HOG library

Generated by Doxygen 1.9.7

1 Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

HOGDescriptor
Class for calculating the HOG (Histogram of oriented gradients) features **??**

texHOG
Class for creating .tex files with plots of HOG feature extraction process **??**

2 File Index

2.1 File List

Here is a list of all files with brief descriptions:

hogdescriptor.cpp **??**

hogdescriptor.hpp **??**

texvisualization.hpp **??**

texvisualization.cpp **??**

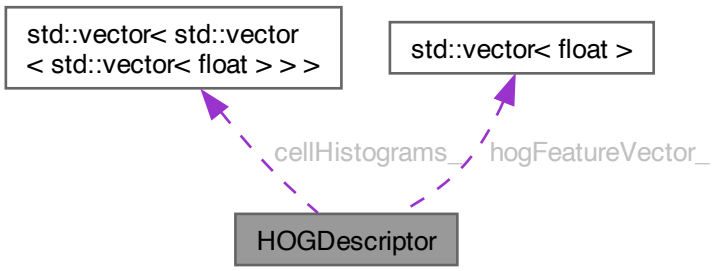
3 Data Structure Documentation

3.1 HOGDescriptor Class Reference

Class for calculating the HOG (Histogram of oriented gradients) features.

`#include <hogdescriptor.hpp>`

Collaboration diagram for HOGDescriptor:



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 with the given parameters.
- HOGDescriptor (const size_t blockSize, const size_t cellSize)

  Construct a new HOGDescriptor object.
- ~HOGDescriptor ()

  Destroy the HOGDescriptor object.
- void visualizeHOG (float scale, bool imposed)

  Method to visualize the final vector in separate window.
- void HOGgrid (cv::Mat &image, float thickness, int cellSize)

  Method to show the grid of cells 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< float > getCellHistogram (int y, int x)

  Get the Cell Histogram object.
- std::vector< std::vector< float > > getBlockHistogram (int y, int x)

  Get the Block Histogram object.
- void saveVectorData (const std::string &executablePath, const std::string &vectorName)

  Save hog vector in a file.

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 each pixel's gradient magnitude and orientation.
- 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 of the sliding window.
- 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 gradients.
- cv::Mat imageOrientation_

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

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

  Final vector of features.

3.1.1 Detailed Description

Class for calculating the HOG (Histogram of oriented gradients) features.

3.1.2 Constructor & Destructor Documentation

HOGDescriptor() [1/3]

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

Default constructor for the HOGDescriptor class.

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 with the given parameters.

Parameters

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

Here is the call graph for this function:

HOGDescriptor() [3/3]

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

Construct a new HOGDescriptor object.

Parameters

blockSize	Block size of the sliding window
cellSize	Size of the cell

Here is the call graph for this function:

~HOGDescriptor()

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

Destroy the HOGDescriptor object.

3.1.3 Member Function Documentation

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.

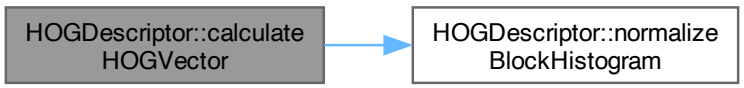
Parameters

cell_histograms	Matrix of histograms

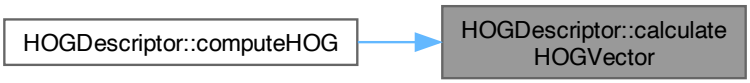
Returns

Final vector

Here is the call graph for this function:



Here is the caller graph for this function:



cellHistogram()

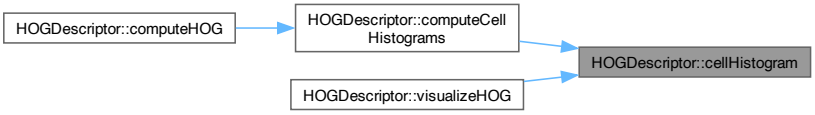
```
std::vector< float > HOGDescriptor::cellHistogram (
            const cv::Mat & cellMagnitude,
            const cv::Mat & cellOrientation )  [private]
```

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:



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.

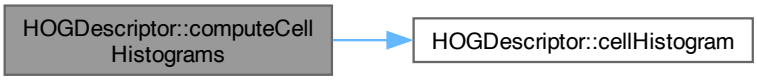
Parameters

gradient	Gradient matrix
orientation	Orientation matrix
cell_histograms	Output vector of HOG feature vectors for each cell

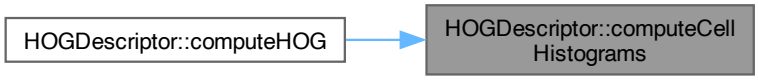
Returns

Matrix of histograms

Here is the call graph for this function:



Here is the caller graph for this function:



computeGradientFeatures()

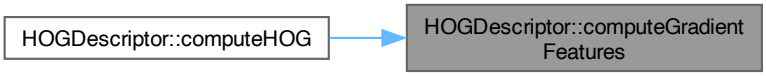
```
void HOGDescriptor::computeGradientFeatures (
            cv::Mat & image )  [private]
```

Function to compute each pixel's gradient magnitude and orientation.

Parameters

image	Input image

Here is the caller graph for this function:



computeHOG()

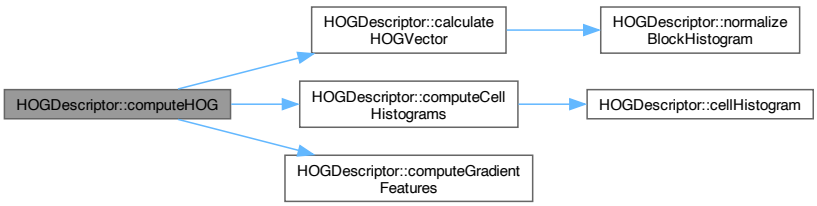
```
void HOGDescriptor::computeHOG (
            cv::Mat & image )
```

Method for computing HOG features.

Parameters

image	Input image

Here is the call graph for this function:



getBlockHistogram()

```
std::vector< std::vector< float > > HOGDescriptor::getBlockHistogram (
            int y,
            int x )
```

Get the Block Histogram object.

Parameters

y	first cell row position
x	first cell column position

Returns

Block histogram matrix

Here is the call graph for this function:

getCellHistogram()

```
std::vector< float > HOGDescriptor::getCellHistogram (
            int y,
            int x )
```

Get the Cell Histogram object.

Parameters

y	Cell row position
x	Cell column position

Returns

Histogram vector for the cell

Here is the caller graph for this function:

getHOGFeatureVector()

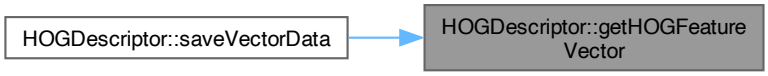
```
std::vector< float > HOGDescriptor::getHOGFeatureVector ( )
```

Method for getting the HOG feature vector.

Returns

Vector of features

Here is the caller graph for this function:



HOGgrid()

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

Method to show the grid of cells on image.

Parameters

image	Input image
thickness	Grid line thickness
cellSize	Cell size in pixels

normalizeBlockHistogram()

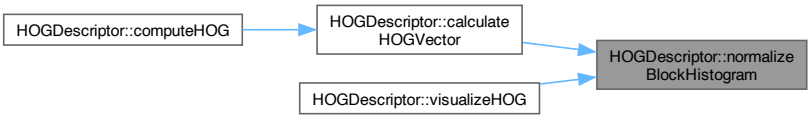
```
void HOGDescriptor::normalizeBlockHistogram (
            std::vector< float > & block_histogram )  [private]
```

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:



saveVectorData()

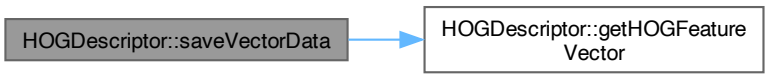
```
void HOGDescriptor::saveVectorData (
            const std::string & executablePath,
            const std::string & vectorName )
```

Save hog vector in a file.

Parameters

executablePath	Path where file will be saved
vectorName	Output vector name

Here is the call graph for this function:



visualizeHOG()

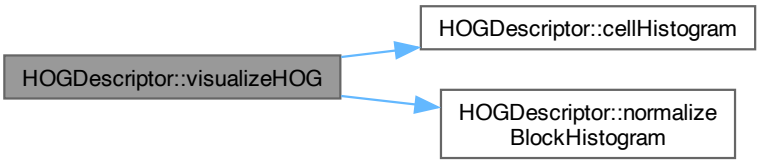
```
void HOGDescriptor::visualizeHOG (
            float scale,
            bool imposed )
```

Method to visualize the final vector in separate window.

Parameters

scale	Scale of the arrows
imposed	Background magnitude image for reference

Here is the call graph for this function:



3.1.4 Field Documentation

binNumber_

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

Number of the bins in the histogram of each cell.

binWidth_

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

Width of the bins in the histogram of each cell.

blockSize_

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

Block size of the sliding window.

cellHistograms_

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

Matrix of cell histograms.

cellSize_

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

Size of the cell in pixels.

GRADIENT_SIGNED

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

360 degree spread of histogram channels

GRADIENT_UNSIGNED

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

180 degree spread of histogram channels

gradType_

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

Type of the gradient calculation (unsigned or signed)

hogFeatureVector_

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

Final vector of features.

hogFlag_

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

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

imageMagnitude_

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

Magnitude of the gradients.

imageOrientation_

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

Orientation of the gradients.

stride_

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

Sliding window stride in pixels.

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

- hogdescriptor.hpp
- hogdescriptor.cpp

3.2 texHOG Class Reference

Class for creating .tex files with plots of HOG feature extraction process.

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

Public Member Functions

- texHOG ()=default

  Construct a new texHOG object.
- void cellHistogramPlot (std::vector< float > cellHistogram, int binWidth, const std::string &executablePath, const std::string &plotName)

  Method for creating a .tex file with the histogram of given cell.
- void blockHistogramPlot (std::vector< std::vector< float > > blockHistogram, int binWidth, const std::string &executablePath, const std::string &plotName)

  Method for creating a .tex file with the histograms of cell within given block.

3.2.1 Detailed Description

Class for creating .tex files with plots of HOG feature extraction process.

3.2.2 Constructor & Destructor Documentation

texHOG()

```
texHOG::texHOG ( )  [default]
```

Construct a new texHOG object.

3.2.3 Member Function Documentation

blockHistogramPlot()

```
void texHOG::blockHistogramPlot (
            std::vector< std::vector< float > > blockHistogram,
            int binWidth,
            const std::string & executablePath,
            const std::string & plotName )
```

Method for creating a .tex file with the histograms of cell within given block.

Parameters

blockHistogram	Matrix of cell histogram values
binWidth	Width of the histogram block
executablePath	Path to the .tex file
plotName	Output file name

cellHistogramPlot()

```
void texHOG::cellHistogramPlot (
            std::vector< float > cellHistogram,
            int binWidth,
            const std::string & executablePath,
            const std::string & plotName )
```

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

Parameters

cellHistogram	Cell histogram values
binWidth	Width of the histogram block
executablePath	Path to the .tex file
plotName	Output file name

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

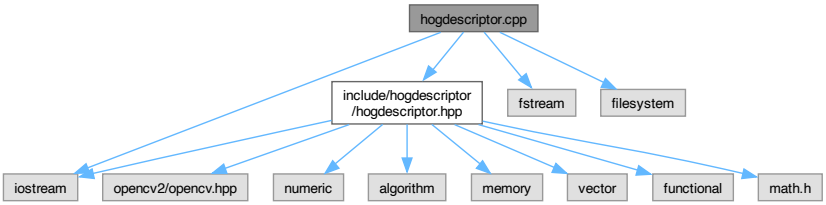
- texvisualization.hpp
- texvisualization.cpp

4 File Documentation

4.1 hogdescriptor.cpp File Reference

```
#include ¨include/hogdescriptor/hogdescriptor.hpp¨
#include <iostream>
#include <fstream>
#include <filesystem>
```

Include dependency graph for hogdescriptor.cpp:



Functions

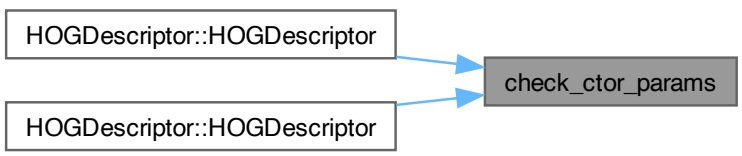
- void check_ctor_params (size_t blockSize, size_t cellSize, size_t stride, size_t binNumber, size_t gradType)

4.1.1 Function Documentation

check_ctor_params()

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

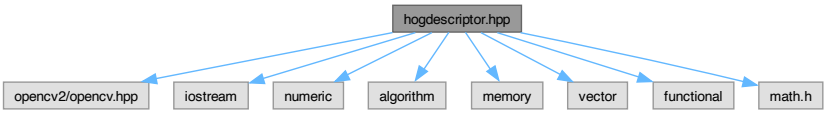
Here is the caller graph for this function:



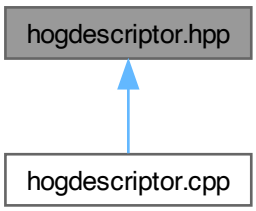
4.2 hogdescriptor.hpp File Reference

```
#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:



Data Structures

- class HOGDescriptor

  Class for calculating the HOG (Histogram of oriented gradients) features.

4.3 hogdescriptor.hpp

Go to the documentation of this file.

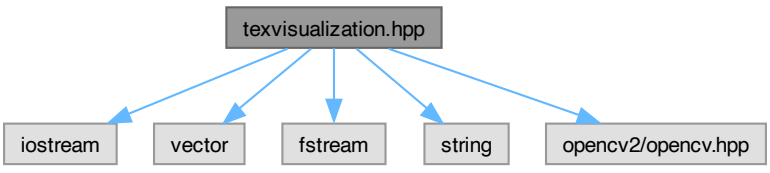
```
00001 #ifndef HOGDESCRIPTOR_H
00002 #define HOGDESCRIPTOR_H
00003
00004 #include <opencv2/opencv.hpp>
00005 #include <iostream>
00006 #include <numeric>
00007 #include <algorithm>
00008 #include <memory>
00009 #include <vector>
00010 #include <functional>
00011 #include <math.h>
00012
```

```
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);
00039     HOGDescriptor(const size_t blockSize, const size_t cellSize);
00043     ~HOGDescriptor();
00044
00045 public:
00052     void visualizeHOG(float scale, bool imposed);
00060     void HOGgrid(cv::Mat& image, float thickness, int cellSize);
00061
00062 public:
00063     static const size_t GRADIENT_SIGNED = 360;
00064     static const size_t GRADIENT_UNSIGNED = 180;
00070     void computeHOG(cv::Mat& image);
00071
00077     std::vector<float> getHOGFeatureVector();
00078
00086     std::vector<float> getCellHistogram(int y, int x);
00094     std::vector<std::vector<float» getBlockHistogram(int y, int x);
00101     void saveVectorData(const std::string& executablePath, const std::string& vectorName);
00102
00103 private:
00109     void computeGradientFeatures(cv::Mat& image);
00110
00119     std::vector<std::vector<std::vector<float»> computeCellHistograms(cv::Mat magnitude, cv::Mat
    orientation, std::vector<std::vector<std::vector<float»>& cell_histograms);
00120
00127     std::vector<float> cellHistogram(const cv::Mat& cellMagnitude, const cv::Mat& cellOrientation);
00128
00134     void normalizeBlockHistogram(std::vector<float>& block_histogram);
00135
00142     const std::vector<float> calculateHOGVector(const std::vector<std::vector<std::vector<float»>&
    cell_histograms);
00143
00144 private:
00145     int blockSize_;
00146     int cellSize_;
00147     int binNumber_;
00148     int binWidth_;
00149     int stride_;
00150     int gradType_;
00151
00152     bool hogFlag_ = false;
00153
00154     cv::Mat imageMagnitude_;
00155     cv::Mat imageOrientation_;
00156
00157     std::vector<std::vector<std::vector<float»> cellHistograms_;
00158     std::vector<float> hogFeatureVector_;
00159 };
00160
00161 #endif //HOGDESCRIPTOR_H
```

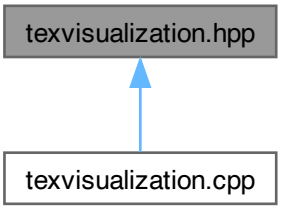
4.4 texvisualization.hpp File Reference

```
#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:



Data Structures

- class texHOG

  Class for creating .tex files with plots of HOG feature extraction process.

4.5 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 texHOG{
00014 public:
00019     texHOG() = default;
00020
00029     void cellHistogramPlot(std::vector<float> cellHistogram, int binWidth, const std::string&
      executablePath, const std::string& plotName);
00030
00039     void blockHistogramPlot(std::vector<std::vector<float» blockHistogram, int binWidth, const
      std::string& executablePath, const std::string& plotName);
00040 };
00041
00042 #endif
```

4.6 texvisualization.cpp File Reference

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

```
#include <filesystem>
```

Include dependency graph for texvisualization.cpp: