HOG feature description and visualization library

Generated by Doxygen 1.9.7

Chapter 1

HOG-feature-descriptor

Course work for ¨OOP¨ 2023 class in NUST MISIS

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Class Documentation

4.1 HOGDescriptor Class Reference

Class for calculating the HOG features.

`#include <hogdescriptor.hpp>`

Public Member Functions

- HOGDescriptor ()

  Default constructor for the HOGDescriptor class.
- HOGDescriptor (const size_t blockSize, const size_t cellSize, const size_t stride, const size_t binNumber, const size_t gradType)

  Construct a new HOGDescriptor object.
- HOGDescriptor (const size_t blockSize, const size_t cellSize)
- ~HOGDescriptor ()

  Constructor for the HOGDescriptor class with parameters for the block size and cell size.
- HOGDescriptor & operator= (const HOGDescriptor &rhs)

  Assignment operator for the HOGDescriptor class.
- void HOGplot (float scale, bool imposed)

  Method for plotting the cellsHistogram of HOGDescriptor as a bunch of arrows within the each cell on image.
- void computeHOG (cv::Mat &image)

  Method for computing HOG features.
- std::vector< float > getHOGFeatureVector ()

  Method for getting the HOG feature vector.
- std::vector< std::vector< std::vector< float > > > getCellHistograms ()

  Get the Cell Histograms object.

Static Public Attributes

- static const size_t GRADIENT_SIGNED = 360

  360 degree spread of histogram channels
- static const size_t GRADIENT_UNSIGNED = 180

  180 degree spread of histogram channels

Private Member Functions

- void computeGradientFeatures (cv::Mat &image)

  Function to compute the magnitude and orientation of each pixel in the input image.
- std::vector< std::vector< std::vector< float > > > computeCellHistograms (cv::Mat magnitude, cv::Mat orientation, std::vector< std::vector< std::vector< float > > > &cell_histograms)

  Compute the HOG feature vectors for each cell in the image.
- std::vector< float > cellHistogram (const cv::Mat &cellMagnitude, const cv::Mat &cellOrientation)

  Method to compute the histogram for the given cell.
- void normalizeBlockHistogram (std::vector< float > &block_histogram)

  Function to normalize the HOG feature vectors for each block of cells in the image.
- const std::vector< float > calculateHOGVector (const std::vector< std::vector< std::vector< float > > > &cell_histograms)

  Method to calculate the HOG feature vector.

Private Attributes

- int blockSize_

  Block size for the sliding window in pixels.
- int cellSize_

  Size of the cell in pixels.
- int binNumber_

  Number of the bins in the histogram of each cell.
- int binWidth_

  Width of the bins in the histogram of each cell.
- int stride_

  Sliding window stride in pixels.
- int gradType_

  Type of the gradient calculation (unsigned or signed)
- bool hogFlag_ = false

  Flag to check if the HOG feature vector has been computed.
- cv::Mat imageMagnitude_

  Magnitude of the gradient image.
- cv::Mat imageOrientation_

  Orientation of the gradient image.
- std::vector< std::vector< std::vector< float > > > cellHistograms_

  Vector of cell histograms.
- std::vector< float > hogFeatureVector_

  Final HOG feature vector.

4.1.1 Detailed Description

Class for calculating the HOG features.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 HOGDescriptor() [1/3]

```
HOGDescriptor::HOGDescriptor ( )
```

Default constructor for the HOGDescriptor class.

4.1.2.2 HOGDescriptor() [2/3]

```
HOGDescriptor::HOGDescriptor (
            const size_t blockSize,
            const size_t cellSize,
            const size_t stride,
            const size_t binNumber,
            const size_t gradType )
```

Construct a new HOGDescriptor object.

Parameters

blockSize	Block size for the sliding window in pixels
cellSize	Size of the cell in pixels
stride	Sliding window stride in pixels
binNumber	Number of the bins in the histogram of each cell
gradType	Type of the gradient calculation (unsigned or signed)

Here is the call graph for this function:

4.1.2.3 HOGDescriptor() [3/3]

```
HOGDescriptor::HOGDescriptor (
            const size_t blockSize,
            const size_t cellSize )
```

Here is the call graph for this function:

4.1.2.4 ~HOGDescriptor()

```
HOGDescriptor::~HOGDescriptor ( )
```

Constructor for the HOGDescriptor class with parameters for the block size and cell size.

Destroy the HOGDescriptor object

4.1.3 Member Function Documentation

4.1.3.1 calculateHOGVector()

```
const std::vector< float > HOGDescriptor::calculateHOGVector (
            const std::vector< std::vector< std::vector< float > > > & cell_histograms )
[private]
```

Method to calculate the HOG feature vector.

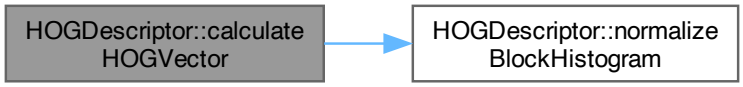
Parameters

cell_histograms	3d vector of cell histograms

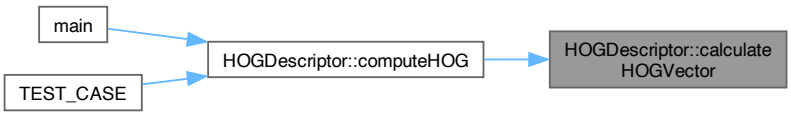
Returns

Final vector

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.3.2 cellHistogram()

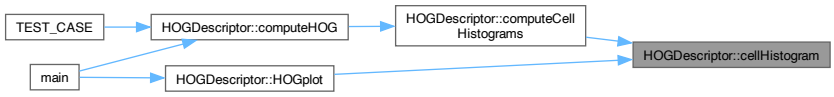
```
std::vector< float > HOGDescriptor::cellHistogram (
            const cv::Mat & cellMagnitude,
            const cv::Mat & cellOrientation )  [private]
```

Method to compute the histogram for the given cell.

Parameters

cellMagnitude	cell magnitude matrix
cellOrientation	cell orientation matrix

Here is the caller graph for this function:



4.1.3.3 computeCellHistograms()

```
std::vector< std::vector< std::vector< float > > > HOGDescriptor::computeCellHistograms (
            cv::Mat magnitude,
            cv::Mat orientation,
            std::vector< std::vector< std::vector< float > > > & cell_histograms ) [private]
```

Compute the HOG feature vectors for each cell in the image.

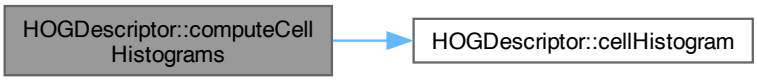
Parameters

gradient	gradient matrix computed by computeGradient()
orientation	orientation matrix computed by computeGradient()
histograms	output vector of HOG feature vectors for each cell

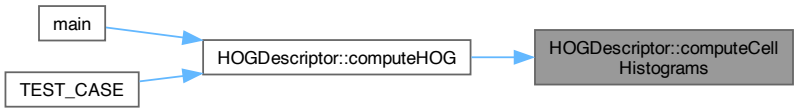
Returns

3d vector of cell histograms

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.3.4 computeGradientFeatures()

```
void HOGDescriptor::computeGradientFeatures (
            cv::Mat & image )  [private]
```

Function to compute the magnitude and orientation of each pixel in the input image.

Parameters

image	input image

Here is the caller graph for this function:

4.1.3.5 computeHOG()

```
void HOGDescriptor::computeHOG (
            cv::Mat & image )
```

Method for computing HOG features.

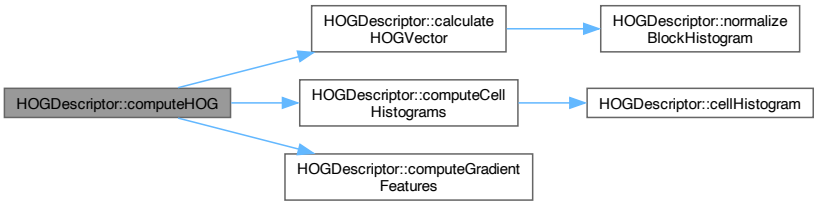
Parameters

image	input image

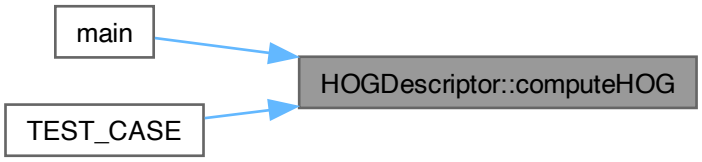
Returns

std::vector<float> - HOG feature vector for the image.

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.3.6 getCellHistograms()

```
std::vector< std::vector< std::vector< float > > > HOGDescriptor::getCellHistograms ( )
```

Get the Cell Histograms object.

Returns

std::vector<std::vector<std::vector<float>>>

Here is the caller graph for this function:

4.1.3.7 getHOGFeatureVector()

```
std::vector< float > HOGDescriptor::getHOGFeatureVector ( )
```

Method for getting the HOG feature vector.

Returns

std::vector<float>

Here is the caller graph for this function:

4.1.3.8 HOGplot()

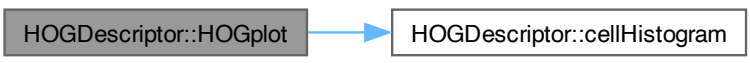
```
void HOGDescriptor::HOGplot (
            float scale,
            bool imposed )
```

Method for plotting the cellsHistogram of HOGDescriptor as a bunch of arrows within the each cell on image.

Parameters

scale	Scale of the arrows
imposed	Background image for reference

Here is the call graph for this function:



Here is the caller graph for this function:

4.1.3.9 normalizeBlockHistogram()

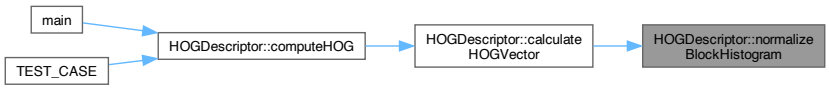
```
void HOGDescriptor::normalizeBlockHistogram (
            std::vector< float > & block_histogram )  [private]
```

Function to normalize the HOG feature vectors for each block of cells in the image.

Parameters

block	vector of histograms representing the cells within a block

Here is the caller graph for this function:



4.1.3.10 operator=()

```
HOGDescriptor & HOGDescriptor::operator= (
            const HOGDescriptor & rhs )
```

Assignment operator for the HOGDescriptor class.

Parameters

rhs	HOGDescriptor object to be copied

Returns

HOGDescriptor&

4.1.4 Member Data Documentation

4.1.4.1 binNumber_

```
int HOGDescriptor::binNumber_  [private]
```

Number of the bins in the histogram of each cell.

4.1.4.2 binWidth_

```
int HOGDescriptor::binWidth_  [private]
```

Width of the bins in the histogram of each cell.

4.1.4.3 blockSize_

```
int HOGDescriptor::blockSize_  [private]
```

Block size for the sliding window in pixels.

4.1.4.4 cellHistograms_

```
std::vector<std::vector<std::vector<float> > > HOGDescriptor::cellHistograms_  [private]
```

Vector of cell histograms.

4.1.4.5 cellSize_

```
int HOGDescriptor::cellSize_  [private]
```

Size of the cell in pixels.

4.1.4.6 GRADIENT_SIGNED

```
const size_t HOGDescriptor::GRADIENT_SIGNED = 360  [static]
```

360 degree spread of histogram channels

4.1.4.7 GRADIENT_UNSIGNED

```
const size_t HOGDescriptor::GRADIENT_UNSIGNED = 180  [static]
```

180 degree spread of histogram channels

4.1.4.8 gradType_

```
int HOGDescriptor::gradType_  [private]
```

Type of the gradient calculation (unsigned or signed)

4.1.4.9 hogFeatureVector_

```
std::vector<float> HOGDescriptor::hogFeatureVector_  [private]
```

Final HOG feature vector.

4.1.4.10 hogFlag_

```
bool HOGDescriptor::hogFlag_ = false  [private]
```

Flag to check if the HOG feature vector has been computed.

4.1.4.11 imageMagnitude_

```
cv::Mat HOGDescriptor::imageMagnitude_  [private]
```

Magnitude of the gradient image.

4.1.4.12 imageOrientation_

```
cv::Mat HOGDescriptor::imageOrientation_  [private]
```

Orientation of the gradient image.

4.1.4.13 stride_

```
int HOGDescriptor::stride_  [private]
```

Sliding window stride in pixels.

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

- src/lib/hogdescriptor/include/hogdescriptor/hogdescriptor.hpp
- src/lib/hogdescriptor/hogdescriptor.cpp

4.2 HOGPlots Class Reference

Class for creating .tex files with plots of HOG.

```
#include <texvisualization.hpp>
```

Public Member Functions

- HOGPlots ()=default

  Construct a new HOGPlots object.
- void HOGgrid (cv::Mat &image, float thickness, int cellSize)

  Method to show the grid of cells on image.
- void cellHistogramPlot (std::vector< float > cellHistogram, int blockWidth, const std::string &executablePath)

  Method for creating a .tex file with the histogram of given cell.

4.2.1 Detailed Description

Class for creating .tex files with plots of HOG.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 HOGPlots()

```
HOGPlots::HOGPlots ( )  [default]
```

Construct a new HOGPlots object.

4.2.3 Member Function Documentation

4.2.3.1 cellHistogramPlot()

```
void HOGPlots::cellHistogramPlot (
            std::vector< float > cellHistogram,
            int blockWidth,
            const std::string & executablePath )
```

Method for creating a .tex file with the histogram of given cell.

Parameters

cellHistogram	vector of cell histogram values
blockWidth	width of the histogram block

Here is the caller graph for this function:

4.2.3.2 HOGgrid()

```
void HOGPlots::HOGgrid (
            cv::Mat & image,
            float thickness,
            int cellSize )
```

Method to show the grid of cells on image.

Parameters

thickness	Grid line thickness
cellSize	Cell size in pixels

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

- src/lib/texvisualization/include/texvisualization/texvisualization.hpp
- src/lib/texvisualization/texvisualization.cpp

Chapter 5

File Documentation

5.1 build/CMakeFiles/3.25.3/CompilerIdCXX/CMakeCXXCompilerId.cpp File Reference

Macros

- #define __has_include(x) 0
- #define COMPILER_ID ""
- #define STRINGIFY_HELPER(X) #X
- #define STRINGIFY(X) STRINGIFY_HELPER(X)
- #define PLATFORM_ID
- #define ARCHITECTURE_ID
- #define DEC(n)
- #define HEX(n)
- #define CXX_STD __cplusplus

Functions

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

Variables

- char const * info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const * info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const * info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char * info_language_standard_default
- const char * info_language_extensions_default

5.1.1 Macro Definition Documentation

5.1.1.1 __has_include

```
#define __has_include(
              x ) 0
```

5.1.1.2 ARCHITECTURE_ID

```
#define ARCHITECTURE_ID
```

5.1.1.3 COMPILER_ID

```
#define COMPILER_ID ...
```

5.1.1.4 CXX_STD

```
#define CXX_STD __cplusplus
```

5.1.1.5 DEC

```
#define DEC(
              n )
```

Value:

```
  ('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))
```

5.1.1.6 HEX

```
#define HEX(
              n )
```

Value:

```
  ('0' + ((n)>>28 & 0xF)), \
  ('0' + ((n)>>24 & 0xF)), \
  ('0' + ((n)>>20 & 0xF)), \
  ('0' + ((n)>>16 & 0xF)), \
  ('0' + ((n)>>12 & 0xF)), \
  ('0' + ((n)>>8  & 0xF)), \
  ('0' + ((n)>>4  & 0xF)), \
  ('0' + ((n)     & 0xF))
```

5.1.1.7 PLATFORM_ID

```
#define PLATFORM_ID
```

5.1.1.8 STRINGIFY

```
#define STRINGIFY(
              X ) STRINGIFY_HELPER(X)
```

5.1.1.9 STRINGIFY_HELPER

```
#define STRINGIFY_HELPER(
              X ) #X
```

5.1.2 Function Documentation

5.1.2.1 main()

```
int main (
             int argc,
             char * argv[] )
```

5.1.3 Variable Documentation

5.1.3.1 info_arch

```
char const* info_arch = ¨INFO¨ ¨:¨ ¨arch[¨ ARCHITECTURE_ID ¨]¨
```

5.1.3.2 info_compiler

```
char const* info_compiler = ¨INFO¨ ¨:¨ ¨compiler[¨ COMPILER_ID ¨]¨
```

5.1.3.3 info_language_extensions_default

```
const char* info_language_extensions_default
```

Initial value:

```
= "INFO" ":" "extensions_default["



  "OFF"

"]"
```

5.1.3.4 info_language_standard_default

```
const char* info_language_standard_default
```

Initial value:

```
= "INFO" ":" "standard_default["









  "98"

"]"
```

5.1.3.5 info_platform

```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

5.2 build/src/example/CMakeFiles/example.dir/main.cpp.o.d File Reference

5.3 build/src/lib/hogdescriptor/CMake↩ Files/hogdescriptor.dir/hogdescriptor.cpp.o.d File Reference

5.4 build/src/lib/texvisualization/CMake↩ Files/texvisualization.dir/texvisualization.cpp.o.d File Reference

5.5 build/src/tests/CMakeFiles/tests.dir/hogtest.cpp.o.d File Reference

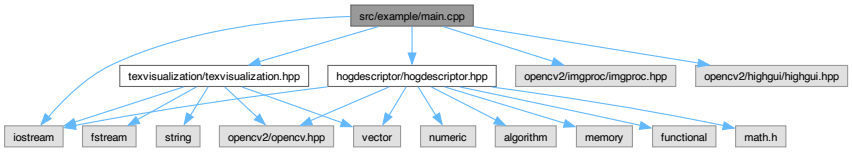
5.6 README.md File Reference

5.7 src/example/main.cpp File Reference

Example usage of HOG descriptor library.

```
#include <hogdescriptor/hogdescriptor.hpp>
#include <texvisualization/texvisualization.hpp>
#include "opencv2/imgproc/imgproc.hpp"
#include "opencv2/highgui/highgui.hpp"
#include <iostream>
```

Include dependency graph for main.cpp:



Functions

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

5.7.1 Detailed Description

Example usage of HOG descriptor library.

Author

Andrey Kadomtsev (m2204942@edu.misis.ru)

Version

0.1

Date

2023-06-03

Copyright

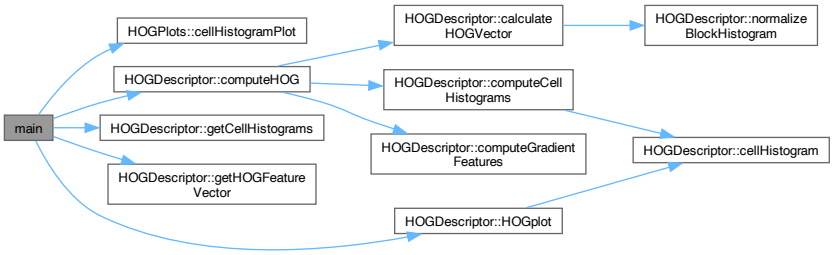
Copyright (c) 2023

5.7.2 Function Documentation

5.7.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

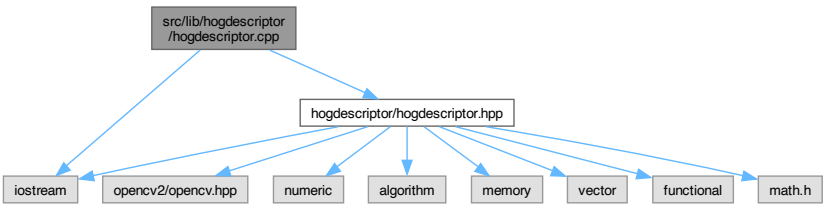
Here is the call graph for this function:



5.8 src/lib/hogdescriptor/hogdescriptor.cpp File Reference

```
#include <hogdescriptor/hogdescriptor.hpp>
#include <iostream>
```

Include dependency graph for hogdescriptor.cpp:



Functions

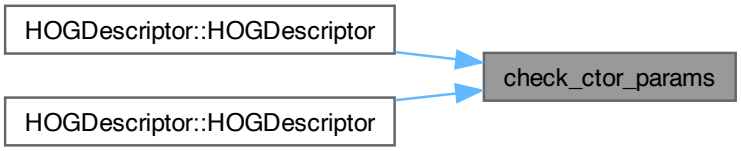
- void check_ctor_params (size_t blockSize, size_t cellSize, size_t stride, size_t binNumber, size_t gradType)

5.8.1 Function Documentation

5.8.1.1 check_ctor_params()

```
void check_ctor_params (
            size_t blockSize,
            size_t cellSize,
            size_t stride,
            size_t binNumber,
            size_t gradType )
```

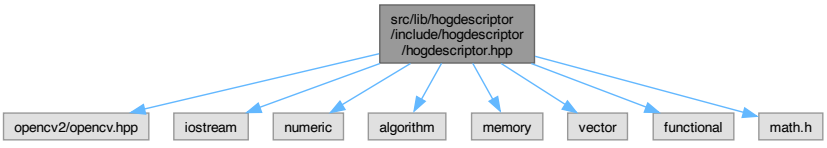
Here is the caller graph for this function:



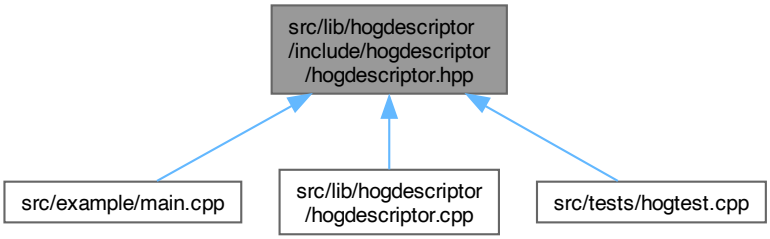
5.9 src/lib/hogdescriptor/include/hogdescriptor/hogdescriptor.hpp File Reference

```
#include <opencv2/opencv.hpp>
#include <iostream>
#include <numeric>
#include <algorithm>
#include <memory>
#include <vector>
#include <functional>
#include <math.h>
```

Include dependency graph for hogdescriptor.hpp:



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



Classes

- class HOGDescriptor

  Class for calculating the HOG features.

5.10 hogdescriptor.hpp

Go to the documentation of this file.

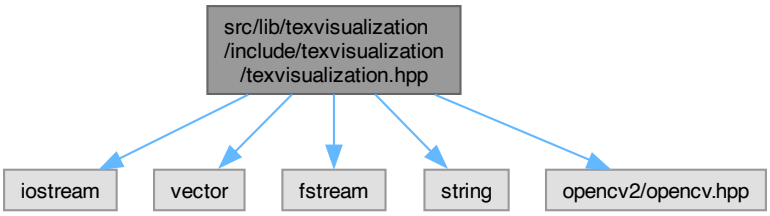
```
00001 #ifndef HOGDESCRIPTOR_H
00002 #define HOGDESCRIPTOR_H
00003
00004 #include <opencv2/opencv.hpp>
00005 #include <iostream>
00006 #include <numeric>
00007 #include <algorithm>
00008 #include <memory>
00009 #include <vector>
00010 #include <functional>
00011 #include <math.h>
00012
00016 class HOGDescriptor {
00017 public:
00021     HOGDescriptor();
00031     HOGDescriptor(const size_t blockSize, const size_t cellSize,
00032         const size_t stride, const size_t binNumber, const size_t gradType);
00033     HOGDescriptor(const size_t blockSize, const size_t cellSize);
00038     ~HOGDescriptor();
00039
00046     HOGDescriptor& operator=(const HOGDescriptor &rhs);
00047
00048 public:
00049
00056     void HOGplot(float scale, bool imposed);
00057
00058 public:
00059     static const size_t GRADIENT_SIGNED = 360;
00060     static const size_t GRADIENT_UNSIGNED = 180;
00061
00068     void computeHOG(cv::Mat& image);
00069
00075     std::vector<float> getHOGFeatureVector();
00076
00082     std::vector<std::vector<std::vector<float>>> getCellHistograms();
00083
00084     // /**
00085     //  * @brief Method for getting the visualization of the cell HOG
00086     //  *
00087     //  * @param x Number of the cell in the x direction
00088     //  * @param y Number of the cell in the y direction
00089     //  */
00090     // void visualizeHOGCell(int x, int y);
00091
00092 private:
```

```
00098     void computeGradientFeatures(cv::Mat& image);
00099
00108     std::vector<std::vector<std::vector<float>> computeCellHistograms(cv::Mat magnitude, cv::Mat
      orientation, std::vector<std::vector<std::vector<float>>& cell_histograms);
00109
00116     std::vector<float> cellHistogram(const cv::Mat& cellMagnitude, const cv::Mat& cellOrientation);
00117
00123     void normalizeBlockHistogram(std::vector<float>& block_histogram);
00124
00131     const std::vector<float> calculateHOGVector(const std::vector<std::vector<std::vector<float>>&
      cell_histograms);
00132
00133
00134
00135 private:
00136     int blockSize_;
00137     int cellSize_;
00138     int binNumber_;
00139     int binWidth_;
00140     int stride_;
00141     int gradType_;
00142
00143     bool hogFlag_ = false;
00144
00145     cv::Mat imageMagnitude_;
00146     cv::Mat imageOrientation_;
00147
00148     std::vector<std::vector<std::vector<float>> cellHistograms_;
00149     std::vector<float> hogFeatureVector_;
00150 };
00151
00152 #endif //HOGDESCRIPTOR_H
```

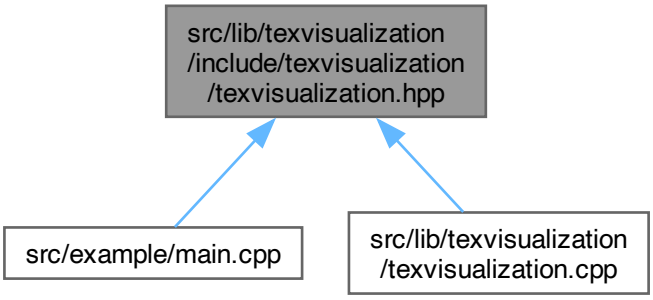
5.11 src/lib/texvisualization/include/texvisualization/texvisualization.hpp File Reference

```
#include <iostream>
#include <vector>
#include <fstream>
#include <string>
#include <opencv2/opencv.hpp>
```

Include dependency graph for texvisualization.hpp:



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



Classes

- class HOGPlots

  Class for creating .tex files with plots of HOG.

5.12 texvisualization.hpp

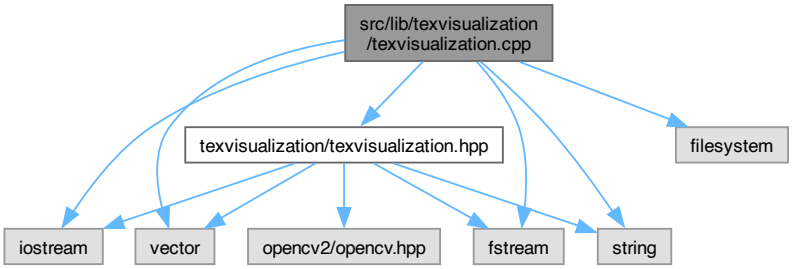
Go to the documentation of this file.

```
00001 #ifndef TEXHOG
00002 #define TEXHOG
00003
00004 #include <iostream>
00005 #include <vector>
00006 #include <fstream>
00007 #include <string>
00008 #include <opencv2/opencv.hpp>
00009
00013 class HOGPlots{
00014 public:
00019     HOGPlots() = default;
00020
00027     void HOGgrid(cv::Mat& image, float thickness, int cellSize);
00028
00035     void cellHistogramPlot(std::vector<float> cellHistogram, int blockWidth, const std::string&
      executablePath);
00036 };
00037
00038 #endif
```

5.13 src/lib/texvisualization/texvisualization.cpp File Reference

```
#include <texvisualization/texvisualization.hpp>
#include <string>
#include <iostream>
#include <vector>
#include <fstream>
```

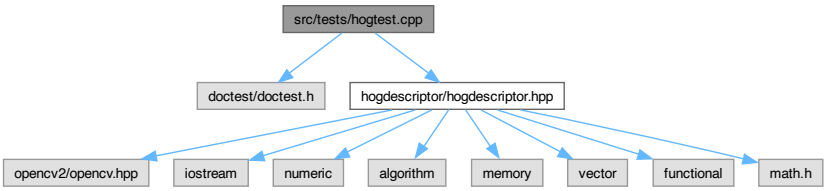
```
#include <filesystem>
```
Include dependency graph for texvisualization.cpp:



5.14 src/tests/hogtest.cpp File Reference

Program for testing HOGDescriptor class.

```
#include <doctest/doctest.h>
#include <hogdescriptor/hogdescriptor.hpp>
```
Include dependency graph for hogtest.cpp:



Macros

- #define DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN

Functions

- TEST_CASE ("HOGDescriptor: output vector")

5.14.1 Detailed Description

Program for testing HOGDescriptor class.

Author

Andrey Kadomtsev (m2204942@edu.misis.ru)

Version

0.1

Date

2023-06-04

Copyright

5.14.2 Macro Definition Documentation

5.14.2.1 DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN

```
#define DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN
```

5.14.3 Function Documentation

5.14.3.1 TEST_CASE()

```
TEST_CASE (
            ¨HOGDescriptor:  output vector¨  )
```

Here is the call graph for this function: