

Definition at line 53 of file webcamwidget.h.

9.15.5.3 volatile int gui::WebcamWidget::detectionFramerate [protected]

Definition at line 68 of file webcamwidget.h.

9.15.5.4 cognition::Detector::RectVector gui::WebcamWidget::faces [protected]

Definition at line 52 of file webcamwidget.h.

9.15.5.5 boost::mutex gui::WebcamWidget::facesLock [protected]

Definition at line 54 of file webcamwidget.h.

9.15.5.6 boost::mutex gui::WebcamWidget::frameLoadLock [protected]

Definition at line 60 of file webcamwidget.h.

9.15.5.7 cv::Mat gui::WebcamWidget::nextFrame [protected]

Definition at line 57 of file webcamwidget.h.

9.15.5.8 boost::posix_time::ptime gui::WebcamWidget::previousTime [protected]

Definition at line 66 of file webcamwidget.h.

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

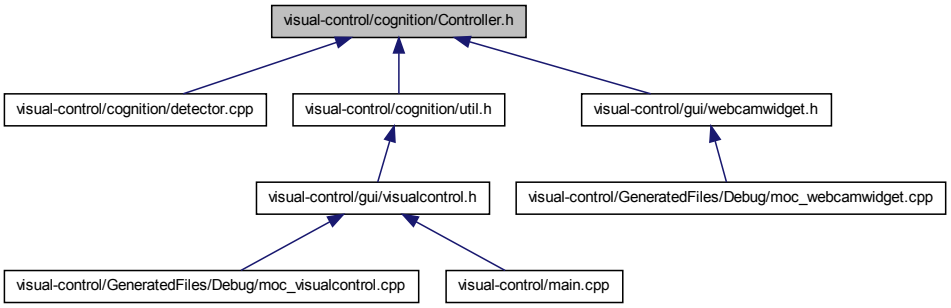
- visual-control/gui/webcamwidget.h
- visual-control/gui/webcamwidget.cpp

Chapter 10

File Documentation

10.1 visual-control/cognition/Controller.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- class cognition::Controller

  Interface for objects that want to receive status change notifications from detectors.

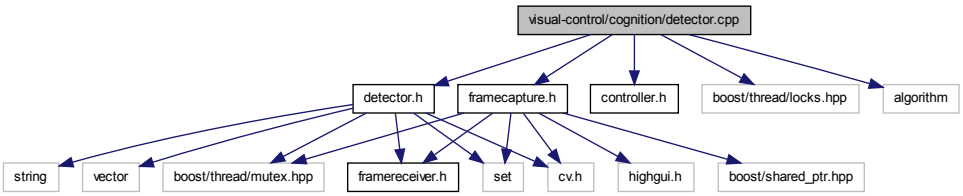
Namespaces

- namespace cognition

10.2 visual-control/cognition/detector.cpp File Reference

```
#include "detector.h"
#include "controller.h"
#include <boost/thread/locks.hpp>
#include <algorithm>
#include "framecapture.h"
```

Include dependency graph for detector.cpp:



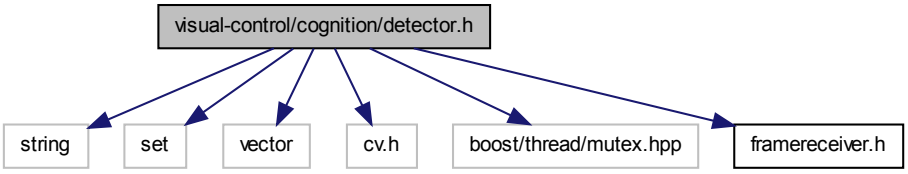
Namespaces

- namespace cognition

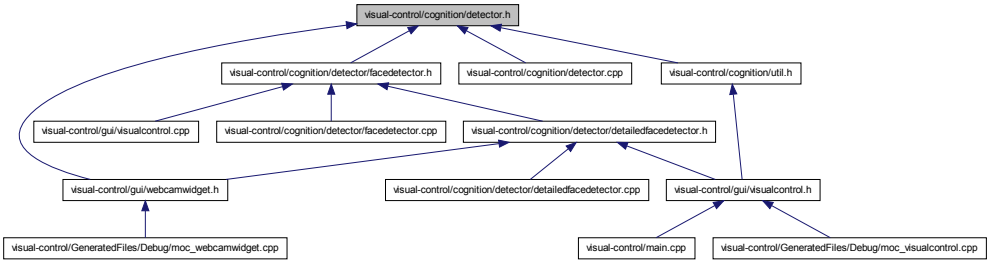
10.3 visual-control/cognition/detector.h File Reference

```
#include <string>
#include <set>
#include <vector>
#include <cv.h>
#include <boost/thread/mutex.hpp>
#include "framereceiver.h"
```

Include dependency graph for detector.h:



This graph shows which files directly or indirectly include this file:



Classes

- class cognition::Detector

  Base class for detectors. Provides threading system and thread safe methods for getting the most recent frame, notifying controllers and getting the detection result.

Namespaces

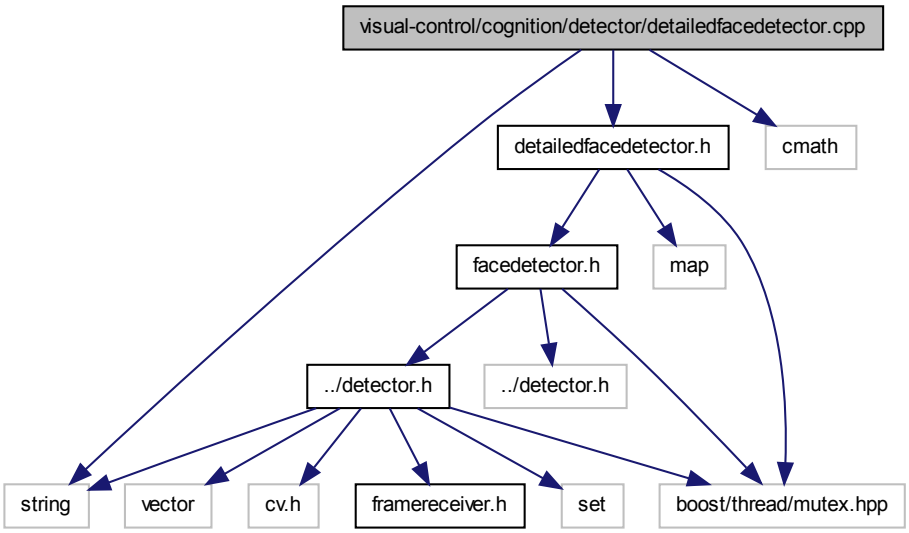
- namespace cognition

10.4 visual-control/cognition/detector/detailedfacedetector.cpp File Reference

```
#include "detailedfacedetector.h"
#include <string>
```

```
#include <cmath>
```

Include dependency graph for detailedfacedetector.cpp:



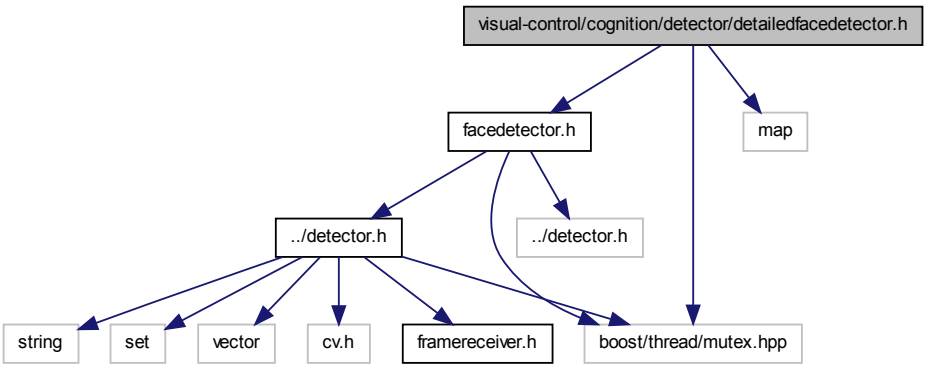
Namespaces

- namespace cognition

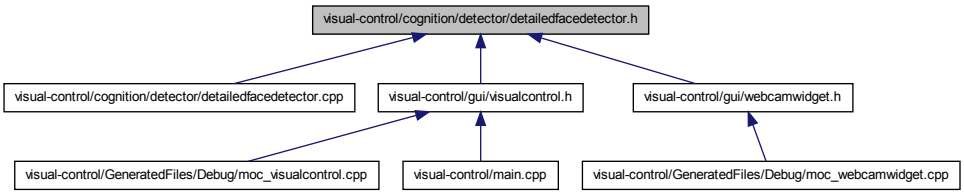
10.5 visual-control/cognition/detector/detailedfacedetector.h File Reference

```
#include "facedetector.h"
#include <map>
#include <boost/thread/mutex.hpp>
```

Include dependency graph for detailedfacedetector.h:



This graph shows which files directly or indirectly include this file:



Classes

- class cognition::DetailedFaceDetector

  Detects faces and the corresponding eyes nose and mouth that are part of that face.

- struct cognition::DetailedFaceDetector::less_cvrect

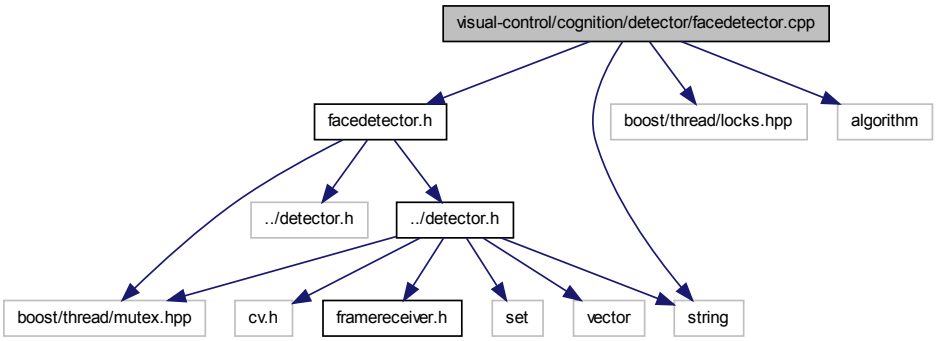
Namespaces

- namespace cognition

10.6 visual-control/cognition/detector/facedetector.cpp File Reference

```
#include "facedetector.h"
#include <boost/thread/locks.hpp>
#include <algorithm>
#include <string>
```

Include dependency graph for facedetector.cpp:



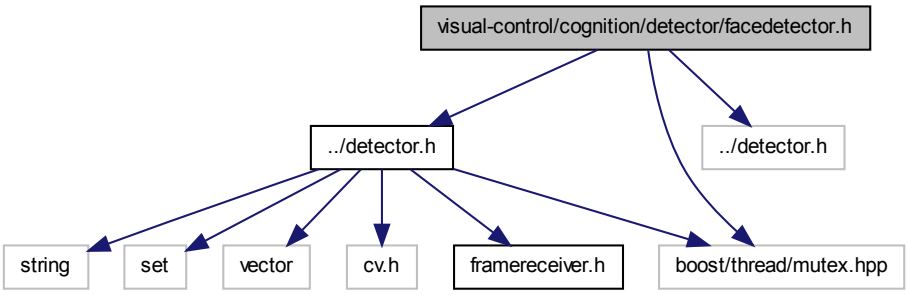
Namespaces

- namespace cognition

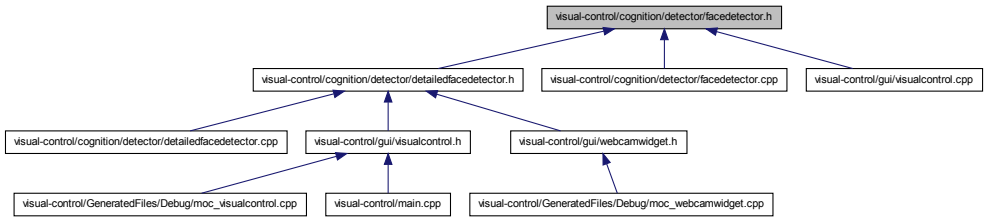
10.7 visual-control/cognition/detector/facedetector.h File Reference

```
#include "../detector.h"
#include <boost/thread/mutex.hpp>
#include "../detector.h"
```

Include dependency graph for facedetector.h:



This graph shows which files directly or indirectly include this file:



Classes

- class cognition::FaceDetector

  Basic detector that detects faces inside a frame.

Namespaces

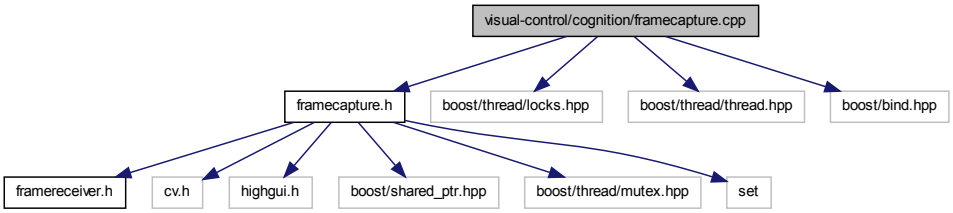
- namespace cognition

10.8 visual-control/cognition/framecapture.cpp File Reference

```
#include "framecapture.h"
#include <boost/thread/locks.hpp>
```

```
#include <boost/thread/thread.hpp>
#include <boost/bind.hpp>
```

Include dependency graph for framecapture.cpp:



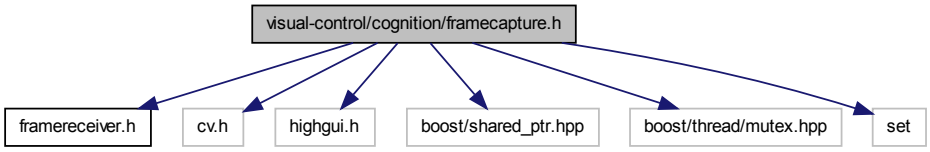
Namespaces

- namespace cognition

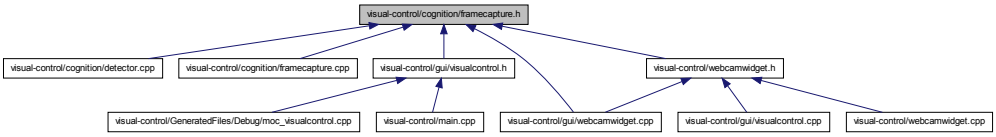
10.9 visual-control/cognition/framecapture.h File Reference

```
#include "framereceiver.h"
#include <cv.h>
#include <highgui.h>
#include <boost/shared_ptr.hpp>
#include <boost/thread/mutex.hpp>
#include <set>
```

Include dependency graph for framecapture.h:



This graph shows which files directly or indirectly include this file:



Classes

- class cognition::FrameCapture

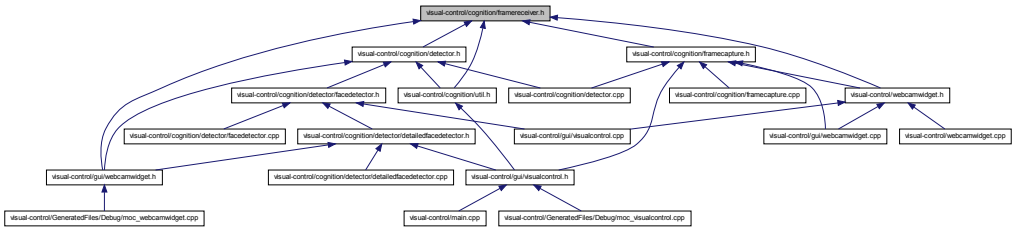
  Captures frames from a capture device (cv::VideoCapture) on a certain framerate and sends the frames to all attached frame receivers.

Namespaces

- namespace cognition

10.10 visual-control/cognition/framereceiver.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- class cognition::FrameReceiver

  Interfaces for classes that receive frames.

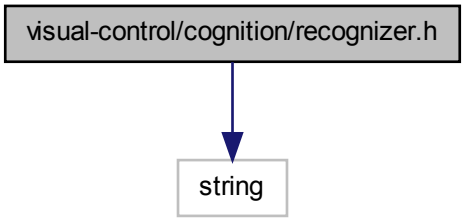
Namespaces

- namespace cv
- namespace cognition

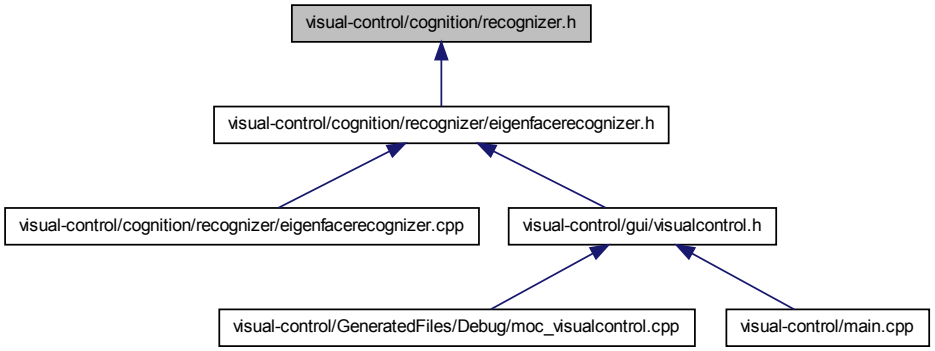
10.11 visual-control/cognition/recognizer.h File Reference

```
#include <string>
```

Include dependency graph for recognizer.h:



This graph shows which files directly or indirectly include this file:



Classes

- class cognition::Recognizer

  Base class for recognizers. Provides the interface for different kinds of recognizers.

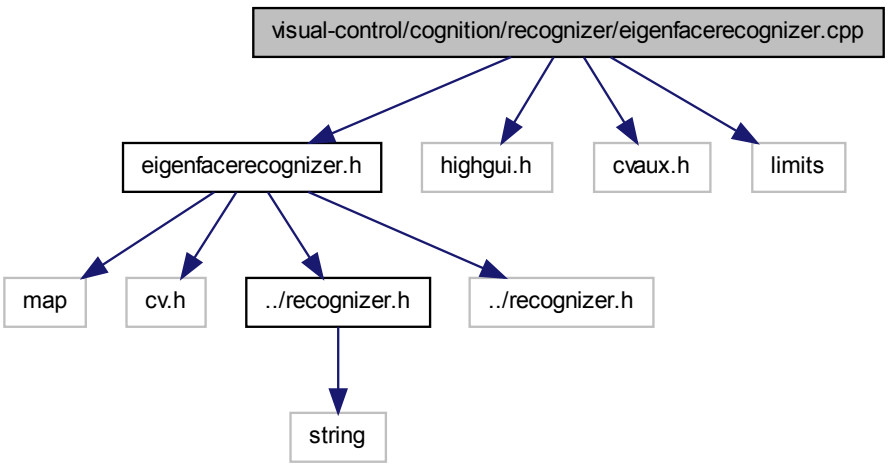
Namespaces

- namespace cv
- namespace cognition

10.12 visual-control/cognition/recognizer/eigenfacerecognizer.cpp File Reference

```
#include "eigenfacerecognizer.h"
#include <highgui.h>
#include <cvaux.h>
#include <limits>
```

Include dependency graph for eigenfacerecognizer.cpp:



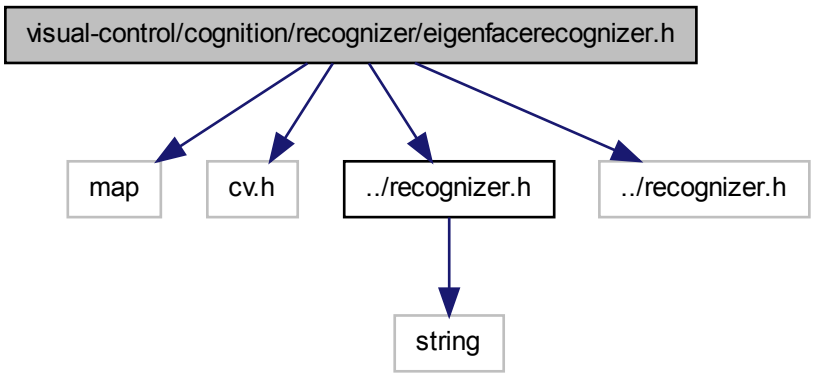
Namespaces

- namespace cognition

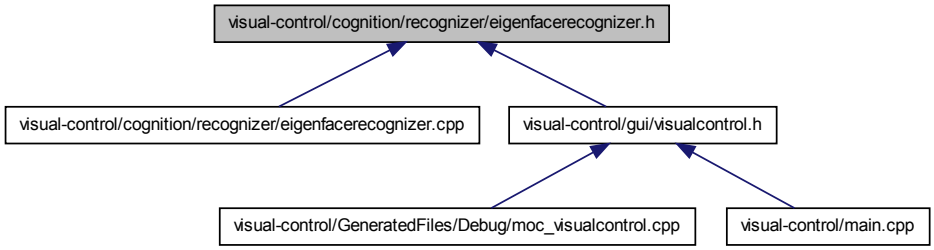
10.13 visual-control/cognition/recognizer/eigenfacerecognizer.h File Reference

```
#include <map>
#include <cv.h>
#include "../recognizer.h"
#include "../recognizer.h"
```

Include dependency graph for eigenfacerecognizer.h:



This graph shows which files directly or indirectly include this file:



Classes

- class cognition::EigenfaceRecognizer

  Implementation of the Eigenface face recognition algorithm it uses old style C functions of the OpenCV library, based on `http://www.cognotics.com/opencv/servo_2007_series/part_5/index.html` *by Robin Hewitt.*

Namespaces

- namespace cognition

Defines

- #define _CRT_SECURE_NO_DEPRECATE
- #define _CRT_SECURE_NO_WARNINGS

10.13.1 Define Documentation

10.13.1.1 #define _CRT_SECURE_NO_DEPRECATE

Definition at line 8 of file eigenfacerecognizer.h.

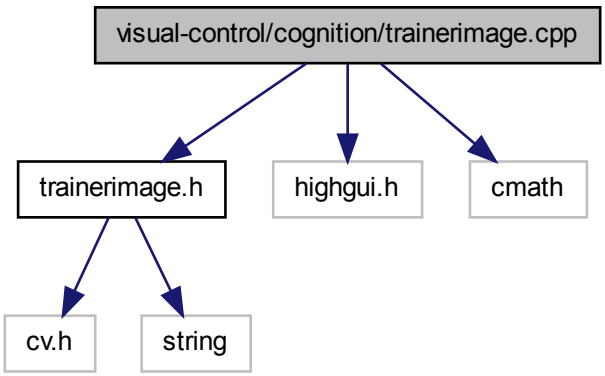
10.13.1.2 #define _CRT_SECURE_NO_WARNINGS

Definition at line 9 of file eigenfacerecognizer.h.

10.14 visual-control/cognition/trainerimage.cpp File Reference

```
#include "trainerimage.h"
#include <highgui.h>
#include <cmath>
```

Include dependency graph for trainerimage.cpp:



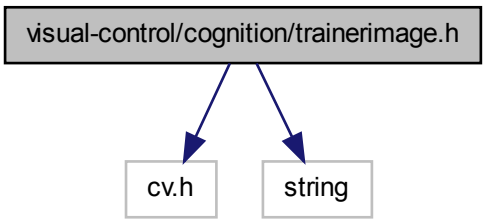
Namespaces

- namespace cognition

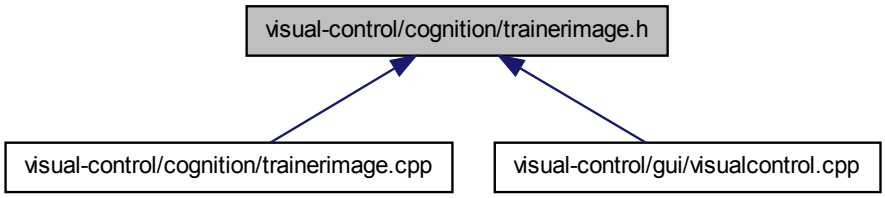
10.15 visual-control/cognition/trainerimage.h File Reference

```
#include <cv.h>
#include <string>
```

Include dependency graph for trainerimage.h:



This graph shows which files directly or indirectly include this file:



Classes

- class cognition::TrainerImage

  Class used to prepare and store trainig images to be used in recognizer classes. This class basically resizes the image and transforms its color to grayscale. Input is assumed to be in BGR format. (captured from cv::VideoCapture).

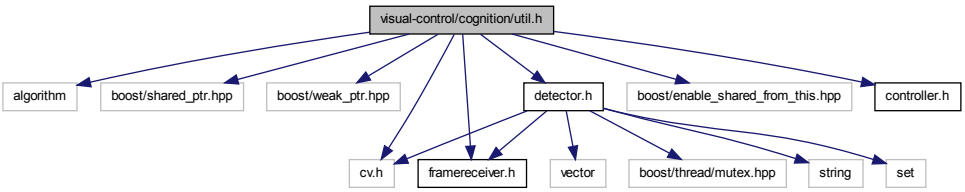
Namespaces

- namespace cognition

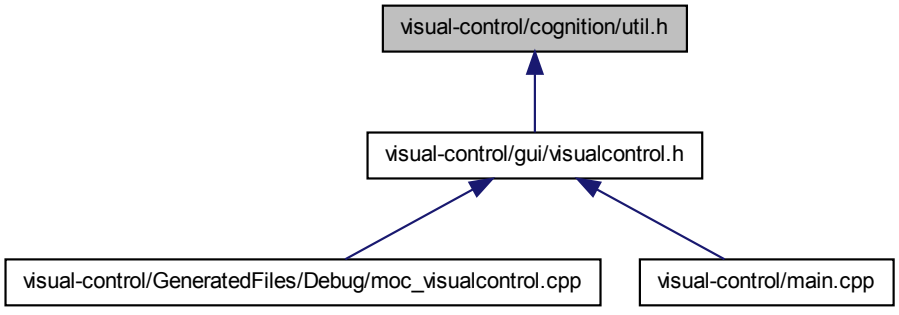
10.16 visual-control/cognition/util.h File Reference

```
#include <algorithm>
#include <boost/shared_ptr.hpp>
#include <boost/weak_ptr.hpp>
#include <cv.h>
#include "framereceiver.h"
#include <boost/enable_shared_from_this.hpp>
#include "detector.h"
#include "controller.h"
```

Include dependency graph for util.h:



This graph shows which files directly or indirectly include this file:



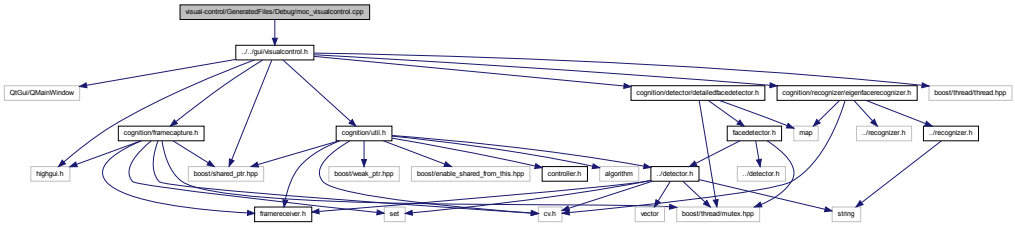
Namespaces

- namespace cognition

10.17 visual-control/GeneratedFiles/Debug/moc_visualcontrol.cpp File Reference

```
#include "../../gui/visualcontrol.h"
```

Include dependency graph for moc_visualcontrol.cpp:



Variables

- static QT_BEGIN_MOC_NAMESPACE const uint qt_meta_data_gui__VisualControl [ ]
- static const char qt_meta_stringdata_gui__VisualControl [ ]

10.17.1 Variable Documentation

10.17.1.1 QT_BEGIN_MOC_NAMESPACE const uint qt_meta_data_gui__VisualControl[ ] [static]

Initial value:

```
{

       5,
       0,
       0,    0,
       3,   14,
       0,    0,
       0,    0,
       0,    0,
       0,
       0,


      20,   19,   19,   19, 0x0a,
```

```
      43,   19,   19,   19, 0x0a,
      61,   19,   19,   19, 0x0a,

       0
}
```

Definition at line 20 of file moc_visualcontrol.cpp.

10.17.1.2 const char qt_meta_stringdata_gui__VisualControl[] [static]

Initial value:

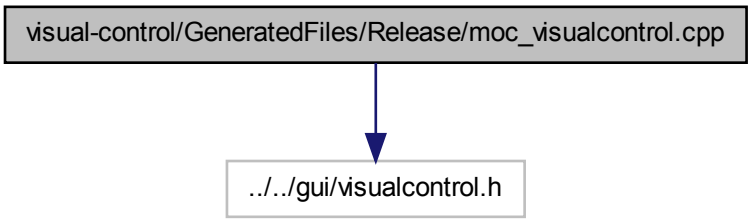
```
 {
    "gui::VisualControl\0\0captureTrainingImage()\0"
    "trainRecognizer()\0recognizeFaces()\0"
}
```

Definition at line 41 of file moc_visualcontrol.cpp.

10.18 visual-control/GeneratedFiles/Release/moc_visualcontrol.cpp File Reference

```
#include "../../gui/visualcontrol.h"
```

Include dependency graph for moc_visualcontrol.cpp:



Variables

- static QT_BEGIN_MOC_NAMESPACE const uint qt_meta_data_gui__VisualControl []
- static const char qt_meta_stringdata_gui__VisualControl []

10.18.1 Variable Documentation

10.18.1.1 QT_BEGIN_MOC_NAMESPACE const uint qt_meta_data_gui__VisualControl[] [static]

Initial value:

```
 {

       5,
       0,
       0,    0,
       3,   14,
       0,    0,
       0,    0,
       0,    0,
       0,
       0,


      20,   19,   19,   19, 0x0a,
      43,   19,   19,   19, 0x0a,
      61,   19,   19,   19, 0x0a,

       0
}
```

Definition at line 20 of file moc_visualcontrol.cpp.

10.18.1.2 const char qt_meta_stringdata_gui__VisualControl[] [static]

Initial value:

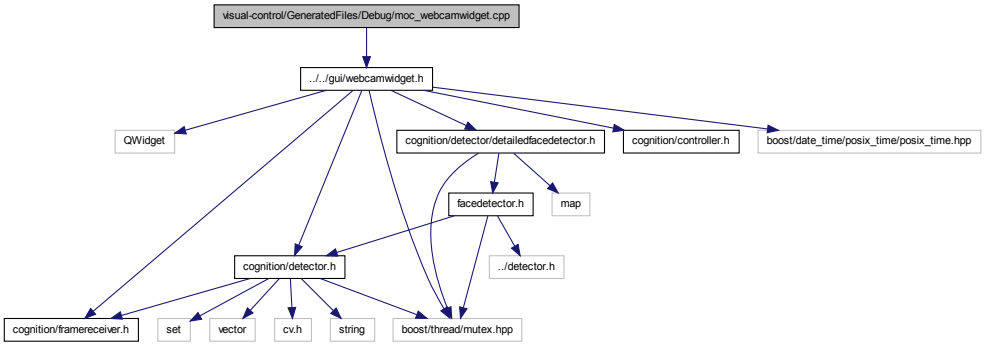
```
 {
    "gui::VisualControl\0\0captureTrainingImage()\0"
    "trainRecognizer()\0recognizeFaces()\0"
}
```

Definition at line 41 of file moc_visualcontrol.cpp.

10.19 visual-control/GeneratedFiles/Debug/moc_webcamwidget.cpp File Reference

```
#include "../../gui/webcamwidget.h"
```

Include dependency graph for moc_webcamwidget.cpp:



Variables

- static QT_BEGIN_MOC_NAMESPACE const uint qt_meta_data_gui__WebcamWidget [ ]
- static const char qt_meta_stringdata_gui__WebcamWidget [ ]

10.19.1 Variable Documentation

10.19.1.1 QT_BEGIN_MOC_NAMESPACE const uint qt_meta_data_gui__WebcamWidget[ ] [static]

Initial value:

```
{

        5,
        0,
        0,    0,
        0,    0,
        0,    0,
        0,    0,
        0,    0,
        0,
        0,

        0
}
```

Definition at line 20 of file moc_webcamwidget.cpp.

10.19.1.2 const char qt_meta_stringdata_gui__WebcamWidget[ ] [static]

Initial value:

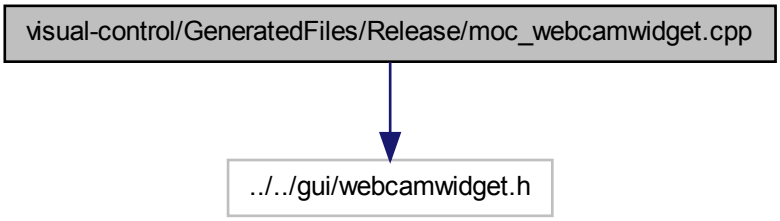
```
 {
    "gui::WebcamWidget\0"
}
```

Definition at line 36 of file moc_webcamwidget.cpp.

10.20 visual-control/GeneratedFiles/Release/moc_webcamwidget.cpp File Reference

```
#include "../../gui/webcamwidget.h"
```

Include dependency graph for moc_webcamwidget.cpp:



Variables

- static QT_BEGIN_MOC_NAMESPACE const uint qt_meta_data_gui__WebcamWidget [ ]
- static const char qt_meta_stringdata_gui__WebcamWidget [ ]

10.20.1 Variable Documentation

10.20.1.1 QT_BEGIN_MOC_NAMESPACE const uint qt_meta_data_gui__WebcamWidget[ ] [static]

Initial value:

```
 {

       5,
       0,
       0,    0,
```

```
        0,    0,
        0,    0,
        0,    0,
        0,    0,
        0,
        0,

        0
}
```

Definition at line 20 of file moc_webcamwidget.cpp.

10.20.1.2 const char qt_meta_stringdata_gui__WebcamWidget[] [static]

Initial value:

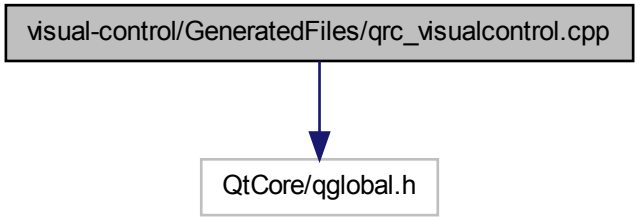
```
 {
    "gui::WebcamWidget\0"
}
```

Definition at line 36 of file moc_webcamwidget.cpp.

10.21 visual-control/GeneratedFiles/qrc_visualcontrol.cpp File Reference

```
#include <QtCore/qglobal.h>
```

Include dependency graph for qrc_visualcontrol.cpp:



Functions

- QT_BEGIN_NAMESPACE QT_END_NAMESPACE int QT_MANGLE_NAMESPACE() qInitResources_visualcontrol ()
- int QT_MANGLE_NAMESPACE() qCleanupResources_visualcontrol ()

10.21.1 Function Documentation

10.21.1.1 int QT_MANGLE_NAMESPACE() qCleanupResources_visualcontrol ( )

Definition at line 24 of file qrc_visualcontrol.cpp.

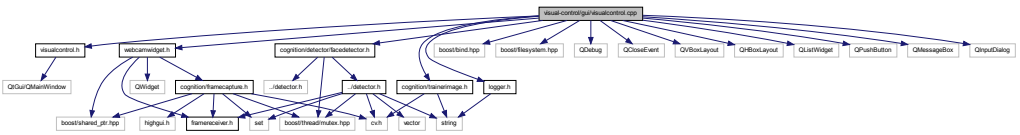
10.21.1.2 QT_BEGIN_NAMESPACE QT_END_NAMESPACE int QT_MANGLE_NAMESPACE() qInitResources_visualcontrol ( )

Definition at line 17 of file qrc_visualcontrol.cpp.

10.22 visual-control/gui/visualcontrol.cpp File Reference

```
#include "visualcontrol.h"
#include "webcamwidget.h"
#include <boost/bind.hpp>
#include <boost/filesystem.hpp>
#include <QDebug>
#include <QCloseEvent>
#include <QVBoxLayout>
#include <QHBoxLayout>
#include <QListWidget>
#include <QPushButton>
#include <QMessageBox>
#include <QInputDialog>
#include <logger.h>
#include "cognition/detector/facedetector.h"
#include "cognition/trainerimage.h"
```

Include dependency graph for visualcontrol.cpp:



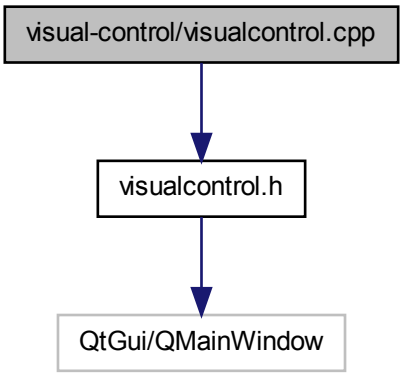
Namespaces

- namespace gui

10.23 visual-control/visualcontrol.cpp File Reference

```
#include "visualcontrol.h"
```

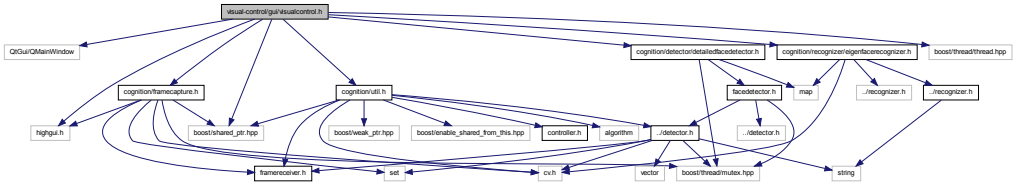
Include dependency graph for visualcontrol.cpp:



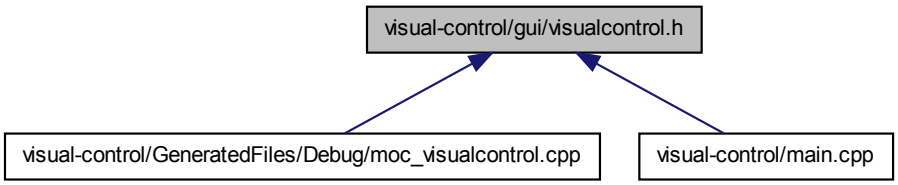
10.24 visual-control/gui/visualcontrol.h File Reference

```
#include <QtGui/QMainWindow>
#include <highgui.h>
#include <boost/shared_ptr.hpp>
#include <boost/thread/thread.hpp>
#include "cognition/framecapture.h"
#include "cognition/detector/detailedfacedetector.h"
#include "cognition/util.h"
#include "cognition/recognizer/eigenfacerecognizer.h"
```

Include dependency graph for visualcontrol.h:



This graph shows which files directly or indirectly include this file:



Classes

- class gui::VisualControl

  Main application window.

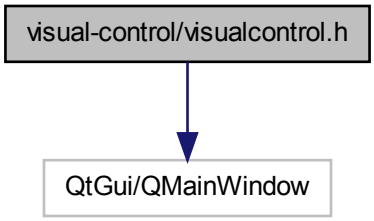
Namespaces

- namespace gui

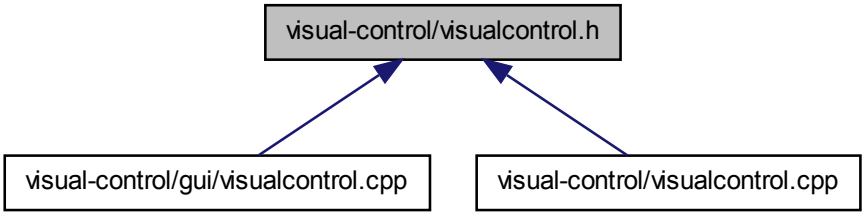
10.25 visual-control/visualcontrol.h File Reference

```
#include <QtGui/QMainWindow>
```

Include dependency graph for visualcontrol.h:



This graph shows which files directly or indirectly include this file:



Classes

- class VisualControl

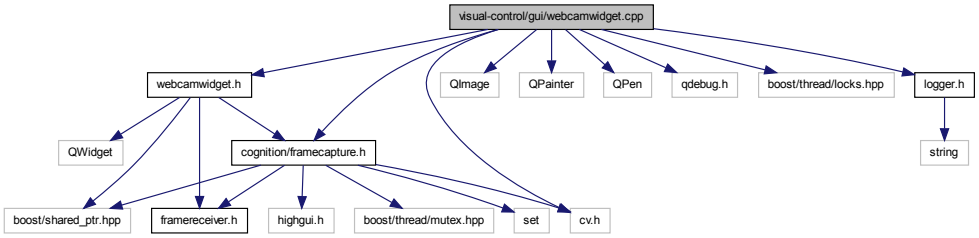
  The main window.

10.26 visual-control/gui/webcamwidget.cpp File Reference

```
#include "webcamwidget.h"
#include <QImage>
#include <QPainter>
```

```
#include <QPen>
#include <qdebug.h>
#include <cv.h>
#include <boost/thread/locks.hpp>
#include "cognition/framecapture.h"
#include "logger.h"
```

Include dependency graph for webcamwidget.cpp:



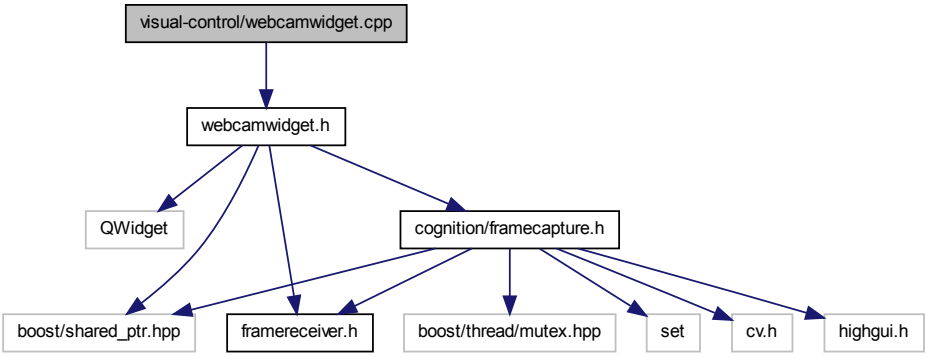
Namespaces

- namespace gui

10.27 visual-control/webcamwidget.cpp File Reference

```
#include "webcamwidget.h"
```

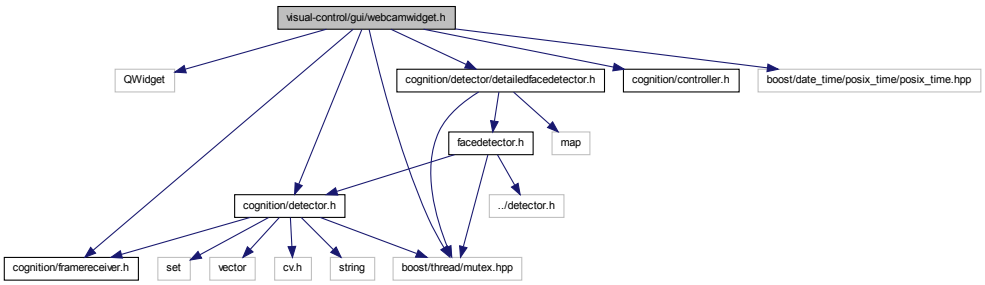
Include dependency graph for webcamwidget.cpp:



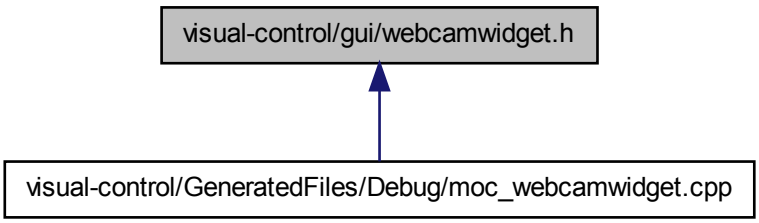
10.28 visual-control/gui/webcamwidget.h File Reference

```
#include <QWidget>
#include "cognition/framereceiver.h"
#include "cognition/controller.h"
#include <boost/thread/mutex.hpp>
#include <boost/date_time/posix_time/posix_time.hpp>
#include "cognition/detector.h"
#include "cognition/detector/detailedfacedetector.h"
```

Include dependency graph for webcamwidget.h:



This graph shows which files directly or indirectly include this file:



Classes

- class gui::WebcamWidget

  Widget that displays the webcam feed, provided by a capture devices it also draws the rectangles it recieves from detectors!

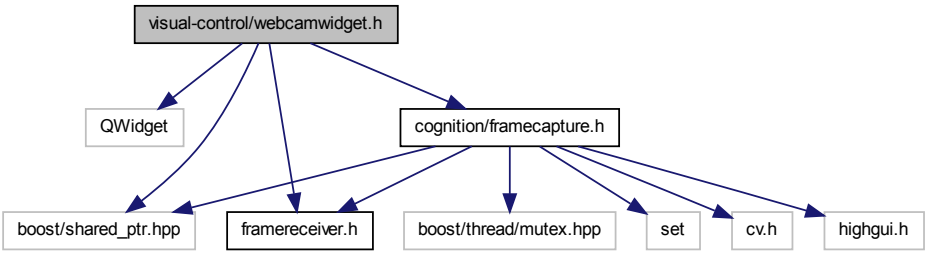
Namespaces

- namespace cognition
- namespace gui

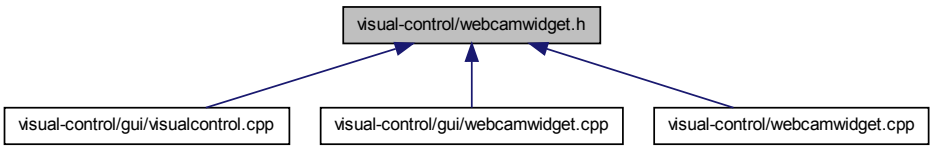
10.29 visual-control/webcamwidget.h File Reference

```
#include <QWidget>
#include <boost/shared_ptr.hpp>
#include "cognition/framecapture.h"
#include "cognition/framereceiver.h"
```

Include dependency graph for webcamwidget.h:



This graph shows which files directly or indirectly include this file:



Classes

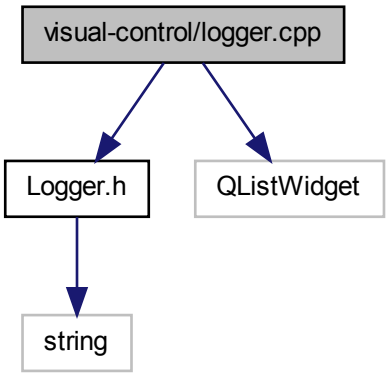
- class WebcamWidget

  A widget that displays feeds that it receives from a FrameCapture device and displays the information of the detector.

10.30 visual-control/logger.cpp File Reference

```
#include "Logger.h"
#include <QListWidget>
```

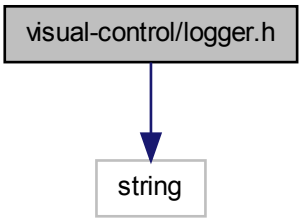
Include dependency graph for logger.cpp:



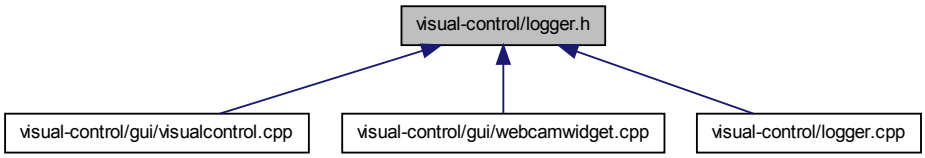
10.31 visual-control/logger.h File Reference

```
#include <string>
```

Include dependency graph for logger.h:



This graph shows which files directly or indirectly include this file:



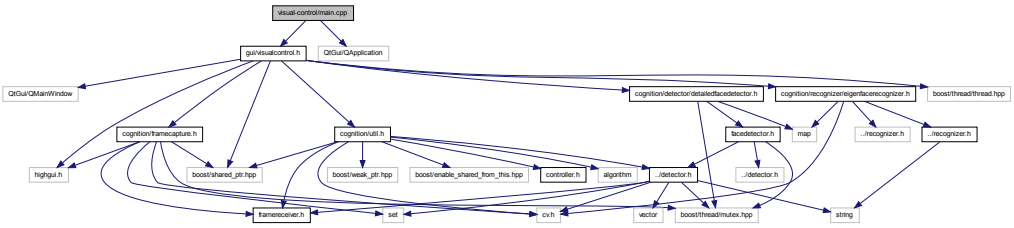
Classes

- class Logger

10.32 visual-control/main.cpp File Reference

```
#include "gui/visualcontrol.h"
#include <QtGui/QApplication>
```

Include dependency graph for main.cpp:



Functions

- int main (int argc, char ∗argv[])

10.32.1 Function Documentation

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

Definition at line 4 of file main.cpp.

Index