LabRoboticsProject

Generated by Doxygen 1.8.14

Contents

1 Class Index 1

1.1 Class List 1

2 File Index 3

2.1 File List 3

3 Class Documentation 5

3.1 Settings Class Reference 5

3.1.1 Member Enumeration Documentation 7

3.1.1.1 InputType 7

3.1.1.2 Pattern 7

3.1.2 Constructor & Destructor Documentation 7

3.1.2.1 Settings() 7

3.1.3 Member Function Documentation 7

3.1.3.1 isListOfImages() 7

3.1.3.2 nextImage() 8

3.1.3.3 read() 8

3.1.3.4 readStringList() 8

3.1.3.5 validate() 9

3.1.3.6 write() 9

3.1.4 Member Data Documentation 10

3.1.4.1 aspectRatio 10

3.1.4.2 atImageList 10

3.1.4.3 boardSize 10

3.1.4.4 calibFixPrincipalPoint 10
3.1.4.5 calibrationPattern 10
3.1.4.6 calibZeroTangentDist 11
3.1.4.7 cameraID 11
3.1.4.8 delay 11
3.1.4.9 fixK1 11
3.1.4.10 fixK2 11
3.1.4.11 fixK3 11
3.1.4.12 fixK4 12
3.1.4.13 fixK5 12
3.1.4.14 flag 12
3.1.4.15 flipVertical 12
3.1.4.16 goodInput 12
3.1.4.17 imageList 12
3.1.4.18 input 12
3.1.4.19 inputCapture 13
3.1.4.20 inputType 13
3.1.4.21 nrFrames 13
3.1.4.22 outputFileName 13
3.1.4.23 showUndistorsed 13
3.1.4.24 squareSize 13
3.1.4.25 useFisheye 14
3.1.4.26 writeExtrinsics 14
3.1.4.27 writePoints 14

4 File Documentation 15
4.1 src/calibration.cc File Reference 15
4.1.1 Function Documentation 16
4.1.1.1 calcBoardCornerPositions() 16
4.1.1.2 calibration() 16
4.1.1.3 computeReprojectionErrors() 16
4.1.1.4 read() 17
4.1.1.5 runCalibration() 17
4.1.1.6 runCalibrationAndSave() 18
4.1.1.7 saveCameraParams() 19
4.2 src/calibration.hh File Reference 19
4.2.1 Detailed Description 20
4.2.2 Enumeration Type Documentation 20
4.2.2.1 anonymous enum 20
4.2.3 Function Documentation 21
4.2.3.1 calibration() 21
4.2.3.2 runCalibrationAndSave() 21
4.3 src/calibration_run.cc File Reference 22
4.3.1 Function Documentation 22
4.3.1.1 main() 22
4.4 src/create_xml.cc File Reference 22
4.4.1 Function Documentation 23
4.4.1.1 main() 23
4.5 src/detection.cc File Reference 23
4.5.1 Macro Definition Documentation 24
4.5.1.1 WAIT 24
4.5.2 Function Documentation 24
4.5.2.1 crop_number_section() 24
4.5.2.2 detection() 24
4.5.2.3 erode_dilation() 25

4.5.2.4 find_contours() 25
4.5.2.5 load_number_template() 25
4.5.2.6 number_recognition() 26
4.5.2.7 save_convex_hull() 26
4.5.2.8 shape_detection() 26
4.5.3 Variable Documentation 27
4.5.3.1 fs_xml 27
4.5.3.2 templates 27
4.5.3.3 xml_settings 27
4.6 src/detection.hh File Reference 27
4.6.1 Function Documentation 28
4.6.1.1 crop_number_section() 28
4.6.1.2 detection() 29
4.6.1.3 erode_dilation() 29
4.6.1.4 find_contours() 29
4.6.1.5 load_number_template() 30
4.6.1.6 number_recognition() 30
4.6.1.7 save_convex_hull() 30
4.6.1.8 shape_detection() 31
4.7 src/detection_run.cc File Reference 31
4.7.1 Function Documentation 31
4.7.1.1 main() 31
4.8 src/main.cc File Reference 32
4.8.1 Function Documentation 32
4.8.1.1 main() 32
4.9 src/unwrapping.cc File Reference 32
4.9.1 Macro Definition Documentation 33
4.9.1.1 area_ratio 33
4.9.1.2 WAIT 33
4.9.2 Function Documentation 33

4.9.2.1 distance() 33
4.9.2.2 loadCoefficients() 34
4.9.2.3 swap() 34
4.9.2.4 unwrapping() 34
4.9.3 Variable Documentation 35
4.9.3.1 xml_settings 35
4.10 src/unwrapping.hh File Reference 35
4.10.1 Function Documentation 36
4.10.1.1 loadCoefficients() 36
4.10.1.2 unwrapping() 36
4.11 src/unwrapping_run.cc File Reference 37
4.11.1 Function Documentation 37
4.11.1.1 main() 37
4.12 src/utils.cc File Reference 37
4.12.1 Function Documentation 37
4.12.1.1 my_imshow() 38
4.13 src/utils.hh File Reference 38
4.13.1 Macro Definition Documentation 38
4.13.1.1 INFO 39
4.13.2 Function Documentation 39
4.13.2.1 my_imshow() 39

Index **41**

Chapter 1

Class Index

1.1 Class List

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

Settings 5

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

src/calibration.cc . . . 15
src/calibration.hh
Library for calibration . . . 19
src/calibration_run.cc . . . 22
src/create_xml.cc . . . 22
src/detection.cc . . . 23
src/detection.hh . . . 27
src/detection_run.cc . . . 31
src/main.cc . . . 32
src/unwrapping.cc . . . 32
src/unwrapping.hh . . . 35
src/unwrapping_run.cc . . . 37
src/utils.cc . . . 37
src/utils.hh . . . 38

Chapter 3

Class Documentation

3.1 Settings Class Reference

```
#include <calibration.hh>
```

Public Types

- enum Pattern { NOT_EXISTING =0, CHESSBOARD =1 }
- enum InputType { INVALID =0, IMAGE_LIST =3 }

Public Member Functions

- Settings ()

  Constructor that sets `goodInput` *to false.*
- void write (FileStorage &fs) const

  Write serialization.
- void read (const FileNode &node)

  Read serialization.
- void validate ()

  This function validate the content of the file.
- Mat nextImage ()

  Get next image from list.

Static Public Member Functions

- static bool readStringList (const string &filename, vector$<$ string $>$ &l)

  Read from file a list of images.
- static bool isListOfImages (const string &filename)

  Check if the file from which is trying to retrive a list is a valid format (xml or yaml).

Public Attributes

- Size boardSize

  The size of the board -> Number of items by width and height.
- Pattern calibrationPattern = CHESSBOARD

  One of the Chessboard, circles, or asymmetric circle pattern.
- float squareSize

  The size of a square in your defined unit (point, millimeter,etc).
- int nrFrames

  The number of frames to use from the input for calibration.
- float aspectRatio

  The aspect ratio.
- int delay

  In case of a video input.
- bool writePoints

  Write detected feature points.
- bool writeExtrinsics

  Write extrinsic parameters.
- bool calibZeroTangentDist

  Assume zero tangential distortion.
- bool calibFixPrincipalPoint

  Fix the principal point at the center.
- bool flipVertical

  Flip the captured images around the horizontal axis.
- string outputFileName

  The name of the file where to write.
- bool showUndistorsed

  Show undistorted images after calibration.
- string input

  The input.
- bool useFisheye = false

  use fisheye camera model for calibration
- bool fixK1

  fix K1 distortion coefficient
- bool fixK2

  fix K2 distortion coefficient
- bool fixK3

  fix K3 distortion coefficient
- bool fixK4

  fix K4 distortion coefficient
- bool fixK5

  fix K5 distortion coefficient
- int cameraID
- vector< string > imageList
- size_t atImageList
- VideoCapture inputCapture
- InputType inputType = IMAGE_LIST
- bool goodInput
- int flag

3.1.1 Member Enumeration Documentation

3.1.1.1 InputType

```
enum Settings::InputType
```

Enumerator

INVALID	
IMAGE_LIST	

3.1.1.2 Pattern

```
enum Settings::Pattern
```

Enumerator

NOT_EXISTING	
CHESSBOARD	

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Settings()

```
Settings::Settings ( )  [inline]
```

Constructor that sets `goodInput` to false.

3.1.3 Member Function Documentation

3.1.3.1 isListOfImages()

```
bool Settings::isListOfImages (
             const string & filename )  [static]
```

Check if the file from which is trying to retrive a list is a valid format (xml or yaml).

Parameters

in	*filename*	The name of the file to check for validity.

Returns

`false` is the file is not xml or yaml
`true` otherwise.

3.1.3.2 nextImage()

```
Mat Settings::nextImage ( )
```

Get next image from list.

Returns

A matrix containing the next image to consider.

3.1.3.3 read()

```
void Settings::read (
             const FileNode & node )
```

Read serialization.

This function read data from a file and stores each node in their corresponding variables.

Parameters

in	*node*	The node of the file to consider.

3.1.3.4 readStringList()

```
bool Settings::readStringList (
             const string & filename,
             vector< string > & l )  [static]
```

Read from file a list of images.

Parameters

in	*filename*	The name of the file from which to read.
out	*l*	A vector which will contain the names of the file from the list.

Returns

`false` if the file could not be opened or if the file doesn't contain a list
`true` otherwise.

3.1.3.5 validate()

```
void Settings::validate ( )
```

This function validate the content of the file.

Even though this function doesn't return anything nor has any parameters for output, it sets a variable of the `Settings` class, that is `googInput`, to `false` if some infos were wrong. `true` otherwise. The options it takes in consideration are the following:

- Size must be positive.

- Cells must be greater than 10^{-6}.
- The number of frames considered, that is images, must be greater than 0.
- Check for valid input, that is a valid list of images.
- Else a list of image is being used.
- Check the field pattern: if it doesn't correspond to a known one than it's invalid.

3.1.3.6 write()

```
void Settings::write (
            FileStorage & fs ) const
```

Write serialization.

This function write data to a file.

Parameters

`in`	*fs*	The filename where to write.

3.1.4 Member Data Documentation

3.1.4.1 aspectRatio

```
float Settings::aspectRatio
```

The aspect ratio.

3.1.4.2 atImageList

```
size_t Settings::atImageList
```

3.1.4.3 boardSize

```
Size Settings::boardSize
```

The size of the board -> Number of items by width and height.

3.1.4.4 calibFixPrincipalPoint

```
bool Settings::calibFixPrincipalPoint
```

Fix the principal point at the center.

3.1.4.5 calibrationPattern

```
Pattern Settings::calibrationPattern = CHESSBOARD
```

One of the Chessboard, circles, or asymmetric circle pattern.

3.1.4.6 calibZeroTangentDist

```
bool Settings::calibZeroTangentDist
```

Assume zero tangential distortion.

3.1.4.7 cameraID

```
int Settings::cameraID
```

3.1.4.8 delay

```
int Settings::delay
```

In case of a video input.

3.1.4.9 fixK1

```
bool Settings::fixK1
```

fix K1 distortion coefficient

3.1.4.10 fixK2

```
bool Settings::fixK2
```

fix K2 distortion coefficient

3.1.4.11 fixK3

```
bool Settings::fixK3
```

fix K3 distortion coefficient

3.1.4.12 fixK4

```
bool Settings::fixK4
```

fix K4 distortion coefficient

3.1.4.13 fixK5

```
bool Settings::fixK5
```

fix K5 distortion coefficient

3.1.4.14 flag

```
int Settings::flag
```

3.1.4.15 flipVertical

```
bool Settings::flipVertical
```

Flip the captured images around the horizontal axis.

3.1.4.16 goodInput

```
bool Settings::goodInput
```

3.1.4.17 imageList

```
vector<string> Settings::imageList
```

3.1.4.18 input

```
string Settings::input
```

The input.

3.1.4.19 inputCapture

```
VideoCapture Settings::inputCapture
```

3.1.4.20 inputType

```
InputType Settings::inputType = IMAGE_LIST
```

3.1.4.21 nrFrames

```
int Settings::nrFrames
```

The number of frames to use from the input for calibration.

3.1.4.22 outputFileName

```
string Settings::outputFileName
```

The name of the file where to write.

3.1.4.23 showUndistorsed

```
bool Settings::showUndistorsed
```

Show undistorted images after calibration.

3.1.4.24 squareSize

```
float Settings::squareSize
```

The size of a square in your defined unit (point, millimeter,etc).

3.1.4.25 useFisheye

```
bool Settings::useFisheye = false
```

use fisheye camera model for calibration

3.1.4.26 writeExtrinsics

```
bool Settings::writeExtrinsics
```

Write extrinsic parameters.

3.1.4.27 writePoints

```
bool Settings::writePoints
```

Write detected feature points.

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

- src/calibration.hh
- src/calibration.cc

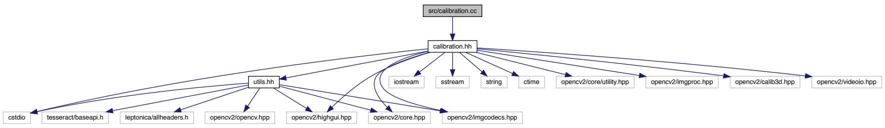
Chapter 4

File Documentation

4.1 src/calibration.cc File Reference

```
#include "calibration.hh"
```

Include dependency graph for calibration.cc:



Functions

- int calibration (const string inputFile)

  Function to run the complete calibration.
- static void read (const FileNode &node, Settings &x, const Settings &default_value)

  Reads settings from file. If there is none then initiate a new `Settings`.
- static double computeReprojectionErrors (const vector< vector< Point3f > > &objectPoints, const vector< vector< Point2f > > &imagePoints, const vector< Mat > &rvecs, const vector< Mat > &tvecs, const Mat &cameraMatrix, const Mat &distCoeffs, vector< float > &perViewErrors, bool fisheye)

  Compute the errors of the projection.
- void calcBoardCornerPositions (Size boardSize, float squareSize, vector< Point3f > &corners)

  This function compute the position of the upper corners of every cell.
- static bool runCalibration (Settings &s, Size &imageSize, Mat &cameraMatrix, Mat &distCoeffs, vector< vector< Point2f > > imagePoints, vector< Mat > &rvecs, vector< Mat > &tvecs, vector< float > &reprojErrs, double &totalAvgErr)

  This function run the calibration creating the matrixed for the camera and the distorsion coefficients.
- static void saveCameraParams (const Settings &s, const Size &imageSize, const Mat &cameraMatrix, const Mat &distCoeffs, const vector< Mat > &rvecs, const vector< Mat > &tvecs, const vector< float > &reprojErrs, const vector< vector< Point2f > > &imagePoints, const double totalAvgErr)

  Function to save the computed parameters to a file.
- bool runCalibrationAndSave (Settings &s, Size imageSize, Mat &cameraMatrix, Mat &distCoeffs, vector< vector< Point2f > > imagePoints)

  Reads settings from file. If there is none then initiate a new `Settings`.

4.1.1 Function Documentation

4.1.1.1 calcBoardCornerPositions()

```
void calcBoardCornerPositions (
             Size boardSize,
             float squareSize,
             vector< Point3f > & corners )
```

This function compute the position of the upper corners of every cell.

Parameters

in	*boardSiz*	The dimension of the chess board.
in	*squareSize*	The dimension of the edge of a cell.
out	*corners*	A vector of Point3fs which equals to the corners of the cells.

4.1.1.2 calibration()

```
int calibration (
             const string inputFile )
```

Function to run the complete calibration.

Parameters

in	*inputFile*	Name of the setting.xml file. It's set to default to default.xml

Returns

-2 if the settings file could be load but the input was not well-formed

-1 if the settings file could not be opened.

0 if everything went fine.

4.1.1.3 computeReprojectionErrors()

```
static double computeReprojectionErrors (
             const vector< vector< Point3f > > & objectPoints,
             const vector< vector< Point2f > > & imagePoints,
             const vector< Mat > & rvecs,
             const vector< Mat > & tvecs,
             const Mat & cameraMatrix,
             const Mat & distCoeffs,
             vector< float > & perViewErrors,
             bool fisheye )  [static]
```

Compute the errors of the projection.

Parameters

in	*objectPoints*	The real image points which will be projected
in	*rvecs*	Input vector of rotation vectors estimated for each pattern view.
in	*tvecs*	Input vector of translation vectors estimated for each pattern view.
in	*cameraMatrix*	The matrix containing the parameters for the camera
in	*distCoeffs*	The matrix containing the distortion coefficients.
in	*fisheye*	A variable which says if a fish eye correction should be applied or no.
out	*perViewErrors*	A vector containing the error for each image.
out	*imagePoints*	The projected points for each image.

Returns

The total error.

4.1.1.4 read()

```
static void read (
            const FileNode & node,
            Settings & x,
            const Settings & default_value )  [inline], [static]
```

Reads settings from file. If there is none then initiate a new `Settings`.

Parameters

in	*node*	node to consider for getting settings;
in	*x*	`Settings` to configure;
in	*default_value*	`Settings` default value. Setted to `Settings()`.

4.1.1.5 runCalibration()

```
static bool runCalibration (
            Settings & s,
            Size & imageSize,
            Mat & cameraMatrix,
            Mat & distCoeffs,
            vector< vector< Point2f > > imagePoints,
            vector< Mat > & rvecs,
            vector< Mat > & tvecs,
            vector< float > & reprojErrs,
            double & totalAvgErr )  [static]
```

This function run the calibration creating the matrixed for the camera and the distorsion coefficients.

Parameters

`in`	*s*	The `Settings` read from the file and memorized.
`in`	*imageSize*	The size of the image used in `calibrateCamera()` to initialize the camera matrix.
`in`	*imagePoints*	The projected points for each image.
`in`	*reprojErrs*	The re-projection error, that is a geometric error corresponding to the image distance between a projected point and a measured one.
`out`	*cameraMatrix*	The matrix of the camera parameters
`out`	*distCoeffs*	The matrix of the distorsion coefficients.
`out`	*rvecs*	Output vector of rotation vectors estimated for each pattern view.
`out`	*tvecs*	Output vector of translation vectors estimated for each pattern view.
`out`	*totalAvgErr*	The total avarage error given from distorsion.

Returns

`false` if one or more elements in the `cameraMatrix` and `distCoeffs` are invalid.
`true` if all the elements are valid.

4.1.1.6 runCalibrationAndSave()

```
bool runCalibrationAndSave (
            Settings & s,
            Size imageSize,
            Mat & cameraMatrix,
            Mat & distCoeffs,
            vector< vector< Point2f > > imagePoints )
```

Reads settings from file. If there is none then initiate a new `Settings`.

Parameters

`in`	*s*	The `Settings` being used during the execution.
`in`	*imageSize*	The dimensions of the images.
`in`	*imagePoints*	The projected points for each image.
`out`	*cameraMatrix*	The matrix which is used to store the values for the camera parameters.
`out`	*distCoeffs*	The matrix which is used to store the distortion coefficients.

Returns

`true` if the calibration succeded.
`false` otherwise.

4.1.1.7 saveCameraParams()

```
static void saveCameraParams (
            const Settings & s,
            const Size & imageSize,
            const Mat & cameraMatrix,
            const Mat & distCoeffs,
            const vector< Mat > & rvecs,
            const vector< Mat > & tvecs,
            const vector< float > & reprojErrs,
            const vector< vector< Point2f > > & imagePoints,
            const double totalAvgErr )  [static]
```

Function to save the computed parameters to a file.

Parameters

in	*s*	Use the Settings got at the beginning for information as the output file name, image and board size.
in	*imageSize*	The size of the imgage.
in	*cameraMatrix*	The camera matrix.
in	*distCoeffs*	The distorsion coefficient matrix.
	[int]	rvecs Vector of rotation vectors estimated for each pattern view.
in	*tvecs*	Vector of translation vectors estimated for each pattern view.
in	*reprojErrs*	The re-projection error, that is a geometric error corresponding to the image distance between a projected point and a measured one.
in	*imagePoints*	The projected points for each image.
in	*totalAvgErr*	The total avarage error given from distorsion.

Open file for writing

Stores time of calibration

Store infos about the images

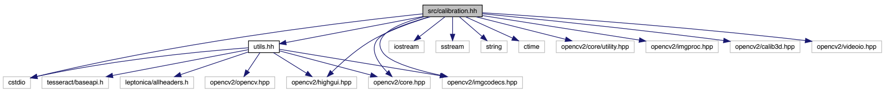
4.2 src/calibration.hh File Reference

Library for calibration.

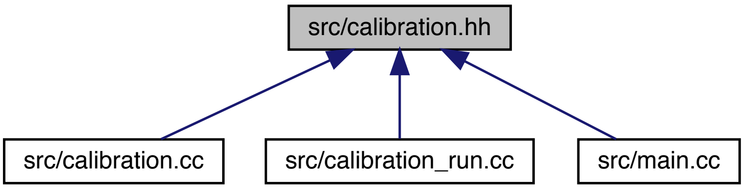
```
#include "utils.hh"
#include <iostream>
#include <sstream>
#include <string>
#include <ctime>
#include <cstdio>
#include <opencv2/core.hpp>
#include <opencv2/core/utility.hpp>
#include <opencv2/imgproc.hpp>
#include <opencv2/calib3d.hpp>
#include <opencv2/imgcodecs.hpp>
#include <opencv2/videoio.hpp>
```

```
#include <opencv2/highgui.hpp>
```

Include dependency graph for calibration.hh:



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



Classes

- class Settings

Enumerations

- enum { DETECTION = 0, CAPTURING = 1, CALIBRATED = 2 }

Functions

- int calibration (const string inputFile="data/calib_config.xml")

  Function to run the complete calibration.

- bool runCalibrationAndSave (Settings &s, Size imageSize, Mat &cameraMatrix, Mat &distCoeffs, vector< vector< Point2f > > imagePoints)

  Reads settings from file. If there is none then initiate a new `Settings`.

4.2.1 Detailed Description

Library for calibration.

4.2.2 Enumeration Type Documentation

4.2.2.1 anonymous enum

```
anonymous enum
```

Enumerator

DETECTION	
CAPTURING	
CALIBRATED	

4.2.3 Function Documentation

4.2.3.1 calibration()

```
int calibration (
             const string inputFile )
```

Function to run the complete calibration.

Parameters

in	*inputFile*	Name of the setting.xml file. It's set to default to default.xml

Returns

-2 if the settings file could be load but the input was not well-formed
-1 if the settings file could not be opened.
0 if everything went fine.

4.2.3.2 runCalibrationAndSave()

```
bool runCalibrationAndSave (
             Settings & s,
             Size imageSize,
             Mat & cameraMatrix,
             Mat & distCoeffs,
             vector< vector< Point2f > > imagePoints )
```

Reads settings from file. If there is none then initiate a new `Settings`.

Parameters

in	*s*	The `Settings` being used during the execution.
in	*imageSize*	The dimensions of the images.
in	*imagePoints*	The projected points for each image.
out	*cameraMatrix*	The matrix which is used to store the values for the camera parameters.
out	*distCoeffs*	The matrix which is used to store the distortion coefficients.

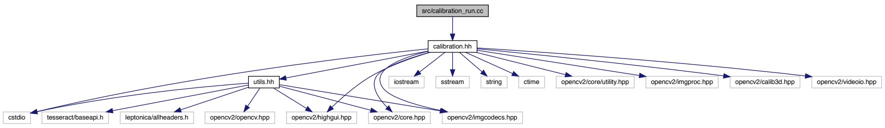
Returns

`true` if the calibration succeded.
`false` otherwise.

4.3 src/calibration_run.cc File Reference

```
#include "calibration.hh"
```

Include dependency graph for calibration_run.cc:



Functions

- int main ()

4.3.1 Function Documentation

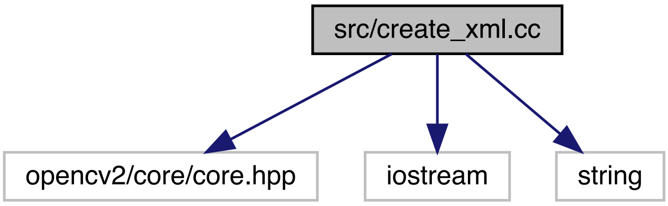
4.3.1.1 main()

```
int main ( )
```

4.4 src/create_xml.cc File Reference

```
#include <opencv2/core/core.hpp>
#include <iostream>
#include <string>
```

Include dependency graph for create_xml.cc:



Functions

- int main ()

4.4.1 Function Documentation

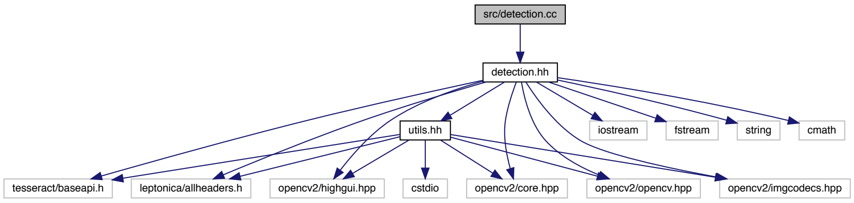
4.4.1.1 main()

```
int main ( )
```

4.5 src/detection.cc File Reference

```
#include "detection.hh"
```

Include dependency graph for detection.cc:



Macros

- #define WAIT

Functions

- int detection ()

  Loads some images and detects shapes according to different colors.
- void load_number_template ()

  Load some templates and save them in the global variable 'templates'.
- void shape_detection (const Mat &img, const int color)

  Detect shapes inside the image according to the variable 'color'.
- void erode_dilation (Mat &img, const int color)

  It apply some filtering function for isolate the subject and remove the noise.
- void find_contours (const Mat &img, Mat original, const int color)

  Given an image, in black/white format, identify all the borders that delimit the shapes.
- void save_convex_hull (const vector< vector< Point >> &contours, const int color, const vector< int > &victims)

  Given some vector save it in a xml file.
- int number_recognition (Rect blob, const Mat &base)

  Detect a number on an image inside a region of interest.
- void crop_number_section (Mat &ROI)

  Given an image identify the region of interest(ROI) and crop it out.

Variables

- const string xml_settings = "data/settings.xml"
- FileStorage fs_xml
- vector< Mat > templates

4.5.1 Macro Definition Documentation

4.5.1.1 WAIT

```
#define WAIT
```

4.5.2 Function Documentation

4.5.2.1 crop_number_section()

```
void crop_number_section (
             Mat & ROI )
```

Given an image identify the region of interest(ROI) and crop it out.

Parameters

in,out	*ROI*	Is the image that the function will going to elaborate.

4.5.2.2 detection()

```
int detection ( )
```

Loads some images and detects shapes according to different colors.

Returns

Return 0 if the function reach the end.

4.5.2.3 erode_dilation()

```
void erode_dilation (
            Mat & img,
            const int color )
```

It apply some filtering function for isolate the subject and remove the noise.

An example of the sub functions called are: GaussianBlur, Erosion, Dilation and Threshold.

Parameters

in,out	*img*	Is the image on which the function apply the filtering.
in	*color*	Can has 4 value: 0 -> Red 1 -> Green 2 -> Blue 3 -> Black According to the color the filtering functions apply can change in the type and in the order.

4.5.2.4 find_contours()

```
void find_contours (
            const Mat & img,
            Mat original,
            const int color )
```

Given an image, in black/white format, identify all the borders that delimit the shapes.

Parameters

in	*img*	Is an image in HSV format at the base of the elaboration process.
out	*original*	Is the original source of 'img', it is used for showing the detected contours.
in	*color*	Can has 3 value: 0 -> Red 1 -> Green 2 -> Blue Is used for decid which procedure apply to the image.

4.5.2.5 load_number_template()

```
void load_number_template ( )
```

Load some templates and save them in the global variable 'templates'.

4.5.2.6 number_recognition()

```
int number_recognition (
            Rect blob,
            const Mat & base )
```

Detect a number on an image inside a region of interest.

Parameters

in	*blob*	Identify the region of interest inside the image 'base'.
in	*base*	Is the image where the function will going to search the number.

Returns

The number recognise, '-1' otherwise.

4.5.2.7 save_convex_hull()

```
void save_convex_hull (
            const vector< vector< Point >> & contours,
            const int color,
            const vector< int > & victims )
```

Given some vector save it in a xml file.

Parameters

in	*contours*	Is a vector that is saved in a xml file.
in	*color*	Is the parameter according to which the function decide if saved ('color==1') or not ('otherwise') the vector 'victims'.
in	*victims*	Is a vector that is saved in a xml file.

4.5.2.8 shape_detection()

```
void shape_detection (
            const Mat & img,
            const int color )
```

Detect shapes inside the image according to the variable 'color'.

Parameters

in	*img*	Image on which the research will done.
in	*color*	Can has 3 value: 0 -> Red 1 -> Green 2 -> Blue These color identify the possible spectrum that the function search on the image.

4.5.3 Variable Documentation

4.5.3.1 fs_xml

```
FileStorage fs_xml
```

4.5.3.2 templates

```
vector<Mat> templates
```

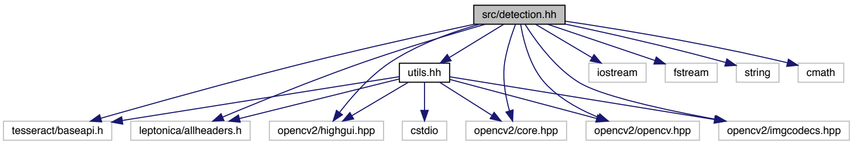
4.5.3.3 xml_settings

```
const string xml_settings = "data/settings.xml"
```

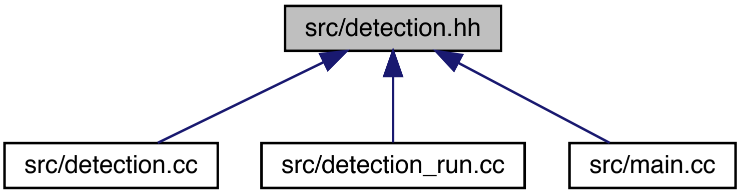
4.6 src/detection.hh File Reference

```
#include <tesseract/baseapi.h>
#include <leptonica/allheaders.h>
#include "utils.hh"
#include <iostream>
#include <fstream>
#include <string>
#include <cmath>
#include <opencv2/highgui.hpp>
#include <opencv2/core.hpp>
#include <opencv2/opencv.hpp>
#include <opencv2/imgcodecs.hpp>
```

Include dependency graph for detection.hh:



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



Functions

- int detection ()

  Loads some images and detects shapes according to different colors.
- void shape_detection (const Mat &img, const int color)

  Detect shapes inside the image according to the variable 'color'.
- void erode_dilation (Mat &img, const int color)

  It apply some filtering function for isolate the subject and remove the noise.
- void find_contours (const Mat &img, Mat original, const int color)

  Given an image, in black/white format, identify all the borders that delimit the shapes.
- int number_recognition (Rect blob, const Mat &base)

  Detect a number on an image inside a region of interest.
- void save_convex_hull (const vector< vector< Point >> &contours, const int color, const vector< int > &victims)

  Given some vector save it in a xml file.
- void load_number_template ()

  Load some templates and save them in the global variable 'templates'.
- void crop_number_section (Mat &processROI)

  Given an image identify the region of interest(ROI) and crop it out.

4.6.1 Function Documentation

4.6.1.1 crop_number_section()

```
void crop_number_section (
            Mat & ROI )
```

Given an image identify the region of interest(ROI) and crop it out.

Parameters

in,out	*ROI*	Is the image that the function will going to elaborate.

4.6.1.2 detection()

```
int detection ( )
```

Loads some images and detects shapes according to different colors.

Returns

Return 0 if the function reach the end.

4.6.1.3 erode_dilation()

```
void erode_dilation (
            Mat & img,
            const int color )
```

It apply some filtering function for isolate the subject and remove the noise.

An example of the sub functions called are: GaussianBlur, Erosion, Dilation and Threshold.

Parameters

in,out	*img*	Is the image on which the function apply the filtering.
in	*color*	Can has 4 value: 0 -> Red 1 -> Green 2 -> Blue 3 -> Black According to the color the filtering functions apply can change in the type and in the order.

4.6.1.4 find_contours()

```
void find_contours (
            const Mat & img,
            Mat original,
            const int color )
```

Given an image, in black/white format, identify all the borders that delimit the shapes.

Parameters

in	*img*	Is an image in HSV format at the base of the elaboration process.
out	*original*	Is the original source of 'img', it is used for showing the detected contours.

Parameters

in	*color*	Can has 3 value: 0 -> Red 1 -> Green 2 -> Blue Is used for decid which procedure apply to the image.

4.6.1.5 load_number_template()

```
void load_number_template ( )
```

Load some templates and save them in the global variable 'templates'.

4.6.1.6 number_recognition()

```
int number_recognition (
            Rect blob,
            const Mat & base )
```

Detect a number on an image inside a region of interest.

Parameters

in	*blob*	Identify the region of interest inside the image 'base'.
in	*base*	Is the image where the function will going to search the number.

Returns

The number recognise, '-1' otherwise.

4.6.1.7 save_convex_hull()

```
void save_convex_hull (
            const vector< vector< Point >> & contours,
            const int color,
            const vector< int > & victims )
```

Given some vector save it in a xml file.

Parameters

in	*contours*	Is a vector that is saved in a xml file.
in	*color*	Is the parameter according to which the function decide if saved ('color==1') or not ('otherwise') the vector 'victims'.
in	*victims*	Is a vector that is saved in a xml file.

4.6.1.8 shape_detection()

```
void shape_detection (
            const Mat & img,
            const int color )
```

Detect shapes inside the image according to the variable 'color'.

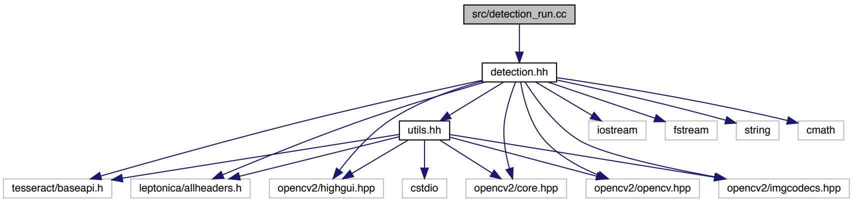
Parameters

in	*img*	Image on which the research will done.
in	*color*	Can has 3 value: 0 -> Red 1 -> Green 2 -> Blue These color identify the possible spectrum that the function search on the image.

4.7 src/detection_run.cc File Reference

```
#include "detection.hh"
```

Include dependency graph for detection_run.cc:



Functions

- int main ()

4.7.1 Function Documentation

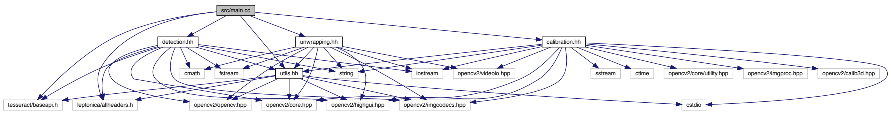
4.7.1.1 main()

```
int main ( )
```

4.8 src/main.cc File Reference

```
#include <tesseract/baseapi.h>
#include <leptonica/allheaders.h>
#include "utils.hh"
#include "detection.hh"
#include "unwrapping.hh"
#include "calibration.hh"
```

Include dependency graph for main.cc:



Functions

- int main ()

4.8.1 Function Documentation

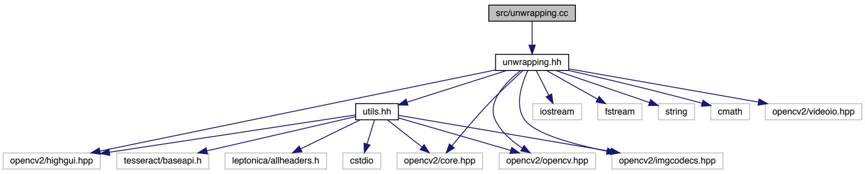
4.8.1.1 main()

```
int main ( )
```

4.9 src/unwrapping.cc File Reference

```
#include "unwrapping.hh"
```

Include dependency graph for unwrapping.cc:



Macros

- #define WAIT
- #define area_ratio 0.7

Functions

- static float distance (Point c1, Point c2)

  Compute the euclidean distance.
- static void swap (int &a, int &b)

  Swap the two integers passed.
- int unwrapping ()

  Take some images according to a xml and unwrap the black rectangle inside the image after appling undistortion trasformation.
- void loadCoefficients (const string filename, Mat &camera_matrix, Mat &dist_coeffs)

  Load coefficients from a file.

Variables

- const string xml_settings = "data/settings.xml"

4.9.1 Macro Definition Documentation

4.9.1.1 area_ratio

```
#define area_ratio 0.7
```

4.9.1.2 WAIT

```
#define WAIT
```

4.9.2 Function Documentation

4.9.2.1 distance()

```
static float distance (
            Point c1,
            Point c2 )  [static]
```

Compute the euclidean distance.

Parameters

in,out	*c1*	The first point.
in,out	*c2*	The second point.

Returns

The euclidean distance.

4.9.2.2 loadCoefficients()

```
void loadCoefficients (
            const string filename,
            Mat & camera_matrix,
            Mat & dist_coeffs )
```

Load coefficients from a file.

Load two matrix 'camera_matrix' and 'distortion_coefficients' from the xml file passed.

Parameters

in	*filename*	The string that identify the location of the xml file.
out	*camera_matrix*	Where the 'camera_matrix' matrix is saved.
out	*dist_coeffs*	Where the 'distortion_coefficients' matrix is saved.

4.9.2.3 swap()

```
static void swap (
            int & a,
            int & b )  [static]
```

Swap the two integers passed.

Parameters

in,out	*a*	First parameter.
in,out	*b*	Second parameter.

4.9.2.4 unwrapping()

```
int unwrapping ( )
```

Take some images according to a xml and unwrap the black rectangle inside the image after appling undistortion trasformation.

Load from the xml file 'data/settings.xml' the name of some images, load the images from the file,
apply the calibration (undistortion trasformation) thanks to the matrices load with the 'loadCoefficients' function.
Then, with the use of a filter for the black the region of interest (a rectangle) is identified and all the perspective is rotated for reach a top view of the rectangle.
Finally, the images are saved on some files.

Returns

A 0 is return if the function reach the end.

4.9.3 Variable Documentation

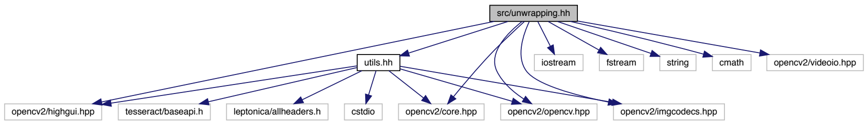
4.9.3.1 xml_settings

```
const string xml_settings = "data/settings.xml"
```

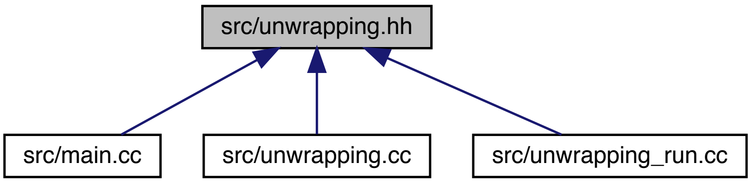
4.10 src/unwrapping.hh File Reference

```
#include "utils.hh"
#include <iostream>
#include <fstream>
#include <string>
#include <cmath>
#include <opencv2/videoio.hpp>
#include <opencv2/highgui.hpp>
#include <opencv2/core.hpp>
#include <opencv2/opencv.hpp>
#include <opencv2/imgcodecs.hpp>
```

Include dependency graph for unwrapping.hh:



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



Functions

- int unwrapping ()

  Take some images according to a xml and unwrap the black rectangle inside the image after appling undistortion trasformation.
- void loadCoefficients (const string filename, Mat &camera_matrix, Mat &dist_coeffs)

  Load coefficients from a file.

4.10.1 Function Documentation

4.10.1.1 loadCoefficients()

```
void loadCoefficients (
             const string filename,
             Mat & camera_matrix,
             Mat & dist_coeffs )
```

Load coefficients from a file.

Load two matrix 'camera_matrix' and 'distortion_coefficients' from the xml file passed.

Parameters

in	*filename*	The string that identify the location of the xml file.
out	*camera_matrix*	Where the 'camera_matrix' matrix is saved.
out	*dist_coeffs*	Where the 'distortion_coefficients' matrix is saved.

4.10.1.2 unwrapping()

```
int unwrapping ( )
```

Take some images according to a xml and unwrap the black rectangle inside the image after appling undistortion trasformation.

Load from the xml file 'data/settings.xml' the name of some images, load the images from the file,
apply the calibration (undistortion trasformation) thanks to the matrices load with the 'loadCoefficients' function.
Then, with the use of a filter for the black the region of interest (a rectangle) is identified and all the perspective is rotated for reach a top view of the rectangle.
Finally, the images are saved on some files.

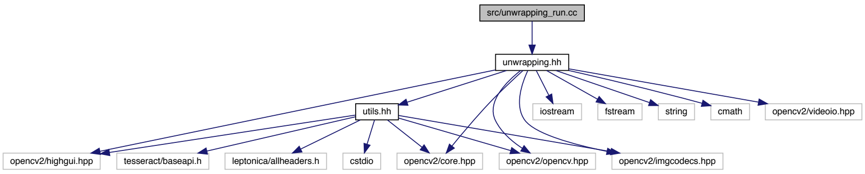
Returns

A 0 is return if the function reach the end.

4.11 src/unwrapping_run.cc File Reference

```
#include "unwrapping.hh"
```

Include dependency graph for unwrapping_run.cc:



Functions

- int main ()

4.11.1 Function Documentation

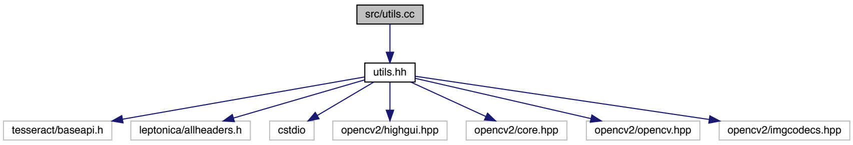
4.11.1.1 main()

```
int main ( )
```

4.12 src/utils.cc File Reference

```
#include "utils.hh"
```

Include dependency graph for utils.cc:



Functions

- void my_imshow (const char ∗win_name, Mat img, bool reset)

4.12.1 Function Documentation

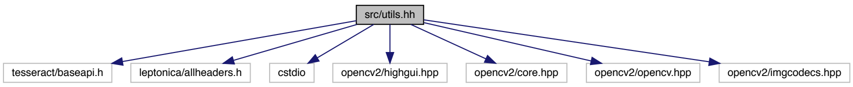
4.12.1.1 my_imshow()

```
void my_imshow (
            const char * win_name,
            Mat img,
            bool reset )
```

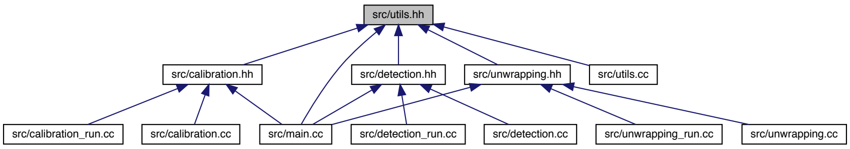
4.13 src/utils.hh File Reference

```
#include <tesseract/baseapi.h>
#include <leptonica/allheaders.h>
#include <cstdio>
#include <opencv2/highgui.hpp>
#include <opencv2/core.hpp>
#include <opencv2/opencv.hpp>
#include <opencv2/imgcodecs.hpp>
```

Include dependency graph for utils.hh:



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



Macros

- #define INFO(msg)

Functions

- void my_imshow (const char *win_name, Mat img, bool reset=false)

4.13.1 Macro Definition Documentation

4.13.1.1 INFO

```
#define INFO(
              msg )
```

4.13.2 Function Documentation

4.13.2.1 my_imshow()

```
void my_imshow (
             const char * win_name,
             Mat img,
             bool reset = false )
```

Index

area_ratio
    unwrapping.cc, 33
aspectRatio
    Settings, 10
atImageList
    Settings, 10

boardSize
    Settings, 10

calcBoardCornerPositions
    calibration.cc, 16
calibFixPrincipalPoint
    Settings, 10
calibZeroTangentDist
    Settings, 10
calibration
    calibration.cc, 16
    calibration.hh, 21
calibration.cc
    calcBoardCornerPositions, 16
    calibration, 16
    computeReprojectionErrors, 16
    read, 17
    runCalibration, 17
    runCalibrationAndSave, 18
    saveCameraParams, 18
calibration.hh
    calibration, 21
    runCalibrationAndSave, 21
calibration_run.cc
    main, 22
calibrationPattern
    Settings, 10
cameraID
    Settings, 11
computeReprojectionErrors
    calibration.cc, 16
create_xml.cc
    main, 23
crop_number_section
    detection.cc, 24
    detection.hh, 28

delay
    Settings, 11
detection
    detection.cc, 24
    detection.hh, 29
detection.cc
    crop_number_section, 24
    detection, 24
    erode_dilation, 24
    find_contours, 25
    fs_xml, 27
    load_number_template, 25
    number_recognition, 25
    save_convex_hull, 26
    shape_detection, 26
    templates, 27
    WAIT, 24
    xml_settings, 27
detection.hh
    crop_number_section, 28
    detection, 29
    erode_dilation, 29
    find_contours, 29
    load_number_template, 30
    number_recognition, 30
    save_convex_hull, 30
    shape_detection, 31
detection_run.cc
    main, 31
distance
    unwrapping.cc, 33

erode_dilation
    detection.cc, 24
    detection.hh, 29

find_contours
    detection.cc, 25
    detection.hh, 29
fixK1
    Settings, 11
fixK2
    Settings, 11
fixK3
    Settings, 11
fixK4
    Settings, 11
fixK5
    Settings, 12
flag
    Settings, 12
flipVertical
    Settings, 12
fs_xml
    detection.cc, 27

goodInput
  Settings, 12

INFO
  utils.hh, 38
imageList
  Settings, 12
input
  Settings, 12
inputCapture
  Settings, 12
inputType
  Settings, 13
InputType
  Settings, 7
isListOfImages
  Settings, 7

load_number_template
  detection.cc, 25
  detection.hh, 30
loadCoefficients
  unwrapping.cc, 34
  unwrapping.hh, 36

main
  calibration_run.cc, 22
  create_xml.cc, 23
  detection_run.cc, 31
  main.cc, 32
  unwrapping_run.cc, 37
main.cc
  main, 32
my_imshow
  utils.cc, 37
  utils.hh, 39

nextImage
  Settings, 8
nrFrames
  Settings, 13
number_recognition
  detection.cc, 25
  detection.hh, 30

outputFileName
  Settings, 13

Pattern
  Settings, 7

read
  calibration.cc, 17
  Settings, 8
readStringList
  Settings, 8
runCalibration
  calibration.cc, 17
runCalibrationAndSave
  calibration.cc, 18
  calibration.hh, 21

save_convex_hull
  detection.cc, 26
  detection.hh, 30
saveCameraParams
  calibration.cc, 18
Settings, 5
  aspectRatio, 10
  atImageList, 10
  boardSize, 10
  calibFixPrincipalPoint, 10
  calibZeroTangentDist, 10
  calibrationPattern, 10
  cameraID, 11
  delay, 11
  fixK1, 11
  fixK2, 11
  fixK3, 11
  fixK4, 11
  fixK5, 12
  flag, 12
  flipVertical, 12
  goodInput, 12
  imageList, 12
  input, 12
  inputCapture, 12
  inputType, 13
  InputType , 7
  isListOfImages, 7
  nextImage, 8
  nrFrames, 13
  outputFileName, 13
  Pattern, 7
  read, 8
  readStringList, 8
  Settings, 7
  showUndistorsed, 13
  squareSize, 13
  useFisheye, 13
  validate, 9
  write, 9
  writeExtrinsics, 14
  writePoints, 14
shape_detection
  detection.cc, 26
  detection.hh, 31
showUndistorsed
  Settings, 13
squareSize
  Settings, 13
src/calibration.cc, 15
src/calibration.hh, 19
src/calibration_run.cc, 22
src/create_xml.cc, 22
src/detection.cc, 23
src/detection.hh, 27
src/detection_run.cc, 31
src/main.cc, 32

src/unwrapping.cc, 32
src/unwrapping.hh, 35
src/unwrapping_run.cc, 37
src/utils.cc, 37
src/utils.hh, 38
swap
    unwrapping.cc, 34

templates
    detection.cc, 27

unwrapping
    unwrapping.cc, 34
    unwrapping.hh, 36
unwrapping.cc
    area_ratio, 33
    distance, 33
    loadCoefficients, 34
    swap, 34
    unwrapping, 34
    WAIT, 33
    xml_settings, 35
unwrapping.hh
    loadCoefficients, 36
    unwrapping, 36
unwrapping_run.cc
    main, 37
useFisheye
    Settings, 13
utils.cc
    my_imshow, 37
utils.hh
    INFO, 38
    my_imshow, 39

validate
    Settings, 9

WAIT
    detection.cc, 24
    unwrapping.cc, 33
write
    Settings, 9
writeExtrinsics
    Settings, 14
writePoints
    Settings, 14

xml_settings
    detection.cc, 27
    unwrapping.cc, 35