Frog (Face RecOgnition with Gabor)

0.1

Generated by Doxygen 1.6.1

Fri Dec 18 17:01:32 2009

Contents

1 CompareFaces

1.1 How I'am??

I'am a module that compute the score verification between 2 geometrically normalized faces. The verification is based on the param input file

I output the score between the two images.

1.2 Usage:

CompareList param InputImage1 InputImage2

Parameters:

param Parameters xml file (see paramfile.xml)

InputImage1 The first geometrically normalized face image (pgm format)

InputImage2 The second geometrically normalized face image (pgm format)

Returns:

0 when succeed

Here is an example of normalized images (use NormFaces to get this kind of images)

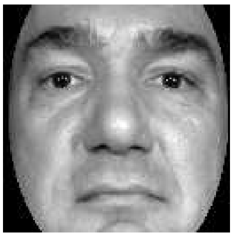


Figure 1: geometrically Normalized

2 CompareList

2.1 How I'am??

I'am the module that allows the comparison of list of tests contained in a text file.

Each comparison test is composed of two geometrically normalized face images.

I output the scores of each test to the terminal and if the output score file is given I'am able to write into it.

2.2 Usage:

CompareList param ListofTest Imagesindir [outputscores]

Parameters:

param Parameters xml file (see paramfile.xml)

ListofTest A text file containing the list of comparisons (see Testlist.lst)

- The reference and test images must be in pgm format

Imagesindir Path to the normalized images (only geometrically)

outputscores [Optional] if set the scores will be directed to this file

Returns:

0 when succeed

3 CompareTemplates

3.1 How I'am??

I'am the module that computes the score verification between 2 constructed templates. I load the two templates and I give you the distance between them.

3.2 Usage:

CompareTemplates Template1 Template2 Distance

Parameters:

Template1 The first face template

Template2 The second face template

Distance The distance/similarity measure

- L1: sum_absolute_value(T1-T2)
- L2: norm_Frobenius(T1-T2)
- COS: -dotproduct (T1,T2) / (sqrt (sum_square (T1) *sum_square (T2)))
- NORMDIST: Norm_Frobenius (T1/Norm_Frobenius(T1) - T2/Norm_Frobenius(T2))

Returns:

0 when succeed

4 frgc_run

4.1 How I'am??

I'am the module that can run the FRGC-v2 experiments (Never ask the database from my author just go to `http://www.frvt.org/FRGC/` and ask).

The author could give you only the xml metadata and the masks.

I use the multi-threading to accelerate your work!

4.2 Usage:

frgc_run param queryxml targetxml inputdir maskfile Similaritymatrix

Parameters:

- ***param*** Parameters xml file (see paramfile.xml)
- ***queryxml*** Xml file containing the list of Query files (Test images) (see Metadata.xml example of the xml format)
- ***targetxml*** Xml file containing the list of Target files (Reference images) (ame xml format as queryxml)
- ***inputdir*** Path to the input geometrically normalized images (use NormFaces for that)
- ***maskfile*** Mask file containing the intra/interclass tests (see MBGC mask format)
- ***Similaritymatrix*** The output file containing the Similarity matrix (see MBGC similarity format)

Returns:

0 when succeed

5 GetPermformance

5.1 How I'am??

I'am the module that computes the performance of the algorithms when you give me the similarity matrix, and the corresponding mask. I can output many things, so read the following, but I give at the minimum the EER and the VR at 0.1% of FAR.

5.2 Usage:

GetPerformance -S Simfile -M maskfile -h Hist-step [-a ouputintraHist [/dev/null]] [-b outputinterHist[/dev/null]] [-R FarFrr[/dev/null]] [-t type (distance/similarity) [distance]] -[f feedback] [-i intrascore] [-e interscores]

Parameters:

- ***Simfile*** The similarity matrix created by frgc_run or mbgc_run
- ***maskfile*** Mask file containing the intra/interclass tests (see MBGC mask format)
- ***Hist-step*** histogramme sampling steps generally (maxscore - minscore)/10000 will be enough.
- ***ouputintraHist*** it's an optional output file, if set, the distribution of intraclass (genuine) comparisons tests will be saved
- ***outputinterHist*** it'salso an optional output file, if set, the distribution of interclass (impostor) comparisons tests will be saved
- ***FarFrr*** it's an optional output file, if set, the score FAR FRR the confidences will be saved
- ***type*** it's optional value [distance/similarity], by default the system will detect if the used measure is similarity or a distance
- ***feedback*** it's allow verbosing
- ***intrascore*** it's an optional output file, if set, the intraclass scores will be saved
- ***interscores*** it's an optional output file, if set, the interclass scores will be saved

Returns:

0 when succeed

6 mbgc_run

6.1 How I'am??

I'am the module that can run the MBGC-v1 and v2 experiments (Never ask the database from my author just go to `http://www.frvt.org/MBGC/` and ask).

The author could give you only the xml metadata and the masks.

I use the multi-threading to accelerate your work!

6.2 Usage:

mbgc_run param queryxml query_meta targetxml target_meta inputdir Similaritymatrix

Parameters:

param Parameters xml file (see paramfile.xml)

queryxml Xml file containing the list of Query files (Test Videos) (see mbgc query xml format)

query_meta Xml file containing the metadata of the Query files (see Metadata.xml example of the xml format)

targetxml Xml file containing the list of Target files (Reference images) (see mbgc target xml format)

target_meta Xml file containing the metadata of the Target files (see Metadata.xml example of the xml format)

inputdir Path to the input geometrically normalized images (use NormFaces for that)

Similaritymatrix The output file containing the Similarity matrix (see MBGC similarity format)

Returns:

0 when succeed

7 NormFaces

7.1 How I'am??

I'am the module that allows the geometric extraction and normalization of a list of faces and store them in a gray-scale pgm file

I use the multi-threading to accelerate your work!

7.2 Usage:

NormFaces param xmllist imageindir imageoutdir

Parameters:

param Parameters xml file (see paramfile.xml)

xmllist xml file containing the list of files with there metadata (see Metadata.xml or Learning.xml as examples)

imageindir path to the original images

imageoutdir path to the output normalized images

Returns:

0 when succeed

Here is an output normalized images

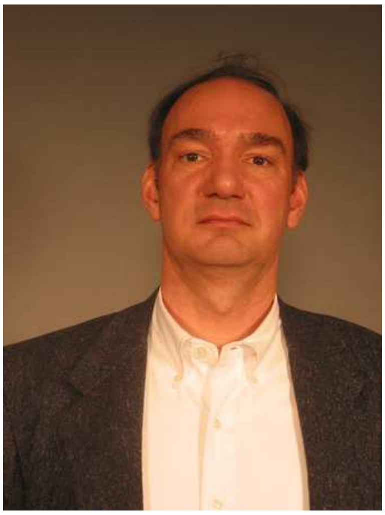


Figure 2: Original Image

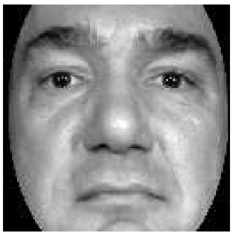


Figure 3: geometrically Normalized

8 SpaceLearner

8.1 How I'am??

I'am the module that will compute the reduced space, based on the parameters given in the param xml file.

I'am able to create different reduced spaces from gray-scale images using just the pixels values, or also using the Gabor filtering method

The follwing is the list of space reduction methods that I could handle (with or without Gabor filtering):

- PCA, LDA, DLDA, KFA, GDA

- when using Gabor I can handle the real, imaginary, magnitude, angle and a combination of those components.

for more details please read the description of the nodes **"gabor"** and **"reduction_space"** at paramfile.xml .

8.2 Usage:

SpaceLearner param xmlfile inputdir

Parameters:

param Parameters xml file (see paramfile.xml)

xmlfile Xml file containing the list of users and there images files (Test images) (see Learning.xml)

inputdir Path to the input geometrically normalized images (use NormFaces for that)

Returns:

0 when succeed

9 TemplateFace

9.1 How I'am??

I'am the module that allows the extraction of template from a geometrically normalized face

9.2 Usage:

TemplateFace param InputImage OutputTemplate

Parameters:

param Parameters xml file (see paramfile.xml)

InputImage The geometrically normalized face image (pgm format)

OutputTemplate The output filename template

Returns:

0 when succeed

Here is an example of normalized face image (use NormFaces to get this kind of images)

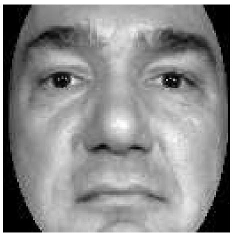


Figure 4: geometrically Normalized face

10 TemplateList

10.1 How I'am??

I'am the module that allows the extraction of templates from a list if geometrically normalized faces

I use the multi-threading to accelerate your work!

10.2 Usage:

TemplateList param xmllist Imagesindir Templateoutdir

Parameters:

param Parameters xml file (see paramfile.xml)

xmllist xml file containing the list of files with there metadata (see Metadata.xml or Learning.xml as examples)

Imagesindir path to the normalized images

Templateoutdir path to the output templates

Returns:

0 when succeed

11 Learning.xml

Example of an xml file used for learning the reduced Space this file is used by SpaceLearner and could be used by NormFaces

```
<?xml version="1.0" encoding="UTF-8"?>
<Faces>

<user id="0001">
<image name="frgc/nd1/Fall2002/2002-240/02463d170.jpg" XleftEye="815" YleftEye="609" XrightEye="595"
       YrightEye="610" Xnose="722" Ynose="684" Xmouth="716" Ymouth="815" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-240/02463d171.jpg" XleftEye="815" YleftEye="608" XrightEye="595"
       YrightEye="606" Xnose="719" Ynose="689" Xmouth="711" Ymouth="820" type="controlled" />
.......
</user>

<user id="0002">
<image name="frgc/nd1/Fall2002/2002-239/04201d52.jpg" XleftEye="631" YleftEye="620" XrightEye="433"
       YrightEye="629" Xnose="545" Ynose="694" Xmouth="539" Ymouth="814" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-239/04201d53.jpg" XleftEye="625" YleftEye="611" XrightEye="424"
       YrightEye="620" Xnose="536" Ynose="692" Xmouth="533" Ymouth="816" type="controlled" />
...
</user>

<user id="0003">
<image name="frgc/nd1/Fall2002/2002-240/04202d57.jpg" XleftEye="637" YleftEye="620" XrightEye="452"
YrightEye="631" Xnose="552" Ynose="727" Xmouth="558" Ymouth="832" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-240/04202d58.jpg" XleftEye="608" YleftEye="592" XrightEye="427"
YrightEye="613" Xnose="522" Ynose="702" Xmouth="534" Ymouth="802" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-255/04202d71.jpg" XleftEye="608" YleftEye="698" XrightEye="422"
YrightEye="722" Xnose="527" Ynose="806" Xmouth="538" Ymouth="906" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-255/04202d72.jpg" XleftEye="604" YleftEye="703" XrightEye="417"
```

```
YrightEye="729" Xnose="525" Ynose="822" Xmouth="538" Ymouth="912" type="controlled" />
....
</user>

<user id="0004">
<image name="frgc/nd1/Fall2002/2002-239/04203d58.jpg" XleftEye="775" YleftEye="978" XrightEye="579"
YrightEye="977" Xnose="689" Ynose="1081" Xmouth="692" Ymouth="1184" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-239/04203d59.jpg" XleftEye="799" YleftEye="974" XrightEye="600"
YrightEye="972" Xnose="712" Ynose="1079" Xmouth="714" Ymouth="1177" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-261/04203d72.jpg" XleftEye="763" YleftEye="598" XrightEye="558"
YrightEye="606" Xnose="670" Ynose="717" Xmouth="674" Ymouth="817" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-261/04203d73.jpg" XleftEye="777" YleftEye="596" XrightEye="568"
YrightEye="601" Xnose="682" Ynose="720" Xmouth="690" Ymouth="811" type="controlled" />
...
</user>
.....
<user id="xxxx">
....
</user>

</Faces>
```

12 Metadata.xml

an example of an xml file used for Normalizing images

- NormFaces

and also for runing the follwing modules

- frgc_run
- mbgc_run

```
<?xml version="1.0" encoding="UTF-8"?>
<Faces>
<image name="frgc/nd1/Fall2002/2002-240/02463d170.jpg" XleftEye="815" YleftEye="609" XrightEye="595"
YrightEye="610" Xnose="722" Ynose="684" Xmouth="716" Ymouth="815" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-240/02463d171.jpg" XleftEye="815" YleftEye="608" XrightEye="595"
YrightEye="606" Xnose="719" Ynose="689" Xmouth="711" Ymouth="820" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-239/04201d52.jpg" XleftEye="631" YleftEye="620" XrightEye="433"
YrightEye="629" Xnose="545" Ynose="694" Xmouth="539" Ymouth="814" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-239/04201d53.jpg" XleftEye="625" YleftEye="611" XrightEye="424"
YrightEye="620" Xnose="536" Ynose="692" Xmouth="533" Ymouth="816" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-240/04202d57.jpg" XleftEye="637" YleftEye="620" XrightEye="452"
YrightEye="631" Xnose="552" Ynose="727" Xmouth="558" Ymouth="832" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-240/04202d58.jpg" XleftEye="608" YleftEye="592" XrightEye="427"
YrightEye="613" Xnose="522" Ynose="702" Xmouth="534" Ymouth="802" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-255/04202d71.jpg" XleftEye="608" YleftEye="698" XrightEye="422"
YrightEye="722" Xnose="527" Ynose="806" Xmouth="538" Ymouth="906" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-255/04202d72.jpg" XleftEye="604" YleftEye="703" XrightEye="417"
YrightEye="729" Xnose="525" Ynose="822" Xmouth="538" Ymouth="912" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-239/04203d58.jpg" XleftEye="775" YleftEye="978" XrightEye="579"
YrightEye="977" Xnose="689" Ynose="1081" Xmouth="692" Ymouth="1184" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-239/04203d59.jpg" XleftEye="799" YleftEye="974" XrightEye="600"
YrightEye="972" Xnose="712" Ynose="1079" Xmouth="714" Ymouth="1177" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-261/04203d72.jpg" XleftEye="763" YleftEye="598" XrightEye="558"
YrightEye="606" Xnose="670" Ynose="717" Xmouth="674" Ymouth="817" type="controlled" />
<image name="frgc/nd1/Fall2002/2002-261/04203d73.jpg" XleftEye="777" YleftEye="596" XrightEye="568"
YrightEye="601" Xnose="682" Ynose="720" Xmouth="690" Ymouth="811" type="controlled" />
```

```
......
</Faces>
```

13 paramfile.xml

An example of parameters file, it’s used by the following modules

- CompareList
- CompareFaces
- NormFaces
- frgc_run
- SpaceLearner

```
<?xml version="1.0" encoding="UTF-8"?>

<PFC_params dir="." threads="2" verb="1">

  <norm eyedist="50" illum="AS" params="150:1"/>

  <gabor method="MAGN" reductcoef="4" scales="5" orientations="8" minWaveLength="8" mult="2" sigmaOnf="0.6

  <reduction_space Method="DLDA" Method-param="" file="Spacefile" nusers="220" control="5" noncontrol="5"/

  <Similarity measure="COS" />

</PFC_params>
```

13.1 Description

This file contains all the parameters that the different programs need

```
the following are the details of the xml structure

-node PFC_params

-*-attribute1 : "dir"
=> the directory used for the workspace

-*-attribute2 : "threads"
=> the number of threads to use (use max 1 for monocore machines)

-*-attribute3 : "verb"
=> the level of the program verbosing (from 0 to 3)

***********************************************************************************************************
-node1: "norm"
image normalisation section (geometric and illumination)

-*-attribute1 : "eyedist"
=> the distance between eyes’ center, it used to normalize the face and to compute the gabor filters dimen

-*-attribute2 : "illum"
=> the illumintaion methods
-NONE: No illumination normalization done
```

```
-AS: anisotropic smoothing
http://blues.ius.cs.cmu.edu/ralph/Publications/avbpa03.pdf
-HISTOGRAMM: Histogramm equalization
-GAMMA: Gamma correction
-LOG: log transformation

-⋆-attribute3 : "params"
=> illumination parameters
-NONE takes no parameters
-AS takes 2 parameters (steps:reg)
-HISTOGRAMM takes no parameters
-GAMMA takes 1 parameter 0 < gamma < 1
-LOG takes no parameters

**********************************************************************************************************
-node2: gabor
to have a good idea on what's gabor wavelet: http://www.csse.uwa.edu.au/~pk/research/matlabfns/PhaseCongru

-⋆-attribute1 : "method"
=> the used part of gabor complex features
- "NONE": only the pixel intensity is used. no gabor features computed
- "REAL": real part of complex features
- "IMAG": imaginary part
- "MAGN": magnitude part
- "PHASE": angle part
- "ALL" : use REAL, IMAG, MAGN and PHASE
=>You can use a combination of features example(method="MAGN,REAL,IMAG")

-⋆-attribute2 : "reductcoef"
=> the coefficient sampling of the gabor features (note that for a reductcoef="2"
just 1 gabor component on 2 is kept) a normalisation of the gabor
features is done before the sampling to get mean=0 and std=1 for the real and imaginary parts.

-⋆-attribute5 : "scales"
=>the number of scales of the gabor filters

-⋆-attribute6 : "orientations"
=>the number of orientation of the gabor filters

-⋆-attribute7 : "minWaveLength"
=>the lentgh of the first scale gabor wavelet

-⋆-attribute8 : "mult"
=>tTo go from a scale wavelet to the next one the lentgh is multiplied by mult.

-⋆-attribute9 : "sigmaOnf"
=>Ratio of the standard deviation of the Gaussian describing the log Gabor filter's
transfer function in the frequency domain to the filter center frequency.

-⋆-attribute10 : "ThetaOnSigma"
=>Ratio of angular interval between filter orientations and the standard deviation
of the angular Gaussian function used to construct filters in the freq. plane

**********************************************************************************************************
-node3: reduction_space

-⋆-attribute1 : "Method"
=>the reduction space method used
- "PCA": Principal Component Analysis: http://face-rec.org/algorithms/#PCA
- "LDA" : Linear Discrimninat Analysis: http://face-rec.org/algorithms/#LDA
- "DLDA" : Direct Linear Discrimninat Analysis: http://dx.doi.org/10.1016/S0031-3203(00)00162-X
- "KFA" : Kernel Fisher Analysis: http://dx.doi.org/10.1109/TPAMI.2006.90
- "GDA" : General Discriminant Analysis: G. Baudat and F. Anouar, "Generalized Discriminant Analysis Using

-⋆-attribute2 : "Method-param"
=> parameters for the chosen method for:
-PCA : Method-param="variance=X"
```

```
with 0 < X < 100
-KFA or GDA : Method-param="type=X;params=Y,Z"
with X= 0 (linear kernel) => c = Y*a + Z;
       1 (Polynomial kernel) => pow ( ( a + Z ), Y );
2 (RBF kernel) =>  exp ( -Y*a ) ;
3 (sigmoid kernel) => tanh ( Y*a + Z );

-*-attribute3 : "file"
=> the path of the space reduced file (saved or loaded)
used by the different module to avoid arguments input error

-*-attribute4 : "nusers"
=> number of user used to learn the space

-*-attribute5 : "control"
=> number of controlled images/user

-*-attribute6 : "noncontrol"
=>number of uncontrolled images/user

**********************************************************************************************************
-node4: Similarity

-*-attribute5 : "measure"
=> the measure used to compare templates
- "L1": sum_absolute_value(T1-T2);
- "L2": norm_Frobenius(T1-T2);
- "COS" : -dotproduct ( T1,T2) / ( sqrt ( sum_square ( T1 ) *sum_square ( T2 ) ) );
- "NORMDIST": Norm_Frobenius (T1/Norm_Frobenius(T1) - T2/Norm_Frobenius(T2));
=>Note that NORMDIST is equivalent to COS

**********************************************************************************************************

-->
```

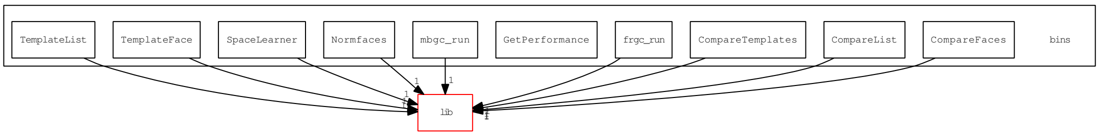
14 Testlist.lst

an example of comparison tests file used by CompareList

```
frgc/nd1/Fall2002/2002-240/04201d52.pgm frgc/nd1/Fall2002/2002-240/04201d53.pgm
frgc/nd1/Fall2002/2002-240/04201d53.pgm frgc/nd1/Fall2002/2002-240/04424d02.pgm
frgc/nd1/Fall2002/2002-240/04201d56.pgm frgc/nd1/Fall2002/2002-240/04203d63.pgm
frgc/nd1/Fall2002/2002-240/04201d57.pgm frgc/nd1/Fall2002/2002-240/04261d37.pgm
frgc/nd1/Fall2002/2002-240/04203d58.pgm frgc/nd1/Fall2002/2002-240/04261d40.pgm
frgc/nd1/Fall2002/2002-240/04203d59.pgm frgc/nd1/Fall2002/2002-240/04203d63.pgm
frgc/nd1/Fall2002/2002-240/04203d62.pgm frgc/nd1/Fall2002/2002-240/04261d36.pgm
frgc/nd1/Fall2002/2002-240/04203d63.pgm frgc/nd1/Fall2002/2002-240/04279d06.pgm
frgc/nd1/Fall2002/2002-240/04261d36.pgm frgc/nd1/Fall2002/2002-240/04203d63.pgm
.....
*
```

15 Directory Documentation

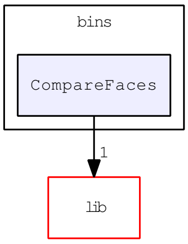
15.1 bins/ Directory Reference



Directories

- directory CompareFaces
- directory CompareList
- directory CompareTemplates
- directory frgc_run
- directory GetPerformance
- directory mbgc_run
- directory Normfaces
- directory SpaceLearner
- directory TemplateFace
- directory TemplateList

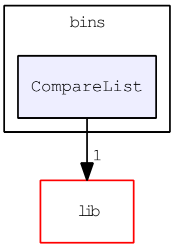
15.2 bins/CompareFaces/ Directory Reference



Files

- file main.cpp

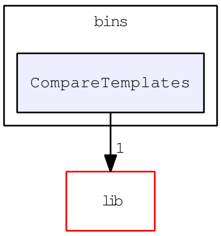
15.3 bins/CompareList/ Directory Reference



Files

- file main.cpp

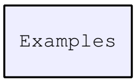
15.4 bins/CompareTemplates/ Directory Reference



Files

- file main.cpp

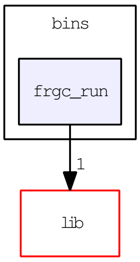
15.5 Examples/ Directory Reference



Files

- file Learning.xml
- file metadata.xml
- file PFC_param.xml
- file TestList.lst

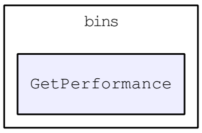
15.6 bins/frgc_run/ Directory Reference



Files

- file main.cpp

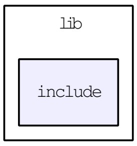
15.7 bins/GetPerformance/ Directory Reference



Files

- file GetPerformance.cpp

15.8 lib/include/ Directory Reference

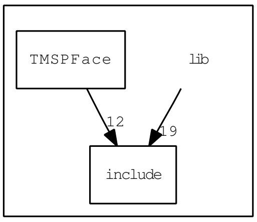


Files

- file comparator.h
- file gaborate.h
- file inputs.h
- file kernel.h
- file linearreducer.h
- file mask.h

- file nonlinearreducer.h
- file pfcface.h
- file templator.h
- file timer.h
- file TMSP_image.h
- file verbose.h

15.9 lib/ Directory Reference



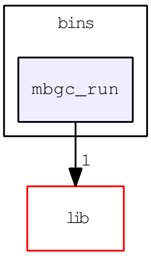
Directories

- directory include
- directory TMSPFace

Files

- file TMSPFace.h

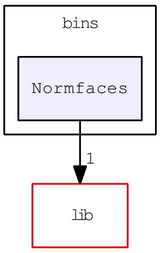
15.10 bins/mbgc_run/ Directory Reference



Files

- file main.cpp

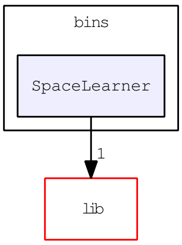
15.11 bins/Normfaces/ Directory Reference



Files

- file main.cpp

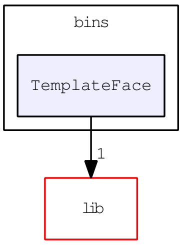
15.12 bins/SpaceLearner/ Directory Reference



Files

- file main.cpp

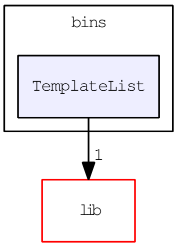
15.13 bins/TemplateFace/ Directory Reference



Files

- file main.cpp

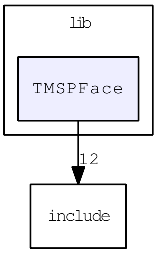
15.14 bins/TemplateList/ Directory Reference



Files

- file main.cpp

15.15 lib/TMSPFace/ Directory Reference



Files

- file comparator.cpp
- file gaborate.cpp
- file inputs.cpp
- file kernel.cpp
- file linearreducer.cpp
- file mask.cpp
- file nonlinearreducer.cpp
- file pfcface.cpp
- file templator.cpp
- file timer.cpp
- file TMSP_image.cpp
- file verbose.cpp

16 Namespace Documentation

16.1 TMSP_Face_Space Namespace Reference

Data Structures

- class Comparator

  Class that compares two given templates based on the chosen Distance.

- class Gaborate

  Class that computes the gabor filtering of an input image.

- class inputs

  Class that reads framework parameters from the xml parameters file paramfile.xml .

- struct kerlin

  structure that stores a Linear kernel parameters Y = X∗alpha + decal;

- struct kerpoly

  structure that stores a Polynomial kernel parameters Y = pow ((X + decal),power)

- struct kerrbf

  structure that stores a Radial Basis Function kernel parameters Y= exp (-X∗power) ;

- struct kersigmoid

  structure that stores a Sigmoid kernel parameters Y=tanh (X∗gamma + decal);

- class kernel

  Class that create kernels and implement kernel methods.

- class LinearReducer

  Class used to reduce the input space by linear methods (PCA,LDA,DLDA).

- class Mask

  Class that creates an elliptic mask to be applied to faces.

- class NonLinearReducer

  Class used to reduce the input space by nonlinear methods using kernel approaches (KFA,GDA).

- class PFCface

  Class that stores all the data we need to do a verification.

- class Templator

  Class that extracts template from a given, geometrically normalized and corrected illumination, face.

- class Timer

  Class that allows to get time performance.

- struct Face_Coordinates

  Structure for storing the coordinates of Eyes, Nose and Mouth location in the original image.

- struct PfcPoint

  structure for storing a point

- class Pfc_Image

  Class that allows the manipulation of image for the baseline.

- class verbose

  Class that controls the verbozing of different classes.

Enumerations

- enum DISTANCE { C_L1 = 1, C_L2, C_Angle, C_NormDist }

  Supported measure distances between templates.

- enum LProblem { PCA = 0, LDA, DLDA }

  the supported linear problems

- enum NProblem { KFA = 0, GDA }

  the supported Nonlinear problems

- enum LightEnhance {
  Im_NoEnhance = 0, Im_Histogram, Im_Gamma, Im_Log,
  Im_AnisSmooth, Im_MultiRetinex, Im_Pers }

  Supported Light correction.

- enum ImFormat { Im_8 = 1, Im_16, Im_24, Im_32 }

  Image pixels format.

Functions

- Pfc_Image PFCImageFromMat (Matrix &A)

  return a pointer to a Pfc_Image from Matrix data

- int ASNorm (Pfc_Image &InputImage, int steps, float lambda, Pfc_Image &ReflectImage, Pfc_Image &LightImage)

  apply the Anisotropic smoothing

- int ASNorm (Pfc_Image &InputImage, int steps, float lambda, Pfc_Image &ReflectImage, Pfc_Image &LightImage, Mask ROI, int meanref, float std)

  apply the Anisotropic smoothing with histogramm correction using a ROI and mean and std

16.1.1 Detailed Description

Author:

Anouar mellakh <me.anouar@gmail.com>

16.1.2 Enumeration Type Documentation

16.1.2.1 enum TMSP_Face_Space::DISTANCE

Supported measure distances between templates.

Enumerator:

C_L1 sum_absolute_value(T1-T2)

C_L2 norm_Frobenius(T1-T2)

C_Angle minus cosinus between the 2 templates vectors [-dotproduct (T1,T2) / (sqrt (sum_square (T1) ∗sum_square (T2)))]

C_NormDist Norm_Frobenius (T1/Norm_Frobenius(T1) - T2/Norm_Frobenius(T2));

16.1.2.2 enum TMSP_Face_Space::ImFormat

Image pixels format.

Enumerator:

Im_8 (1 Byte by pixel)

Im_16 (2 Byte by pixel)

Im_24 (3 Byte by pixel)

Im_32 (4 Byte by pixel)

16.1.2.3 enum TMSP_Face_Space::LightEnhance

Supported Light correction.

Enumerator:

Im_NoEnhance no Light correction

Im_Histogram Histogram Equalization

Im_Gamma Gamma correction

Im_Log logarithmic correction

Im_AnisSmooth Anisotropic smoothing

Im_MultiRetinex MultiRetinex **Not-Implemented**

Im_Pers CLS (correction of light by symmetry) **Not-Implemented**

16.1.2.4 enum TMSP_Face_Space::LProblem

the supported linear problems

Enumerator:

PCA Principal component Analysis

LDA Linear Disciminant Analysis

DLDA Direct Linear Discriminant Analysis

16.1.2.5 enum TMSP_Face_Space::NProblem

the supported Nonlinear problems

Enumerator:

KFA Kernel Fisher Analysis

GDA General Discriminant Analysis

16.1.3 Function Documentation

16.1.3.1 int TMSP_Face_Space::ASNorm (Pfc_Image & *InputImage*, int *steps*, float *lambda*, Pfc_Image & *ReflectImage*, Pfc_Image & *LightImage*, Mask *ROI*, int *meanref*, float *std*)

apply the Anisotropic smoothing with histogramm correction using a ROI and mean and std

Parameters:

InputImage a Pfc_Image to be treated

steps number of iterations for the method

lambda controlling coefficient

ReflectImage returned reflectance component

LightImage returned illumination component

ROI Mask of the region of interest

meanref the mean value to center the reflectance component value between (0-255)

std the std value to strech the histogram of the reflectance component value between (30-60)

Returns:

0 when well done

16.1.3.2 int TMSP_Face_Space::ASNorm (Pfc_Image & *InputImage*, int *steps*, float *lambda*, Pfc_Image & *ReflectImage*, Pfc_Image & *LightImage*)

apply the Anisotropic smoothing

Parameters:

InputImage a Pfc_Image to be treated

steps number of iterations for the method

lambda controlling coefficient

ReflectImage returned reflectance component

LightImage returned illumination component

Returns:

0 when well done

16.1.3.3 Pfc_Image TMSP_Face_Space::PFCImageFromMat (Matrix & *A*)

return a pointer to a Pfc_Image from Matrix data

Parameters:

A Matrix Object

Returns:

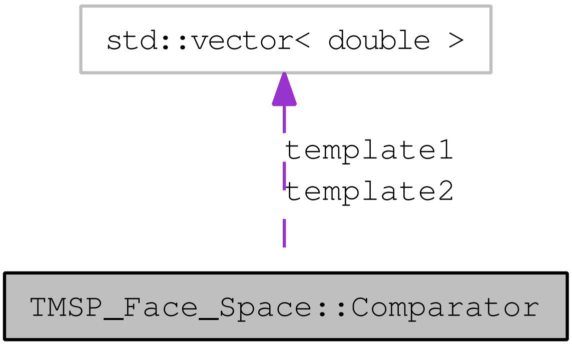
Pfc_Image object of type Im_8

17 Data Structure Documentation

17.1 TMSP_Face_Space::Comparator Class Reference

Class that compares two given templates based on the chosen Distance.

`#include <comparator.h>`Collaboration diagram for TMSP_Face_Space::Comparator:



Public Member Functions

- Comparator ()
- ~Comparator ()
- Comparator (vector< double > &template1, vector< double > &template2)
- Comparator (vector< double > &template1, vector< double > &template2, DISTANCE dist)
- DISTANCE Get_Distance ()
- void Set_Distance (DISTANCE dist)
- void Set_Distance (string dist)
- char GetMeasuretype ()
- double GetTemplatesDistance ()
- double GetTemplatesDistance (vector< double > &template1, vector< double > &template2, DISTANCE dist)
- double GetTemplatesDistance (vector< double > &template1, vector< double > &template2)

17.1.1 Detailed Description

Class that compares two given templates based on the chosen Distance.

17.1.2 Constructor & Destructor Documentation

17.1.2.1 TMSP_Face_Space::Comparator::Comparator ()

Constrcutor

17.1.2.2 TMSP_Face_Space::Comparator::~Comparator ()

Destructor

17.1.2.3 TMSP_Face_Space::Comparator::Comparator (vector< double > & *template1*, vector< double > & *template2*)

initialize the comparator with the two templates

Parameters:

template1 a vector of double (Reference)

template2 a vector of double (test)

17.1.2.4 TMSP_Face_Space::Comparator::Comparator (vector< double > & *template1*, vector< double > & *template2*, DISTANCE *dist*)

initialize the comparator with the two templates and the distance

Parameters:

template1 a vector of double (Reference)

template2 a vector of double (test)

dist Set the distance (C_L1, C_L2, C_Angle, C_NormDist)

17.1.3 Member Function Documentation

17.1.3.1 DISTANCE TMSP_Face_Space::Comparator::Get_Distance ()

Returns:

the distance chosen

17.1.3.2 char TMSP_Face_Space::Comparator::GetMeasuretype ()

Returns:

’S’ for similarity or ’D’ for distance

17.1.3.3 double TMSP_Face_Space::Comparator::GetTemplatesDistance (vector< double > & *template1*, vector< double > & *template2*)

Parameters:

template1 a vector of double (Reference)
template2 a vector of double (test)

Returns:

the score similarity or distance between the templates

17.1.3.4 double TMSP_Face_Space::Comparator::GetTemplatesDistance (vector< double > & *template1*, vector< double > & *template2*, DISTANCE *dist*)

Parameters:

template1 a vector of double (Reference)
template2 a vector of double (test)
dist

Returns:

the score similarity or distance between the templates

17.1.3.5 double TMSP_Face_Space::Comparator::GetTemplatesDistance ()

Returns:

the score similarity or distance between the templates

17.1.3.6 void TMSP_Face_Space::Comparator::Set_Distance (string *dist*)

Parameters:

dist Set the chosen distance (L1, L2, COS, NORMDIST)

17.1.3.7 void TMSP_Face_Space::Comparator::Set_Distance (DISTANCE *dist*)

Parameters:

dist Set the chosen distance (C_L1, C_L2, C_Angle, C_NormDist)

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

- lib/include/comparator.h
- lib/TMSPFace/comparator.cpp

17.2 TMSP_Face_Space::Face_Coordinates Struct Reference

Structure for storing the coordinates of Eyes, Nose and Mouth location in the original image.

```
#include <TMSP_image.h>
```

Data Fields

- int LeftEyeCenterX
- int LeftEyeCenterY
- int RightEyeCenterX
- int RightEyeCenterY
- int NoseCenterX
- int NoseCenterY
- int MouthCenterX
- int MouthCenterY

17.2.1 Detailed Description

Structure for storing the coordinates of Eyes, Nose and Mouth location in the original image.

17.2.2 Field Documentation

17.2.2.1 int TMSP_Face_Space::Face_Coordinates::LeftEyeCenterX

The X Coordinate of the center's left Eye

17.2.2.2 int TMSP_Face_Space::Face_Coordinates::LeftEyeCenterY

The Y Coordinate of the center's left Eye

17.2.2.3 int TMSP_Face_Space::Face_Coordinates::MouthCenterX

The X Coordinate of the center's mouth

17.2.2.4 int TMSP_Face_Space::Face_Coordinates::MouthCenterY

The Y Coordinate of the center's mouth

17.2.2.5 int TMSP_Face_Space::Face_Coordinates::NoseCenterX

The X Coordinate of the center's nose

17.2.2.6 int TMSP_Face_Space::Face_Coordinates::NoseCenterY

The Y Coordinate of the center's nose

17.2.2.7 int TMSP_Face_Space::Face_Coordinates::RightEyeCenterX

The X Coordinate of the center's right Eye

17.2.2.8 int TMSP_Face_Space::Face_Coordinates::RightEyeCenterY

The Y Coordinate of the center's right Eye

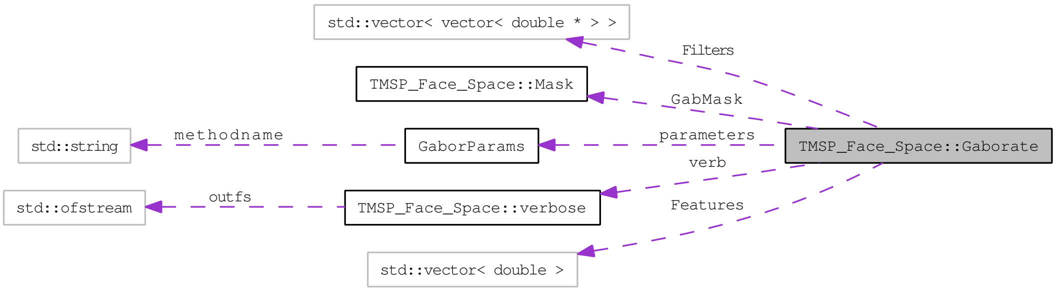
The documentation for this struct was generated from the following file:

- lib/include/TMSP_image.h

17.3 TMSP_Face_Space::Gaborate Class Reference

Class that computes the gabor filtering of an input image.

`#include <gaborate.h>`Collaboration diagram for TMSP_Face_Space::Gaborate:



Public Member Functions

- Gaborate ()
- Gaborate (GaborParams Gabparam)
- Gaborate (GaborParams Gabparam, const char ∗problem)
- ∼Gaborate ()
- void SetProblem (string P)
- void SetVerbose (verbose &v)
- int SetReduction (int redcoef)
- void SetGaborParams (int height, int width, int nscale, int norient, int minWaveLength, float mult, float sigmaOnf, float dThetaOnSigma)
- void SetGaborParams (GaborParams param)
- void CreateFFTGaborFilters (GaborParams param)
- void CreateFFTGaborFilters ()
- void SetFace (uint8_t ∗data, int rows, int cols)
- void SetFace (Pfc_Image &face)
- vector< double > Gaborating (uint8_t ∗data, int rows, int cols)
- vector< double > Gaborating (Pfc_Image &face)
- vector< double > Gaborating ()
- int GetFeaturesize ()
- string GetMethod ()
- vector< double > GetGaborFeatures ()
- void SetMask (int eyedist)
- void UnSetMask ()
- int SaveFeatures (const char ∗filename)
- int SaveFilters (const char ∗filename)
- void SetMutex (pthread_mutex_t ∗mux)

Data Fields

- GaborParams parameters

  Structure in which are stored the gabor filters parameters.

17.3.1 Detailed Description

Class that computes the gabor filtering of an input image. It creates the log-gabor filters in frequency space based on the work of Peter Kovesi http://www.csse.uwa.edu.au/~pk/research/matlabfns/PhaseCongruency/Docs/convexpl.html

17.3.2 Constructor & Destructor Documentation

17.3.2.1 TMSP_Face_Space::Gaborate::Gaborate ()

Constructor

17.3.2.2 TMSP_Face_Space::Gaborate::Gaborate (GaborParams *Gabparam*)

Constructor

Parameters:

Gabparam Structure in which are stored the gabor filters parameters.

17.3.2.3 TMSP_Face_Space::Gaborate::Gaborate (GaborParams *Gabparam*, const char $*$ *problem*)

Parameters:

Gabparam Structure in which are stored the gabor filters parameters

problem Complexe part to be computed:

- "REAL": real part of complex features
- "IMAG": imaginary part
- "MAGN": magnitude part
- "PHASE": angle part

17.3.2.4 TMSP_Face_Space::Gaborate::~Gaborate ()

Destructor

17.3.3 Member Function Documentation

17.3.3.1 void TMSP_Face_Space::Gaborate::CreateFFTGaborFilters ()

Create the gabor filters in frequency domain.

17.3.3.2 void TMSP_Face_Space::Gaborate::CreateFFTGaborFilters (GaborParams *param*)

Parameters:

param structure of gabor filters parameters to be copied to GaborParams structure of this class and create the gabor filters in frequency domain.

17.3.3.3 vector< double > TMSP_Face_Space::Gaborate::Gaborating ()

Returns:

a vector containing of the gabor components of the face based on the chosen gabor parts

17.3.3.4 vector< double > TMSP_Face_Space::Gaborate::Gaborating (Pfc_Image & *face*)

Parameters:

face Pfc_Image class used to be gaborated.

Returns:

a vector containing of the gabor components of data based on the chosen gabor part

17.3.3.5 vector< double > TMSP_Face_Space::Gaborate::Gaborating (uint8_t ∗ *data*, int *rows*, int *cols*)

Parameters:

data Pointer to the data of the gray-scale face image

rows height of the image

cols width of the image

Returns:

a vector containing of the gabor components of data based on the chosen gabor part

17.3.3.6 int TMSP_Face_Space::Gaborate::GetFeaturesize ()

Returns:

the size of the vector Features to be computed based on the given GaborParams

17.3.3.7 vector< double > TMSP_Face_Space::Gaborate::GetGaborFeatures ()

Returns:

return the vector of computed features

17.3.3.8 string TMSP_Face_Space::Gaborate::GetMethod ()

Returns:

return a string containing the used gabor parts

17.3.3.9 int TMSP_Face_Space::Gaborate::SaveFeatures (const char $*$ *filename*)

Parameters:

filename file name to output the computed features

Returns:

0 if well done

17.3.3.10 int TMSP_Face_Space::Gaborate::SaveFilters (const char $*$ *filename*)

Parameters:

filename name to output the log-gabor filters in Frequency domain.

Returns:

0 if well done

17.3.3.11 void TMSP_Face_Space::Gaborate::SetFace (Pfc_Image & *face*)

Parameters:

face the Pfc_Image is used to be gaborated.

17.3.3.12 void TMSP_Face_Space::Gaborate::SetFace (uint8_t $*$ *data*, int *rows*, int *cols*)

Parameters:

data Pointer to the data of the gray-scale face image

rows height of the image

cols width of the image

17.3.3.13 void TMSP_Face_Space::Gaborate::SetGaborParams (GaborParams *param*)

Parameters:

param structure of gabor filters parameters to be copied to GaborParams structure of this class.

17.3.3.14 void TMSP_Face_Space::Gaborate::SetGaborParams (int *height*, int *width*, int *nscale*, int *norient*, int *minWaveLength*, float *mult*, float *sigmaOnf*, float *dThetaOnSigma*)

Parameters:

height The height of the gabor filters

width The width of the gabor filters

nscale The number of scales of the gabor filters

norient The number of orientation of the gabor filters

minWaveLength The lentgh of the first scale gabor wavelet

mult To go from a scale wavelet to the next one the lentgh is multiplied by mult

sigmaOnf Ratio of the standard deviation of the Gaussian describing the log Gabor filter's transfer function in the frequency domain to the filter center frequency.

dThetaOnSigma Ratio of angular interval between filter orientations and the standard deviation of the angular Gaussian function used to construct filters in the freq. plane

17.3.3.15 void TMSP_Face_Space::Gaborate::SetMask (int *eyedist*)

Parameters:

eyedist create a mask to be applied to the gabor filtrerd response

17.3.3.16 void TMSP_Face_Space::Gaborate::SetMutex (pthread_mutex_t ∗ *mux*)

Parameters:

mux set mutex to mux to avoid fftw3 problem when using multi-threading.

17.3.3.17 void TMSP_Face_Space::Gaborate::SetProblem (string *P*)

Parameters:

P complexe part to be computed

17.3.3.18 int TMSP_Face_Space::Gaborate::SetReduction (int *redcoef*)

Parameters:

redcoef the coefficient sampling of the gabor features (note that for a reductcoef="2" just 1 gabor component on 2 is kept) a normalisation of the gabor features is done before the sampling to get mean=0 and std=1 for the real and imaginary parts.

Returns:

0 if well done

17.3.3.19 void TMSP_Face_Space::Gaborate::SetVerbose (verbose & *v*)

Parameters:

v copy the verbosing v to the member verb of ∗this;

17.3.3.20 void TMSP_Face_Space::Gaborate::UnSetMask ()

remove the mask

17.3.4 Field Documentation

17.3.4.1 GaborParams TMSP_Face_Space::Gaborate::parameters

Structure in which are stored the gabor filters parameters.

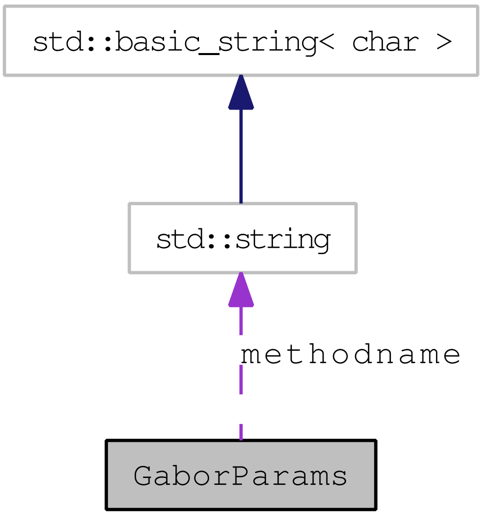
The documentation for this class was generated from the following files:

- lib/include/gaborate.h
- lib/TMSPFace/gaborate.cpp

17.4 GaborParams Struct Reference

Structure that stores the gabor filters parameters and the returned complexe part (real, imaginary, magnitude, phase).

`#include <gaborate.h>`Collaboration diagram for GaborParams:



Data Fields

- int height
- int width
- int nscale
- int norient
- int minWaveLength
- int reductcoef
- float mult
- float sigmaOnf
- float dThetaOnSigma
- string methodname

17.4.1 Detailed Description

Structure that stores the gabor filters parameters and the returned complexe part (real, imaginary, magnitude, phase).

17.4.2 Field Documentation

17.4.2.1 float GaborParams::dThetaOnSigma

Ratio of angular interval between filter orientations and the standard deviation of the angular Gaussian function used to construct filters in the freq. plane

17.4.2.2 int GaborParams::height

The height of the gabor filters

17.4.2.3 string GaborParams::methodname

Complexe part to be computed:

- "REAL": real part of complex features
- "IMAG": imaginary part
- "MAGN": magnitude part
- "PHASE": angle part
- a combination example methodname="REAL, IMAG, MAGN"

17.4.2.4 int GaborParams::minWaveLength

The lentgh of the first scale gabor wavelet

17.4.2.5 float GaborParams::mult

To go from a scale wavelet to the next one the lentgh is multiplied by mult

17.4.2.6 int GaborParams::norient

The number of orientation of the gabor filters

17.4.2.7 int GaborParams::nscale

The number of scales of the gabor filters

17.4.2.8 int GaborParams::reductcoef

the coefficient sampling of the gabor features (note that for a reductcoef="2" just 1 gabor component on 2 is kept)

17.4.2.9 float GaborParams::sigmaOnf

Ratio of the standard deviation of the Gaussian describing the log Gabor filter's transfer function in the frequency domain to the filter center frequency.

17.4.2.10 int GaborParams::width

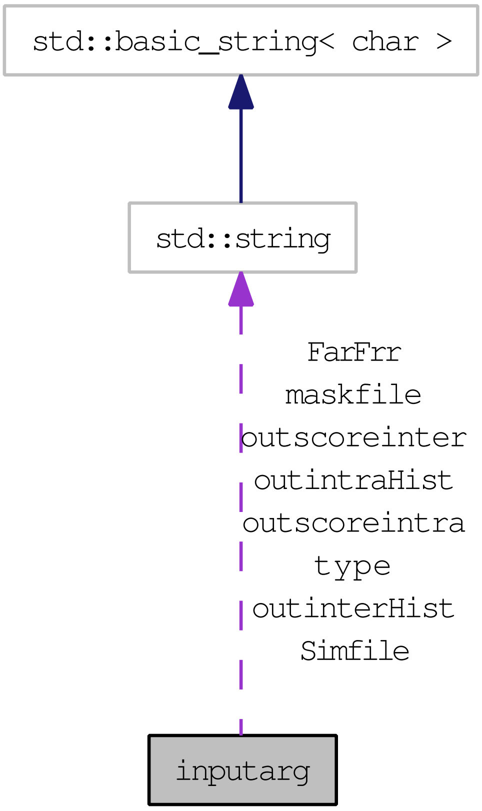
The width of the gabor filters

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

- lib/include/gaborate.h

17.5 inputarg Class Reference

Collaboration diagram for inputarg:



Public Member Functions

- inputarg ()
- ∼inputarg ()
- bool Getinputs (int ac, char ∗av[ ])
- void printNonnullinputs ()
- void printout (char ∗prog)

Data Fields

- string Simfile
- string maskfile
- string outintraHist
- string outinterHist
- string FarFrr
- string type
- bool feed
- float histstep
- bool outscores
- string outscoreintra
- string outscoreinter

17.5.1 Constructor & Destructor Documentation

17.5.1.1 inputarg::inputarg () `[inline]`

17.5.1.2 inputarg::∼inputarg () `[inline]`

17.5.2 Member Function Documentation

17.5.2.1 bool inputarg::Getinputs (int *ac*, char ∗ *av*[ ]) `[inline]`

17.5.2.2 void inputarg::printNonnullinputs () `[inline]`

17.5.2.3 void inputarg::printout (char ∗ *prog*) `[inline]`

17.5.3 Field Documentation

17.5.3.1 string inputarg::FarFrr

17.5.3.2 bool inputarg::feed

17.5.3.3 float inputarg::histstep

17.5.3.4 string inputarg::maskfile

17.5.3.5 string inputarg::outinterHist

17.5.3.6 string inputarg::outintraHist

17.5.3.7 string inputarg::outscoreinter

17.5.3.8 string inputarg::outscoreintra

17.5.3.9 bool inputarg::outscores

17.5.3.10 string inputarg::Simfile

17.5.3.11 string inputarg::type

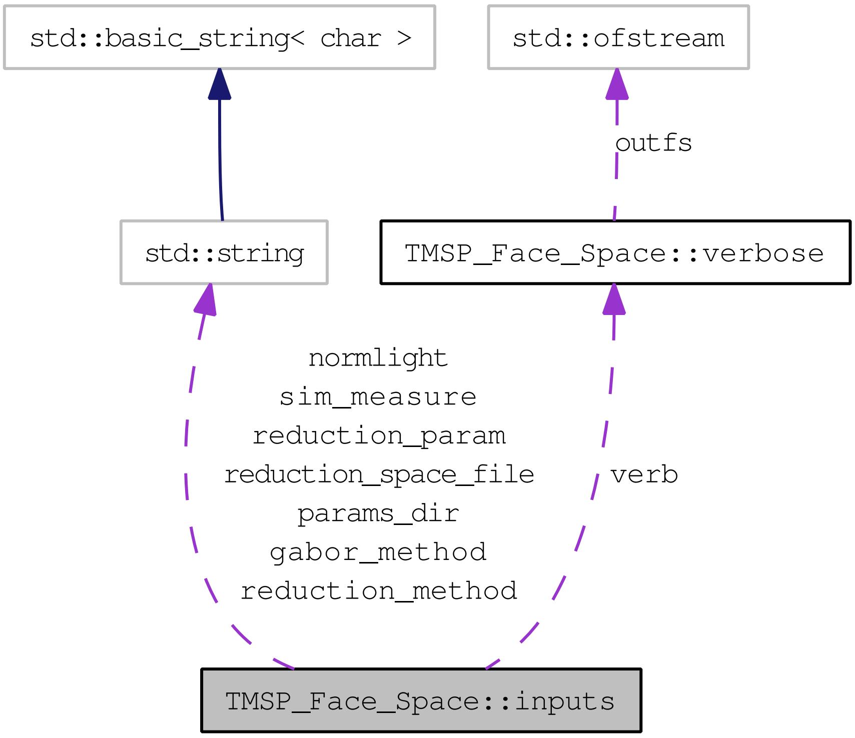
The documentation for this class was generated from the following file:

- bins/GetPerformance/GetPerformance.cpp

17.6 TMSP_Face_Space::inputs Class Reference

Class that reads framework parameters from the xml parameters file paramfile.xml .

`#include <inputs.h>`Collaboration diagram for TMSP_Face_Space::inputs:



Public Member Functions

- inputs ()
- inputs (const char ∗file)
- ∼inputs ()
- int loadfromxml (const char ∗file)
- vector< string > loadwatchlist (const char ∗pFilename)
- void SetVerbose (verbose &v)
- void SetVerbose ()
- void SetMute ()

Data Fields

- string params_dir
- string normlight
- string reduction_method
- string reduction_param
- string gabor_method
- string reduction_space_file
- string sim_measure
- int nthreads
- int verboz
- int norm_eye_dist
- int Gabscales

- int Gaborientations
- int GabminWavelet
- int Gabreduction
- int controlled
- int noncontrolled
- int nusers
- float GabsigmaOnf
- float GabdThetaOnSigma
- float Gabmult
- double ∗ paramnorm

17.6.1 Detailed Description

Class that reads framework parameters from the xml parameters file paramfile.xml .

17.6.2 Constructor & Destructor Documentation

17.6.2.1 TMSP_Face_Space::inputs::inputs ()

Constructor

17.6.2.2 TMSP_Face_Space::inputs::inputs (const char ∗ *file*)

Constructor

Parameters:

file

17.6.2.3 TMSP_Face_Space::inputs::∼inputs ()

Destructor

17.6.3 Member Function Documentation

17.6.3.1 int TMSP_Face_Space::inputs::loadfromxml (const char ∗ *file*)

Parameters:

file

Returns:

17.6.3.2 vector< string > TMSP_Face_Space::inputs::loadwatchlist (const char $*$ *pFilename*)

Parameters:

pFilename

Returns:

17.6.3.3 void TMSP_Face_Space::inputs::SetMute ()

17.6.3.4 void TMSP_Face_Space::inputs::SetVerbose ()

17.6.3.5 void TMSP_Face_Space::inputs::SetVerbose (verbose & *v*)

Parameters:

v

17.6.4 Field Documentation

17.6.4.1 int TMSP_Face_Space::inputs::controlled

17.6.4.2 float TMSP_Face_Space::inputs::GabdThetaOnSigma

17.6.4.3 int TMSP_Face_Space::inputs::GabminWavelet

17.6.4.4 float TMSP_Face_Space::inputs::Gabmult

17.6.4.5 string TMSP_Face_Space::inputs::gabor_method

17.6.4.6 int TMSP_Face_Space::inputs::Gaborientations

17.6.4.7 int TMSP_Face_Space::inputs::Gabreduction

17.6.4.8 int TMSP_Face_Space::inputs::Gabscales

17.6.4.9 float TMSP_Face_Space::inputs::GabsigmaOnf

17.6.4.10 int TMSP_Face_Space::inputs::noncontrolled

17.6.4.11 int TMSP_Face_Space::inputs::norm_eye_dist

17.6.4.12 string TMSP_Face_Space::inputs::normlight

17.6.4.13 int TMSP_Face_Space::inputs::nthreads

17.6.4.14 int TMSP_Face_Space::inputs::nusers

17.6.4.15 double∗ TMSP_Face_Space::inputs::paramnorm

17.6.4.16 string TMSP_Face_Space::inputs::params_dir

17.6.4.17 string TMSP_Face_Space::inputs::reduction_method

17.6.4.18 string TMSP_Face_Space::inputs::reduction_param

17.6.4.19 string TMSP_Face_Space::inputs::reduction_space_file

17.6.4.20 string TMSP_Face_Space::inputs::sim_measure

17.6.4.21 int TMSP_Face_Space::inputs::verboz

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

- lib/include/inputs.h
- lib/TMSPFace/inputs.cpp

17.7 TMSP_Face_Space::kerlin Struct Reference

structure that stores a Linear kernel parameters Y = X∗alpha + decal;

```
#include <kernel.h>
```

Data Fields

- float alpha
- float decal

17.7.1 Detailed Description

structure that stores a Linear kernel parameters Y = X∗alpha + decal;

17.7.2 Field Documentation

17.7.2.1 float TMSP_Face_Space::kerlin::alpha

17.7.2.2 float TMSP_Face_Space::kerlin::decal

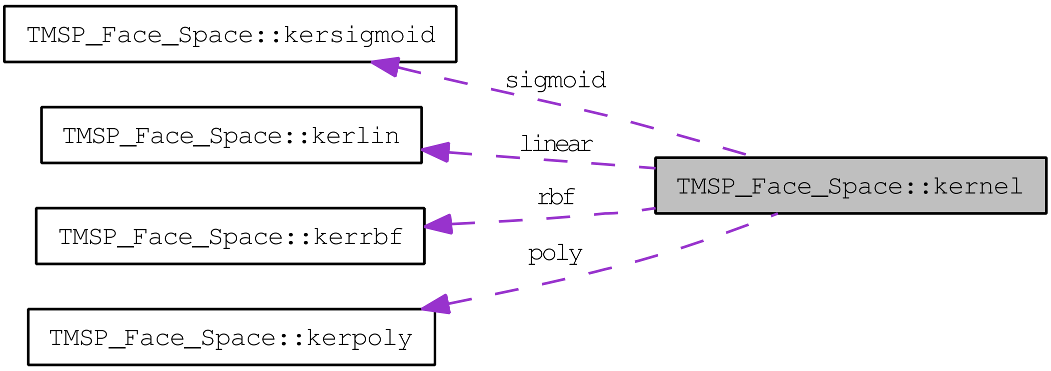
The documentation for this struct was generated from the following file:

- lib/include/kernel.h

17.8 TMSP_Face_Space::kernel Class Reference

Class that create kernels and implement kernel methods.

`#include <kernel.h>`Collaboration diagram for TMSP_Face_Space::kernel:



Public Member Functions

- kernel ()
- kernel (GeneralMatrix &Data)
- kernel (GeneralMatrix &Data, kerlin k)
- kernel (GeneralMatrix &Data, kerpoly k)
- kernel (GeneralMatrix &Data, kerrbf k)
- kernel (GeneralMatrix &Data, kersigmoid k)
- kernel (GeneralMatrix &Data1, GeneralMatrix &Data2)
- kernel (GeneralMatrix &Data1, GeneralMatrix &Data2, kerlin k)
- kernel (GeneralMatrix &Data1, GeneralMatrix &Data2, kerpoly k)
- kernel (GeneralMatrix &Data1, GeneralMatrix &Data2, kerrbf k)
- kernel (GeneralMatrix &Data1, GeneralMatrix &Data2, kersigmoid k)
- ~kernel ()
- void SetData (const Matrix &Data)
- void SetData (const Matrix &Data1, const Matrix &Data2)
- void SetFData (const Matrix &Data1)
- void SetSData (const Matrix &Data1)
- void settype (int T)
- void setker (kerlin k)
- void setker (kerpoly k)
- void setker (kerrbf k)
- void setker (kersigmoid k)
- void Computekernel ()
- kernel & operator= (const kernel &k)
- void Setmaxmin (float _min, float _max)

Data Fields

- Matrix DonneesA
- Matrix DonneesB
- Matrix noyau
- int type
- float maxR
- float minR
- struct kerlin linear
- struct kerpoly poly
- struct kerrbf rbf
- struct kersigmoid sigmoid

17.8.1 Detailed Description

Class that create kernels and implement kernel methods.

17.8.2 Constructor & Destructor Documentation

17.8.2.1 TMSP_Face_Space::kernel::kernel ()

17.8.2.2 TMSP_Face_Space::kernel::kernel (GeneralMatrix & *Data*)

Parameters:

Data

17.8.2.3 TMSP_Face_Space::kernel::kernel (GeneralMatrix & *Data*, kerlin *k*)

Parameters:

Data

k

Data

k

17.8.2.4 TMSP_Face_Space::kernel::kernel (GeneralMatrix & *Data*, kerpoly *k*)

Parameters:

Data

k

17.8.2.5 TMSP_Face_Space::kernel::kernel (GeneralMatrix & *Data*, kerrbf *k*)

Parameters:

Data

k

17.8.2.6 TMSP_Face_Space::kernel::kernel (GeneralMatrix & *Data*, kersigmoid *k*)

Parameters:

Data

k

17.8.2.7 TMSP_Face_Space::kernel::kernel (GeneralMatrix & *Data1*, GeneralMatrix & *Data2*)

Parameters:

Data1

Data2

17.8.2.8 TMSP_Face_Space::kernel::kernel (GeneralMatrix & *Data1*, GeneralMatrix & *Data2*, kerlin *k*)

Parameters:

Data1

Data2

k

17.8.2.9 TMSP_Face_Space::kernel::kernel (GeneralMatrix & *Data1*, GeneralMatrix & *Data2*, kerpoly *k*)

Parameters:

Data1

Data2

k

17.8.2.10 TMSP_Face_Space::kernel::kernel (GeneralMatrix & *Data1*, GeneralMatrix & *Data2*, kerrbf *k*)

Parameters:

Data1

Data2

k

17.8.2.11 TMSP_Face_Space::kernel::kernel (GeneralMatrix & *Data1*, GeneralMatrix & *Data2*, kersigmoid *k*)

Parameters:

Data1

Data2

k

17.8.2.12 TMSP_Face_Space::kernel::∼kernel ()

17.8.3 Member Function Documentation

17.8.3.1 void TMSP_Face_Space::kernel::Computekernel ()

17.8.3.2 kernel & TMSP_Face_Space::kernel::operator= (const kernel & *k*)

Parameters:

k

Returns:

17.8.3.3 void TMSP_Face_Space::kernel::SetData (const Matrix & *Data1*, const Matrix & *Data2*)

Parameters:

Data1

Data2

17.8.3.4 void TMSP_Face_Space::kernel::SetData (const Matrix & *Data*)

Parameters:

Data

17.8.3.5 void TMSP_Face_Space::kernel::SetFData (const Matrix & *Data1*)

Parameters:

Data1

17.8.3.6 void TMSP_Face_Space::kernel::setker (kersigmoid *k*)

Parameters:

k

17.8.3.7 void TMSP_Face_Space::kernel::setker (kerrbf *k*)

Parameters:

k

17.8.3.8 void TMSP_Face_Space::kernel::setker (kerpoly *k*)

Parameters:

k

17.8.3.9 void TMSP_Face_Space::kernel::setker (kerlin *k*)

Parameters:

k

17.8.3.10 void TMSP_Face_Space::kernel::Setmaxmin (float *_min*, float *_max*)

Parameters:

_min

_max

17.8.3.11 void TMSP_Face_Space::kernel::SetSData (const Matrix & *Data1*)

Parameters:

Data1

17.8.3.12 void TMSP_Face_Space::kernel::settype (int *T*)

Parameters:

T

17.8.4 Field Documentation

17.8.4.1 Matrix TMSP_Face_Space::kernel::DonneesA

17.8.4.2 Matrix TMSP_Face_Space::kernel::DonneesB

17.8.4.3 struct kerlin TMSP_Face_Space::kernel::linear `[read]`

17.8.4.4 float TMSP_Face_Space::kernel::maxR

17.8.4.5 float TMSP_Face_Space::kernel::minR

17.8.4.6 Matrix TMSP_Face_Space::kernel::noyau

17.8.4.7 struct kerpoly TMSP_Face_Space::kernel::poly `[read]`

17.8.4.8 struct kerrbf TMSP_Face_Space::kernel::rbf `[read]`

17.8.4.9 struct kersigmoid TMSP_Face_Space::kernel::sigmoid `[read]`

17.8.4.10 int TMSP_Face_Space::kernel::type

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

- lib/include/kernel.h
- lib/TMSPFace/kernel.cpp

17.9 TMSP_Face_Space::kerpoly Struct Reference

structure that stores a Polynomial kernel parameters Y = pow ((X + decal),power)

```
#include <kernel.h>
```

Data Fields

- float decal
- float power

17.9.1 Detailed Description

structure that stores a Polynomial kernel parameters Y = pow ((X + decal),power)

17.9.2 Field Documentation

17.9.2.1 float TMSP_Face_Space::kerpoly::decal

17.9.2.2 float TMSP_Face_Space::kerpoly::power

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

- lib/include/kernel.h

17.10 TMSP_Face_Space::kerrbf Struct Reference

structure that stores a Radial Basis Function kernel parameters Y= exp (-X∗power) ;

`#include <kernel.h>`

Data Fields

- float power

17.10.1 Detailed Description

structure that stores a Radial Basis Function kernel parameters Y= exp (-X∗power) ;

17.10.2 Field Documentation

17.10.2.1 float TMSP_Face_Space::kerrbf::power

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

- lib/include/kernel.h

17.11 TMSP_Face_Space::kersigmoid Struct Reference

structure that stores a Sigmoid kernel parameters Y=tanh (X∗gamma + decal);

`#include <kernel.h>`

Data Fields

- float decal
- float gamma

17.11.1 Detailed Description

structure that stores a Sigmoid kernel parameters Y=tanh (X∗gamma + decal);

17.11.2 Field Documentation

17.11.2.1 float TMSP_Face_Space::kersigmoid::decal

17.11.2.2 float TMSP_Face_Space::kersigmoid::gamma

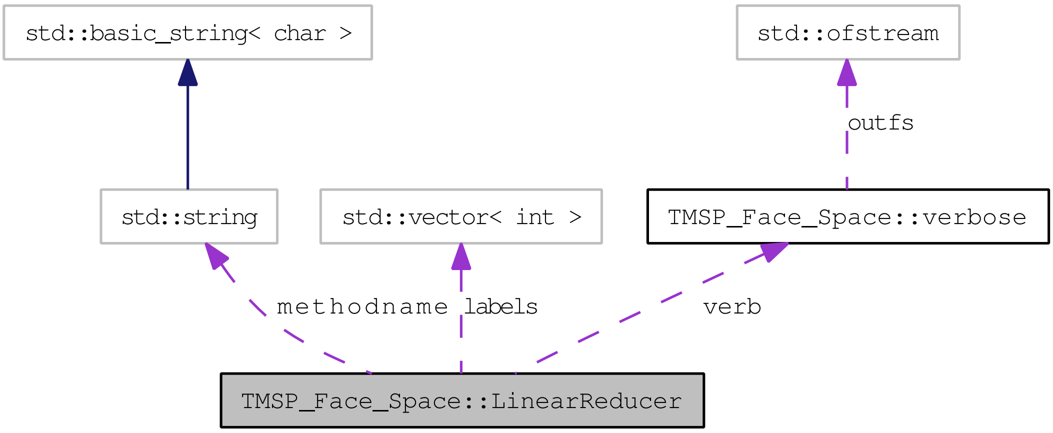
The documentation for this struct was generated from the following file:

- lib/include/kernel.h

17.12 TMSP_Face_Space::LinearReducer Class Reference

Class used to reduce the input space by linear methods (PCA,LDA,DLDA).

`#include <linearreducer.h>`Collaboration diagram for TMSP_Face_Space::LinearReducer:



Public Member Functions

- LinearReducer ()
- LinearReducer (Matrix &A)
- LinearReducer (Matrix &A, LProblem w)
- LinearReducer (Matrix &A, vector< int > lab)
- LinearReducer (LProblem w)
- ∼LinearReducer ()
- void free ()
- void Init ()
- void SetData (Matrix &A)
- void SetProblem (LProblem w)
- void SetProblem (string w)
- void SetLabels (vector< int > labs)
- LProblem GetProblem ()

- string GetProblemName ()
- bool GetPCACompound ()
- bool GetLDACompound ()
- bool GetDLDACompound ()
- void GetNonZeroEigVal ()
- void GetNonZeroEigVal (int NVal)
- void GetNonZeroEigVect (int NVect)
- int ComputeEigens (GeneralMatrix &CM, DiagonalMatrix &D)
- int ComputeSpace ()
- void SaveSpace (char ∗filename)
- void SaveSpace (char ∗filename, GaborParams &params)
- void LoadSpace (const char ∗filename)
- ReturnMatrix Projection (ColumnVector &face)
- double Distance (ColumnVector &face1, ColumnVector &face2, string Method)
- double Distance (ColumnVector &face1, ColumnVector &face2, string Method, ColumnVector &wheigts)
- void SetSpaceUsedSize (long size)
- long GetSpaceSize ()
- ReturnMatrix GetEigensVariance ()
- int GetVarianceCount ()
- void Setmaxvariance (string line)
- void Setmaxvariance (float var)
- float Getmaxvariance ()
- void ReduceToVariance ()
- ReturnMatrix Reconstruction (ColumnVector &Comp)
- void SetVerbose (verbose &mx)

Data Fields

- ColumnVector MeanFace
- ColumnVector EigenValues
- Matrix EigenVectors

17.12.1 Detailed Description

Class used to reduce the input space by linear methods (PCA,LDA,DLDA).

17.12.2 Constructor & Destructor Documentation

17.12.2.1 TMSP_Face_Space::LinearReducer::LinearReducer ()

Constructor

17.12.2.2 TMSP_Face_Space::LinearReducer::LinearReducer (Matrix & *A*)

Parameters:

A

17.12.2.3 TMSP_Face_Space::LinearReducer::LinearReducer (Matrix & *A*, LProblem *w*)

Parameters:

A

w

17.12.2.4 TMSP_Face_Space::LinearReducer::LinearReducer (Matrix & *A*, vector< int > *lab*)

Parameters:

A

lab

17.12.2.5 TMSP_Face_Space::LinearReducer::LinearReducer (LProblem *w*)

Parameters:

w

17.12.2.6 TMSP_Face_Space::LinearReducer::∼LinearReducer ()

17.12.3 Member Function Documentation

17.12.3.1 int TMSP_Face_Space::LinearReducer::ComputeEigens (GeneralMatrix & *CM*, DiagonalMatrix & *D*)

Parameters:

CM

D

Returns:

17.12.3.2 int TMSP_Face_Space::LinearReducer::ComputeSpace ()

Returns:

17.12.3.3 double TMSP_Face_Space::LinearReducer::Distance (ColumnVector & *face1*, ColumnVector & *face2*, string *Method*, ColumnVector & *wheigts*)

Parameters:

face1

face2

Method

wheigts

Returns:

17.12.3.4 double TMSP_Face_Space::LinearReducer::Distance (ColumnVector & *face1*, ColumnVector & *face2*, string *Method*)

Parameters:

face1

face2

Method

Returns:

17.12.3.5 void TMSP_Face_Space::LinearReducer::free ()

17.12.3.6 bool TMSP_Face_Space::LinearReducer::GetDLDACompound ()

Returns:

17.12.3.7 ReturnMatrix TMSP_Face_Space::LinearReducer::GetEigensVariance ()

Returns:

17.12.3.8 bool TMSP_Face_Space::LinearReducer::GetLDACompound ()

Returns:

17.12.3.9 float TMSP_Face_Space::LinearReducer::Getmaxvariance ()

Returns:

17.12.3.10 void TMSP_Face_Space::LinearReducer::GetNonZeroEigVal (int *NVal*)

Parameters:

NVal

17.12.3.11 void TMSP_Face_Space::LinearReducer::GetNonZeroEigVal ()

17.12.3.12 void TMSP_Face_Space::LinearReducer::GetNonZeroEigVect (int *NVect*)

Parameters:

NVect

17.12.3.13 bool TMSP_Face_Space::LinearReducer::GetPCACompound ()

Returns:

17.12.3.14 LProblem TMSP_Face_Space::LinearReducer::GetProblem ()

Returns:

17.12.3.15 string TMSP_Face_Space::LinearReducer::GetProblemName ()

Returns:

17.12.3.16 long TMSP_Face_Space::LinearReducer::GetSpaceSize ()

Returns:

17.12.3.17 int TMSP_Face_Space::LinearReducer::GetVarianceCount ()

Returns:

17.12.3.18 void TMSP_Face_Space::LinearReducer::Init ()

17.12.3.19 void TMSP_Face_Space::LinearReducer::LoadSpace (const char ∗*filename*)

Parameters:

filename

17.12.3.20 ReturnMatrix TMSP_Face_Space::LinearReducer::Projection (ColumnVector &*face*)

Parameters:

face

Returns:

17.12.3.21 ReturnMatrix TMSP_Face_Space::LinearReducer::Reconstruction (ColumnVector & *Comp*)

Parameters:

Comp

Returns:

17.12.3.22 void TMSP_Face_Space::LinearReducer::ReduceToVariance ()

17.12.3.23 void TMSP_Face_Space::LinearReducer::SaveSpace (char ∗ *filename*, GaborParams & *params*)

Parameters:

filename

params

17.12.3.24 void TMSP_Face_Space::LinearReducer::SaveSpace (char ∗ *filename*)

Parameters:

filename

17.12.3.25 void TMSP_Face_Space::LinearReducer::SetData (Matrix & *A*)

Parameters:

A

17.12.3.26 void TMSP_Face_Space::LinearReducer::SetLabels (vector< int > *labs*)

Parameters:

labs

17.12.3.27 void TMSP_Face_Space::LinearReducer::Setmaxvariance (float *var*)

Parameters:

var

17.12.3.28 void TMSP_Face_Space::LinearReducer::Setmaxvariance (string *line*)

Parameters:

line

17.12.3.29 void TMSP_Face_Space::LinearReducer::SetProblem (string *w*)

Parameters:

w

17.12.3.30 void TMSP_Face_Space::LinearReducer::SetProblem (LProblem *w*)

Parameters:

w

17.12.3.31 void TMSP_Face_Space::LinearReducer::SetSpaceUsedSize (long *size*)

Parameters:

size

17.12.3.32 void TMSP_Face_Space::LinearReducer::SetVerbose (verbose & *mx*)

Parameters:

mx

17.12.4 Field Documentation

17.12.4.1 ColumnVector TMSP_Face_Space::LinearReducer::EigenValues

eigenvalues

17.12.4.2 Matrix TMSP_Face_Space::LinearReducer::EigenVectors

Eigen vectors (Eigen faces)

17.12.4.3 ColumnVector TMSP_Face_Space::LinearReducer::MeanFace

the mean face

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

- lib/include/linearreducer.h
- lib/TMSPFace/linearreducer.cpp

17.13 TMSP_Face_Space::Mask Class Reference

Class that creates an elliptic mask to be applied to faces.

```
#include <mask.h>
```

Public Member Functions

- Mask ()
- Mask (int EYE_DISTANCE)
- ∼Mask ()
- void Init (int EYE_DISTANCE)
- bool Ismask (const int &x, const int &y)
- void ApplyMask (double ∗&imageout, int imageRows, int imageCols)
- int getwidth ()
- int getheight ()

17.13.1 Detailed Description

Class that creates an elliptic mask to be applied to faces.

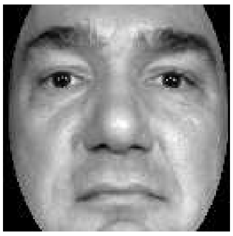


Figure 5: Face normalized to 50 pixels between eyes and with elliptic mask

17.13.2 Constructor & Destructor Documentation

17.13.2.1 TMSP_Face_Space::Mask::Mask ()

Constructor

17.13.2.2 TMSP_Face_Space::Mask::Mask (int *EYE_DISTANCE*)

Constructor initialize the mask with the eyes distance

Parameters:

EYE_DISTANCE distance between eyes' centers

17.13.2.3 TMSP_Face_Space::Mask::∼Mask ()

Destructor

17.13.3 Member Function Documentation

17.13.3.1 void TMSP_Face_Space::Mask::ApplyMask (double ∗& *imageout*, int *imageRows*, int *imageCols*)

apply mask (set to zero the non mask point)

Parameters:

imageout a pointer to a double data

imageRows height of the data matrix

imageCols width of the data matrix

17.13.3.2 int TMSP_Face_Space::Mask::getheight ()

Returns:

the mask's height

17.13.3.3 int TMSP_Face_Space::Mask::getwidth ()

Returns:

the mask's width

17.13.3.4 void TMSP_Face_Space::Mask::Init (int *EYE_DISTANCE*)

initialize the mask with the eyes distance

Parameters:

EYE_DISTANCE

17.13.3.5 bool TMSP_Face_Space::Mask::Ismask (const int & *x*, const int & *y*)

test if the point belgons to the mask or not

Parameters:

x point x coordinate

y point y coordinate

Returns:

True if the point belongs else False

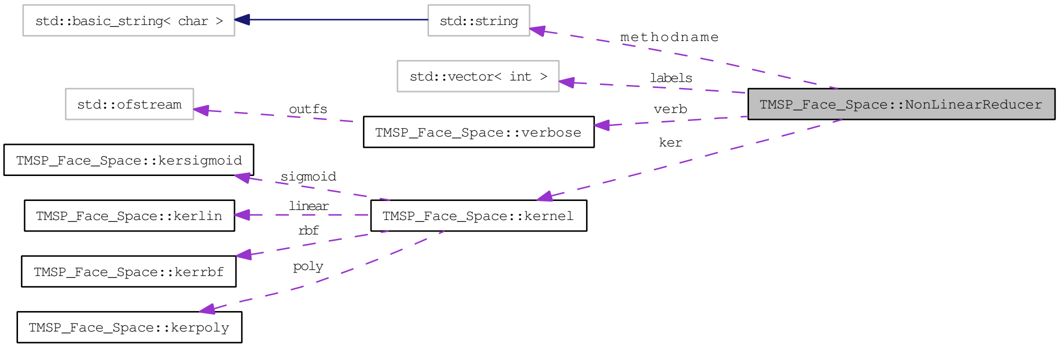
The documentation for this class was generated from the following files:

- lib/include/mask.h
- lib/TMSPFace/mask.cpp

17.14 TMSP_Face_Space::NonLinearReducer Class Reference

Class used to reduce the input space by nonlinear methods using kernel approaches (KFA,GDA).

`#include <nonlinearreducer.h>`Collaboration diagram for TMSP_Face_Space::NonLinearReducer:



Public Member Functions

- NonLinearReducer ()
- NonLinearReducer (Matrix &A, vector< int > &L)
- NonLinearReducer (Matrix &A, vector< int > &L, kernel &k)
- NonLinearReducer (Matrix &A, vector< int > &L, kerlin &k)

- NonLinearReducer (Matrix &A, vector< int > &L, kerpoly &k)
- NonLinearReducer (Matrix &A, vector< int > &L, kerrbf &k)
- NonLinearReducer (Matrix &A, vector< int > &L, kersigmoid &k)
- ∼NonLinearReducer ()
- void SetDatawithLabels (Matrix &A, vector< int > &L)
- void Set (Matrix &A, vector< int > &L, kerlin &k)
- void Set (Matrix &A, vector< int > &L, kerpoly &k)
- void Set (Matrix &A, vector< int > &L, kerrbf &k)
- void Set (Matrix &A, vector< int > &L, kersigmoid &k)
- int ComputeSpace ()
- int GetKFACompound ()
- int GetGDACompound ()
- ReturnMatrix Projection (GeneralMatrix &Datain)
- void SaveSpace (char ∗filename)
- void SaveSpace (char ∗filename, GaborParams &params)
- void LoadSpace (const char ∗filename)
- void SetProblem (string w)
- void SetProblem (NProblem w)
- NProblem GetProblem ()
- string GetProblemName ()
- void SetVerbose (verbose &v)
- void SetkernelPartFromline (string line)

Data Fields

- kernel ker
- vector< int > labels
- Matrix EigenVectors
- ColumnVector bias

17.14.1 Detailed Description

Class used to reduce the input space by nonlinear methods using kernel approaches (KFA,GDA).

17.14.2 Constructor & Destructor Documentation

17.14.2.1 TMSP_Face_Space::NonLinearReducer::NonLinearReducer ()

Constructor

17.14.2.2 TMSP_Face_Space::NonLinearReducer::NonLinearReducer (Matrix & *A*, vector< int > & *L*)

Parameters:

A

L

17.14.2.3 TMSP_Face_Space::NonLinearReducer::NonLinearReducer (Matrix & *A*, vector< int > & *L*, kernel & *k*)

Parameters:

A

L

k

17.14.2.4 TMSP_Face_Space::NonLinearReducer::NonLinearReducer (Matrix & *A*, vector< int > & *L*, kerlin & *k*)

Parameters:

A

L

k

17.14.2.5 TMSP_Face_Space::NonLinearReducer::NonLinearReducer (Matrix & *A*, vector< int > & *L*, kerpoly & *k*)

Parameters:

A

L

k

17.14.2.6 TMSP_Face_Space::NonLinearReducer::NonLinearReducer (Matrix & *A*, vector< int > & *L*, kerrbf & *k*)

Parameters:

A

L

k

17.14.2.7 TMSP_Face_Space::NonLinearReducer::NonLinearReducer (Matrix & *A*, vector< int > & *L*, kersigmoid & *k*)

Parameters:

A

L

k

17.14.2.8 TMSP_Face_Space::NonLinearReducer::∼NonLinearReducer ()

17.14.3 Member Function Documentation

17.14.3.1 int TMSP_Face_Space::NonLinearReducer::ComputeSpace ()

Returns:

17.14.3.2 int TMSP_Face_Space::NonLinearReducer::GetGDACompound ()

Returns:

17.14.3.3 int TMSP_Face_Space::NonLinearReducer::GetKFACompound ()

Returns:

17.14.3.4 NProblem TMSP_Face_Space::NonLinearReducer::GetProblem ()

Returns:

17.14.3.5 string TMSP_Face_Space::NonLinearReducer::GetProblemName ()

Returns:

17.14.3.6 void TMSP_Face_Space::NonLinearReducer::LoadSpace (const char ∗ *filename*)

Parameters:

filename

17.14.3.7 ReturnMatrix TMSP_Face_Space::NonLinearReducer::Projection (GeneralMatrix & *Datain*)

Parameters:

Datain

Returns:

17.14.3.8 void TMSP_Face_Space::NonLinearReducer::SaveSpace (char ∗ *filename*, GaborParams & *params*)

Parameters:

filename

params

17.14.3.9 void TMSP_Face_Space::NonLinearReducer::SaveSpace (char ∗ *filename*)

Parameters:

filename

17.14.3.10 void TMSP_Face_Space::NonLinearReducer::Set (Matrix & *A*, vector$<$ int $>$ & *L*, kersigmoid & *k*)

Parameters:

A

L

k

17.14.3.11 void TMSP_Face_Space::NonLinearReducer::Set (Matrix & *A*, vector$<$ int $>$ & *L*, kerrbf & *k*)

Parameters:

A

L

k

17.14.3.12 void TMSP_Face_Space::NonLinearReducer::Set (Matrix & *A*, vector$<$ int $>$ & *L*, kerpoly & *k*)

Parameters:

A

L

k

17.14.3.13 void TMSP_Face_Space::NonLinearReducer::Set (Matrix & *A*, vector$<$ int $>$ & *L*, kerlin & *k*)

Parameters:

A

L

k

17.14.3.14 void TMSP_Face_Space::NonLinearReducer::SetDatawithLabels (Matrix & *A*, vector< int > & *L*)

Parameters:

A

L

17.14.3.15 void TMSP_Face_Space::NonLinearReducer::SetkernelPartFromline (string *line*)

Parameters:

line

17.14.3.16 void TMSP_Face_Space::NonLinearReducer::SetProblem (NProblem *w*)

Parameters:

w

17.14.3.17 void TMSP_Face_Space::NonLinearReducer::SetProblem (string *w*)

Parameters:

w

17.14.3.18 void TMSP_Face_Space::NonLinearReducer::SetVerbose (verbose & *v*)

Parameters:

v

17.14.4 Field Documentation

17.14.4.1 ColumnVector TMSP_Face_Space::NonLinearReducer::bias

17.14.4.2 Matrix TMSP_Face_Space::NonLinearReducer::EigenVectors

17.14.4.3 kernel TMSP_Face_Space::NonLinearReducer::ker

17.14.4.4 vector<int> TMSP_Face_Space::NonLinearReducer::labels

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

- lib/include/nonlinearreducer.h
- lib/TMSPFace/nonlinearreducer.cpp

17.15 TMSP_Face_Space::Pfc_Image Class Reference

Class that allows the manipulation of image for the baseline.

```
#include <TMSP_image.h>
```

Public Member Functions

- Pfc_Image ()
- Pfc_Image (const char ∗filename)
- Pfc_Image (uint8_t ∗Dataptr, int width, int height, int Bytebp)
- Pfc_Image (uint8_t ∗Dataptr, int width, int height)
- Pfc_Image (int w, int h)
- ∼Pfc_Image ()
- int GetImWidth ()
- int GetImHeight ()
- int GetImSize ()
- void SetImWidth (int w)
- void SetImHeight (int h)
- void SetImDim (int w, int h)
- void SetImParam (int w, int h, int Bb)
- void SetImBytebp (int Bb)
- int GetImBytebp ()
- uint8_t maximum ()
- uint8_t minimum ()
- int AllocImData (int width, int height, int Bytebp)
- uint8_t ∗ GetDataptr ()
- void SetData (int pos, uint8_t val)
- uint8_t GetData (int pos)
- void SetImData (uint8_t ∗Dataptr, int width, int height, int Bytebp)
- Pfc_Image & operator= (const Pfc_Image &o)
- void copy (const Pfc_Image &o)
- void Stretch (int method)

- void Stretch (int method, Mask &ROI)
- void Histeq ()
- std::vector< int > GetHistogram ()
- std::vector< int > GetCumHistogramme ()
- void Rotate90 ()
- int ReadImage (const char ∗filename)
- int ReadImage (const char ∗filename, Extension ext)
- int ReadPpm (const char ∗filename)
- int ReadPgm (const char ∗filename)
- int ReadJpeg (const char ∗filename)
- Pfc_Image ∗ LightCorrect (string method, double ∗param)
- Pfc_Image ∗ LightCorrect (string methodname, double ∗param, Mask ROI)
- Pfc_Image ∗ LightCorrect (LightEnhance method, double ∗param)
- Pfc_Image ∗ LightCorrect (LightEnhance method, double ∗param, Mask ROI)
- void SaveImage (char ∗filename)
- float sum_square ()
- float mean ()
- float std ()
- uint8_t operator() (int i, int j)
- ReturnMatrix MatFromPFCImage ()

  return a pointer to a Matrix from Im_8 Pfc_Image data

17.15.1 Detailed Description

Class that allows the manipulation of image for the baseline.

17.15.2 Constructor & Destructor Documentation

17.15.2.1 TMSP_Face_Space::Pfc_Image::Pfc_Image ()

Constructor

17.15.2.2 TMSP_Face_Space::Pfc_Image::Pfc_Image (const char ∗ *filename*)

Constructor that initialize the class by loading image from a file

Parameters:

filename file name of the image to be loaded (supported format JPEG, PGM and PPM)

17.15.2.3 TMSP_Face_Space::Pfc_Image::Pfc_Image (uint8_t ∗ *Dataptr*, int *width*, int *height*, int *Bytebp*)

Constructor that initialize the class from a memory block

Parameters:

Dataptr pointer to the image data

width width of the image

height height of the image

Bytebp number of Byte by pixel

17.15.2.4 TMSP_Face_Space::Pfc_Image::Pfc_Image (uint8_t ∗ *Dataptr*, int *width*, int *height*)

Constructor that initialize the class from a memory block supposing that the image is in grey level

Parameters:

Dataptr pointer to the image data

width width of the image

height height of the image

17.15.2.5 TMSP_Face_Space::Pfc_Image::Pfc_Image (int *w*, int *h*)

Constructor that initialize the class by allocating a memory block of size=w∗h

Parameters:

w width of the image

h height of the image

17.15.2.6 TMSP_Face_Space::Pfc_Image::∼Pfc_Image ()

Destructor

17.15.3 Member Function Documentation

17.15.3.1 int TMSP_Face_Space::Pfc_Image::AllocImData (int *width*, int *height*, int *Bytebp*)

allocate a memory space for the image (width∗height∗Bytebp)

Parameters:

width width of the image

height height of the image

Bytebp bytesbypixel of the image

Returns:

17.15.3.2 void TMSP_Face_Space::Pfc_Image::copy (const Pfc_Image & *o*)

fill the image object with the same data of another Pfc_Image object (allocate then copy the data object)

Parameters:

o another Pfc_Image object

17.15.3.3 vector< int > TMSP_Face_Space::Pfc_Image::GetCumHistogramme ()

Returns:

the Cumulative histogramm of the image

17.15.3.4 uint8_t TMSP_Face_Space::Pfc_Image::GetData (int *pos*)

Get the value of a pixel at a specific pointer position

Parameters:

pos he pointer position

Returns:

the value of a pixel

17.15.3.5 uint8_t * TMSP_Face_Space::Pfc_Image::GetDataptr ()

Returns:

the pointer to the image data

17.15.3.6 vector< int > TMSP_Face_Space::Pfc_Image::GetHistogram ()

Returns:

a vector of the pixels values distribution

17.15.3.7 int TMSP_Face_Space::Pfc_Image::GetImBytebp ()

Returns:

the bytesbypixel of the image

17.15.3.8 int TMSP_Face_Space::Pfc_Image::GetImHeight ()

Returns:

the image height

17.15.3.9 int TMSP_Face_Space::Pfc_Image::GetImSize ()

Returns:

the data size = Width∗Height∗Bytebp

17.15.3.10 int TMSP_Face_Space::Pfc_Image::GetImWidth ()

Returns:

the image width

17.15.3.11 void TMSP_Face_Space::Pfc_Image::Histeq ()

Histogramm Equalization

17.15.3.12 Pfc_Image ∗ TMSP_Face_Space::Pfc_Image::LightCorrect (LightEnhance *method*, double ∗ *param*, Mask *ROI*)

Correct illumination problem using a specified method and a Mask

Parameters:

method the LightEnhance method

param parameters of the method

ROI a Mask to specify the Region Of Interest

Returns:

a pointer to one or two Pfc_Image object (depends on the chosen LightEnhance method)

17.15.3.13 Pfc_Image ∗ TMSP_Face_Space::Pfc_Image::LightCorrect (LightEnhance *method*, double ∗ *param*)

Correct illumination problem using a specified method

Parameters:

method the LightEnhance method

param parameters of the method

Returns:

a pointer to one or two Pfc_Image objects (depends on the chosen LightEnhance method)

17.15.3.14 Pfc_Image ∗ TMSP_Face_Space::Pfc_Image::LightCorrect (string *methodname*, double ∗ *param*, Mask *ROI*)

Correct illumination problem using a specified method and a Mask

Parameters:

methodname the name of the illumination method values are ("NONE","HISTOGRAM","GAMMA","LOG" or "AS")

param parameters of the method

ROI a Mask to specify the Region Of Interest

Returns:

a pointer to one or two Pfc_Image object (depends on the chosen LightEnhance method)

17.15.3.15 Pfc_Image ∗ TMSP_Face_Space::Pfc_Image::LightCorrect (string *method*, double ∗ *param*)

Correct illumination problem using a specified method

Parameters:

method the illumination method values are ("NONE","HISTOGRAM","GAMMA","LOG" or "AS")

param parameters of the method

Returns:

a pointer to one or two Pfc_Image objects (depends on the chosen LightEnhance method)

17.15.3.16 ReturnMatrix TMSP_Face_Space::Pfc_Image::MatFromPFCImage ()

return a pointer to a Matrix from Im_8 Pfc_Image data

Returns:

Matrix

17.15.3.17 uint8_t TMSP_Face_Space::Pfc_Image::maximum ()

Returns:

the maximum value of the gray scale image

17.15.3.18 float TMSP_Face_Space::Pfc_Image::mean ()

Returns:

the mean of the pixels values

17.15.3.19 uint8_t TMSP_Face_Space::Pfc_Image::minimum ()

Returns:

the minimum value of the gray scale image

17.15.3.20 uint8_t TMSP_Face_Space::Pfc_Image::operator() (int *i*, int *j*)

return the value of a pixel at the image coordinates i and j

Parameters:

i the row coordinate

j the column coordinate

Returns:

the value of the pixel

17.15.3.21 Pfc_Image & TMSP_Face_Space::Pfc_Image::operator= (const Pfc_Image & *o*)

fill the image with the same data of another Pfc_Image (don't copy just point is quicker)

Parameters:

o another Pfc_Image object

Returns:

this

17.15.3.22 int TMSP_Face_Space::Pfc_Image::ReadImage (const char ∗*filename*, Extension *ext*)

Set the data of the object by reading an image from file specifying the extension of the file

Parameters:

filename the filename of the image

ext the extension or format values are JPG, PGM or PPM

Returns:

0 if well done

17.15.3.23 int TMSP_Face_Space::Pfc_Image::ReadImage (const char ∗*filename*)

Set the data of the object by reading an image from file

Parameters:

filename the filename of the image

Returns:

0 if well done

17.15.3.24 int TMSP_Face_Space::Pfc_Image::ReadJpeg (const char ∗ *filename*)

Set the data of the object by reading an image from JPEG file

Parameters:

filename the filename of the image

Returns:

0 if well done

17.15.3.25 int TMSP_Face_Space::Pfc_Image::ReadPgm (const char ∗ *filename*)

Set the data of the object by reading an image from PGM file

Parameters:

filename the filename of the image

Returns:

0 if well done

17.15.3.26 int TMSP_Face_Space::Pfc_Image::ReadPpm (const char ∗ *filename*)

Set the data of the object by reading an image from PPM file

Parameters:

filename the filename of the image

Returns:

0 if well done

17.15.3.27 void TMSP_Face_Space::Pfc_Image::Rotate90 ()

Rotate image by +90°

17.15.3.28 void TMSP_Face_Space::Pfc_Image::SaveImage (char ∗ *filename*)

save the image to file

Parameters:

filename the output filename

17.15.3.29 void TMSP_Face_Space::Pfc_Image::SetData (int *pos*, uint8_t *val*)

Set the value of the image pixel at a specific pointer position

Parameters:

pos the pointer position

val the value to be set

Returns:

17.15.3.30 void TMSP_Face_Space::Pfc_Image::SetImBytebp (int *Bb*)

Set the bytesbypixel of the image

Parameters:

Bb bytesbypixel

17.15.3.31 void TMSP_Face_Space::Pfc_Image::SetImData (uint8_t ∗ *Dataptr*, int *width*, int *height*, int *Bytebp*)

fill the image to the specified pointer (don't copy the data but just point to it)

Parameters:

Dataptr the pointer to the image data

width width of the image

height height of the image

Bytebp bytesbypixel of the image

17.15.3.32 void TMSP_Face_Space::Pfc_Image::SetImDim (int *w*, int *h*)

set the width and the height of the image

Parameters:

w width

h height

17.15.3.33 void TMSP_Face_Space::Pfc_Image::SetImHeight (int *h*)

set the height of the image

Parameters:

h height

17.15.3.34 void TMSP_Face_Space::Pfc_Image::SetImParam (int *w*, int *h*, int *Bb*)

set the width, the height and the bytesbypixel of the image

Parameters:

w width

h height

Bb bytesbypixel

17.15.3.35 void TMSP_Face_Space::Pfc_Image::SetImWidth (int *w*)

set the width of the image

Parameters:

w width

17.15.3.36 float TMSP_Face_Space::Pfc_Image::std ()

Returns:

the standard deviation of the pixels values

17.15.3.37 void TMSP_Face_Space::Pfc_Image::Stretch (int *method*, Mask & *ROI*)

stretching the histogram of the Masked image

Parameters:

method two values are available 0 and 1

ROI the mask to be applied

17.15.3.38 void TMSP_Face_Space::Pfc_Image::Stretch (int *method*)

stretch the histogram of the image

Parameters:

method two values are available

- 0 : do a min-max stretching
- 1: center the histogram to 128 and standard deviation of 50

17.15.3.39 float TMSP_Face_Space::Pfc_Image::sum_square ()

Returns:

the sum square of the pixels values

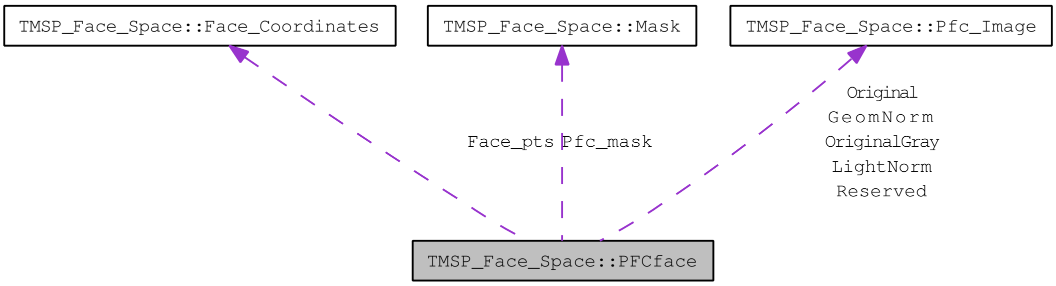
The documentation for this class was generated from the following files:

- lib/include/TMSP_image.h
- lib/TMSPFace/TMSP_image.cpp

17.16 TMSP_Face_Space::PFCface Class Reference

Class that stores all the data we need to do a verification.

`#include <pfcface.h>`Collaboration diagram for TMSP_Face_Space::PFCface:



Public Member Functions

- PFCface ()
- ~PFCface ()
- void SetNormLight (int Norm)
- int GetNormLight ()
- void SetEyeDistance (int Eydist)
- int GetEyeDistance ()
- int NormGeometric ()
- int LightCorrect (string method, double ∗param)
- int LightCorrect (string methodname, double ∗param, Mask ROI)
- void SetOriginal (Pfc_Image &face)
- void SetFaceCoordinate (Face_Coordinates Fc)
- void SetFaceCoordinate (int LEX, int LEY, int REX, int REY, int NX, int NY, int MX, int MY)
- PFCface & operator= (const PFCface &o)
- void Printmeta ()
- void LoadOriginal (const char ∗filename)
- void LoadOriginal (const char ∗filename, Extension ext)

Data Fields

- Pfc_Image Original
- Pfc_Image OriginalGray
- Pfc_Image GeomNorm
- Pfc_Image LightNorm
- Pfc_Image Reserved

17.16.1 Detailed Description

Class that stores all the data we need to do a verification.

17.16.2 Constructor & Destructor Documentation

17.16.2.1 TMSP_Face_Space::PFCface::PFCface ()

constructor

17.16.2.2 TMSP_Face_Space::PFCface::~PFCface ()

destructor

17.16.3 Member Function Documentation

17.16.3.1 int TMSP_Face_Space::PFCface::GetEyeDistance ()

Returns:

17.16.3.2 int TMSP_Face_Space::PFCface::GetNormLight ()

Returns:

17.16.3.3 int TMSP_Face_Space::PFCface::LightCorrect (string *methodname*, double ∗ *param*, Mask *ROI*)

Parameters:

methodname

param

ROI

Returns:

17.16.3.4 int TMSP_Face_Space::PFCface::LightCorrect (string *method*, double ∗ *param*)

Parameters:

method
param

Returns:

17.16.3.5 void TMSP_Face_Space::PFCface::LoadOriginal (const char ∗ *filename*, Extension *ext*)

Parameters:

filename
ext

17.16.3.6 void TMSP_Face_Space::PFCface::LoadOriginal (const char ∗ *filename*)

Parameters:

filename

17.16.3.7 int TMSP_Face_Space::PFCface::NormGeometric ()

Returns:

17.16.3.8 PFCface & TMSP_Face_Space::PFCface::operator= (const PFCface & *o*)

Parameters:

o

Returns:

17.16.3.9 void TMSP_Face_Space::PFCface::Printmeta ()

17.16.3.10 void TMSP_Face_Space::PFCface::SetEyeDistance (int *Eydist*)

Parameters:

Eydist

17.16.3.11 void TMSP_Face_Space::PFCface::SetFaceCoordinate (int *LEX*, int *LEY*, int *REX*, int *REY*, int *NX*, int *NY*, int *MX*, int *MY*)

Parameters:

LEX

LEY

REX

REY

NX

NY

MX

MY

17.16.3.12 void TMSP_Face_Space::PFCface::SetFaceCoordinate (Face_Coordinates *Fc*)

Parameters:

Fc

17.16.3.13 void TMSP_Face_Space::PFCface::SetNormLight (int *Norm*)

Parameters:

Norm

17.16.3.14 void TMSP_Face_Space::PFCface::SetOriginal (Pfc_Image & *face*)

Parameters:

face

17.16.4 Field Documentation

17.16.4.1 Pfc_Image TMSP_Face_Space::PFCface::GeomNorm

Geometrically Normalized face

17.16.4.2 Pfc_Image TMSP_Face_Space::PFCface::LightNorm

Illumination correction (Histogram Equalization,Gamma transformation,Log)

17.16.4.3 Pfc_Image TMSP_Face_Space::PFCface::Original

The Gobal image scene

17.16.4.4 Pfc_Image TMSP_Face_Space::PFCface::OriginalGray

17.16.4.5 Pfc_Image TMSP_Face_Space::PFCface::Reserved

Reserved for the illumination componnet extracted fro anisotropic smoothing LightEnhence

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

- lib/include/pfcface.h
- lib/TMSPFace/pfcface.cpp

17.17 TMSP_Face_Space::PfcPoint Struct Reference

structure for storing a point

```
#include <TMSP_image.h>
```

Data Fields

- int x
- int y

17.17.1 Detailed Description

structure for storing a point

17.17.2 Field Documentation

17.17.2.1 int TMSP_Face_Space::PfcPoint::x

The X Coordinate

17.17.2.2 int TMSP_Face_Space::PfcPoint::y

The Y Coordinate

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

- lib/include/TMSP_image.h

17.18 point Struct Reference

```
#include <TMSPFace.h>
```

Data Fields

- int x
- int y

17.18.1 Field Documentation

17.18.1.1 int point::x

17.18.1.2 int point::y

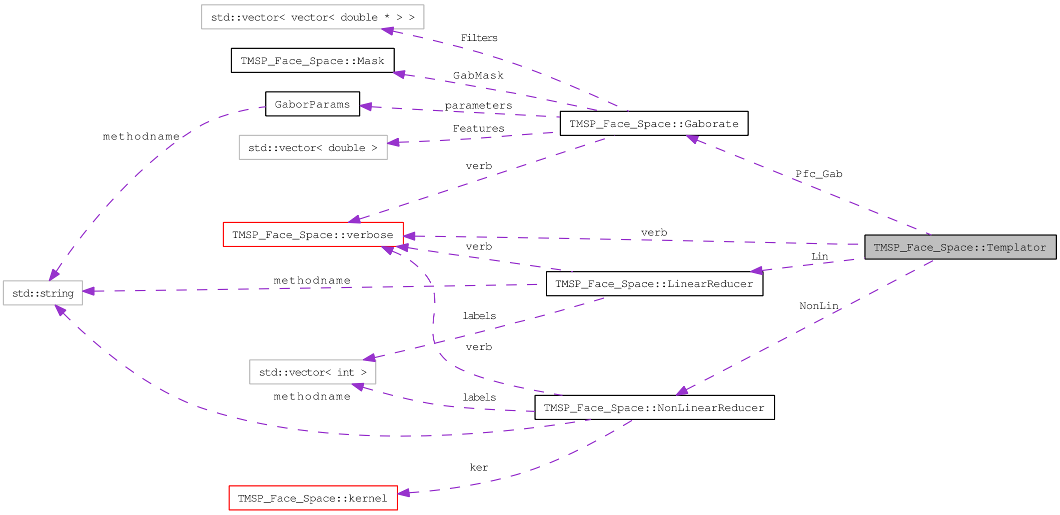
The documentation for this struct was generated from the following files:

- lib/TMSPFace.h
- bins/GetPerformance/GetPerformance.cpp

17.19 TMSP_Face_Space::Templator Class Reference

Class that extracts template from a given, geometrically normalized and corrected illumination, face.

`#include <templator.h>`Collaboration diagram for TMSP_Face_Space::Templator:



Public Member Functions

- Templator ()
- Templator (const char ∗SpaceRed)
- ∼Templator ()
- void SetSpacefile (const char ∗SpaceRed)
- vector< double > Template ()
- vector< double > Template (Matrix &F)
- vector< double > Template (Pfc_Image &F)
- void CreateFilters (int height, int width, int nscale, int norient, int minWaveLength, float mult, float sigmaOnf, float dThetaOnSigma)
- vector< double > LoadTemplate (const char ∗templ)
- void SetNoSpace ()
- bool GetProblem ()
- void SetMask (int eyedist)
- void SetGaborReduction (int val)
- void SaveTemplate (const char ∗templ)
- ReturnMatrix GetTemplate ()
- void SetGaborMethod (string meth)
- string GetGaborMethod ()
- void SetVerbose (verbose &v)
- void SetMutex (pthread_mutex_t ∗mx)

Data Fields

- LinearReducer Lin
- NonLinearReducer NonLin
- Gaborate Pfc_Gab

17.19.1 Detailed Description

Class that extracts template from a given, geometrically normalized and corrected illumination, face.

17.19.2 Constructor & Destructor Documentation

17.19.2.1 TMSP_Face_Space::Templator::Templator ()

Constructor

17.19.2.2 TMSP_Face_Space::Templator::Templator (const char ∗ *SpaceRed*)

Constructor

Parameters:

SpaceRed

17.19.2.3 TMSP_Face_Space::Templator::∼Templator ()

Destructor

17.19.3 Member Function Documentation

17.19.3.1 void TMSP_Face_Space::Templator::CreateFilters (int *height*, int *width*, int *nscale*, int *norient*, int *minWaveLength*, float *mult*, float *sigmaOnf*, float *dThetaOnSigma*)

Parameters:

height

width

nscale

norient

minWaveLength

mult

sigmaOnf

dThetaOnSigma

17.19.3.2 string TMSP_Face_Space::Templator::GetGaborMethod ()

Returns:

17.19.3.3 bool TMSP_Face_Space::Templator::GetProblem ()

Returns:

17.19.3.4 ReturnMatrix TMSP_Face_Space::Templator::GetTemplate ()

Returns:

17.19.3.5 vector< double > TMSP_Face_Space::Templator::LoadTemplate (const char * *templ*)

Parameters:

templ

Returns:

17.19.3.6 void TMSP_Face_Space::Templator::SaveTemplate (const char * *templ*)

Parameters:

templ

17.19.3.7 void TMSP_Face_Space::Templator::SetGaborMethod (string *meth*)

Parameters:

meth

17.19.3.8 void TMSP_Face_Space::Templator::SetGaborReduction (int *val*)

Parameters:

val

17.19.3.9 void TMSP_Face_Space::Templator::SetMask (int *eyedist*)

Parameters:

eyedist

17.19.3.10 void TMSP_Face_Space::Templator::SetMutex (pthread_mutex_t ∗ *mx*)

Parameters:

mx

17.19.3.11 void TMSP_Face_Space::Templator::SetNoSpace ()

17.19.3.12 void TMSP_Face_Space::Templator::SetSpacefile (const char ∗ *SpaceRed*)

Parameters:

SpaceRed

17.19.3.13 void TMSP_Face_Space::Templator::SetVerbose (verbose & *v*)

Parameters:

v

17.19.3.14 vector< double > TMSP_Face_Space::Templator::Template (Pfc_Image & *F*)

Parameters:

F

Returns:

17.19.3.15 vector<double> TMSP_Face_Space::Templator::Template (Matrix & *F*)

Parameters:

F

Returns:

17.19.3.16 vector<double> TMSP_Face_Space::Templator::Template ()

Returns:

17.19.4 Field Documentation

17.19.4.1 LinearReducer TMSP_Face_Space::Templator::Lin

17.19.4.2 NonLinearReducer TMSP_Face_Space::Templator::NonLin

LinearReducer object for the Linear reduction problem like (PCA, LDA and DLDA)

17.19.4.3 Gaborate TMSP_Face_Space::Templator::Pfc_Gab

NonLinearReducer object for the NonLinear reduction problem like (KFA and GDA) a Gaborate object to compute the gabor filtering step

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

- lib/include/templator.h
- lib/TMSPFace/templator.cpp

17.20 thread_data Struct Reference

```
#include <TMSPFace.h>
```

Data Fields

- int thread_id
- int startimage
- int endimage

17.20.1 Field Documentation

17.20.1.1 int thread_data::endimage

17.20.1.2 int thread_data::startimage

17.20.1.3 int thread_data::thread_id

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

- lib/TMSPFace.h

17.21 TMSP_Face_Space::Timer Class Reference

Class that allows to get time performance.

```
#include <timer.h>
```

Public Member Functions

- Timer ()
- ∼Timer ()
- void start ()
- float Get_Elapsed_restart ()
- float Get_Elapsed ()
- float Get_Elapsed_s ()
- float Get_Elapsed_s_restart ()

17.21.1 Detailed Description

Class that allows to get time performance.

17.21.2 Constructor & Destructor Documentation

17.21.2.1 TMSP_Face_Space::Timer::Timer ()

Constructor

17.21.2.2 TMSP_Face_Space::Timer::∼Timer ()

Destructor

17.21.3 Member Function Documentation

17.21.3.1 float TMSP_Face_Space::Timer::Get_Elapsed ()

return the elapsed time from Timer starting without restarting

Returns:

the elapsed time

17.21.3.2 float TMSP_Face_Space::Timer::Get_Elapsed_restart ()

turn the elapsed time in second from Timer starting and restart

Returns:

the elapsed time

17.21.3.3 float TMSP_Face_Space::Timer::Get_Elapsed_s ()

return the elapsed time in second from Timer starting without restarting

Returns:

the elapsed time in seconds

17.21.3.4 float TMSP_Face_Space::Timer::Get_Elapsed_s_restart ()

turn the elapsed time in second from Timer starting and restart

Returns:

the elapsed time in seconds

17.21.3.5 void TMSP_Face_Space::Timer::start ()

start the timer

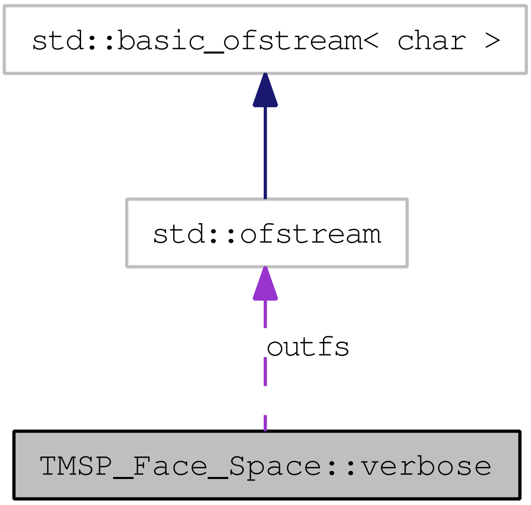
The documentation for this class was generated from the following files:

- lib/include/timer.h
- lib/TMSPFace/timer.cpp

17.22 TMSP_Face_Space::verbose Class Reference

Class that controls the verbozing of different classes.

`#include <verbose.h>`Collaboration diagram for TMSP_Face_Space::verbose:



Public Member Functions

- verbose ()
- verbose (bool f)
- verbose (int maxlevel)
- verbose (const char ∗f)
- verbose (bool f1, const char ∗f2)
- ∼verbose ()
- void Setlog (const char ∗filename)
- void Closelog ()
- void Verbose ()
- void UnVerbose ()
- verbose & operator<< (const char ∗)
- verbose & operator<< (const string &)
- verbose & operator<< (char)
- verbose & operator<< (long)
- verbose & operator<< (double)
- verbose & operator<< (float)
- verbose & operator<< (int)
- verbose & operator<< (bool)
- verbose & operator= (verbose &verb)
- void SetMutex (pthread_mutex_t ∗mx)
- int setmaxlevel (int l)

17.22.1 Detailed Description

Class that controls the verbozing of different classes.

Author:

Anouar mellakh <me.anouar@gmail.com>

17.22.2 Constructor & Destructor Documentation

17.22.2.1 TMSP_Face_Space::verbose::verbose ()

Contructor

17.22.2.2 TMSP_Face_Space::verbose::verbose (bool *f*)

Contructor

Parameters:

f

17.22.2.3 TMSP_Face_Space::verbose::verbose (int *maxlevel*)

Parameters:

maxlevel

17.22.2.4 TMSP_Face_Space::verbose::verbose (const char ∗ *f*)

Parameters:

f

17.22.2.5 TMSP_Face_Space::verbose::verbose (bool *f1*, const char ∗ *f2*)

Parameters:

f1

f2

17.22.2.6 TMSP_Face_Space::verbose::∼verbose ()

17.22.3 Member Function Documentation

17.22.3.1 void TMSP_Face_Space::verbose::Closelog ()

17.22.3.2 verbose & TMSP_Face_Space::verbose::operator<< (bool)

17.22.3.3 verbose & TMSP_Face_Space::verbose::operator<< (int *f*)

17.22.3.4 verbose & TMSP_Face_Space::verbose::operator<< (float *f*)

17.22.3.5 verbose & TMSP_Face_Space::verbose::operator<< (double *f*)

17.22.3.6 verbose & TMSP_Face_Space::verbose::operator<< (long *f*)

17.22.3.7 verbose& TMSP_Face_Space::verbose::operator<< (char)

17.22.3.8 verbose & TMSP_Face_Space::verbose::operator<< (const string & *f*)

17.22.3.9 verbose & TMSP_Face_Space::verbose::operator<< (const char ∗ *f*)

17.22.3.10 verbose & TMSP_Face_Space::verbose::operator= (verbose & *verb*)

Parameters:

verb

Returns:

17.22.3.11 void TMSP_Face_Space::verbose::Setlog (const char ∗ *filename*)

Parameters:

filename

17.22.3.12 int TMSP_Face_Space::verbose::setmaxlevel (int *l*)

Parameters:

l

Returns:

17.22.3.13 void TMSP_Face_Space::verbose::SetMutex (pthread_mutex_t ∗ *mx*)

Parameters:

mx

17.22.3.14 void TMSP_Face_Space::verbose::UnVerbose ()

17.22.3.15 void TMSP_Face_Space::verbose::Verbose ()

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

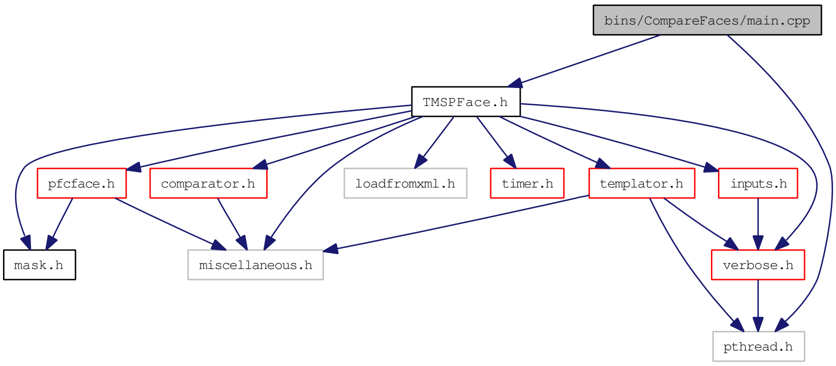
- lib/include/verbose.h
- lib/TMSPFace/verbose.cpp

18 File Documentation

18.1 bins/CompareFaces/main.cpp File Reference

```
#include <pthread.h>
#include "TMSPFace.h"
```

Include dependency graph for main.cpp:



Functions

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

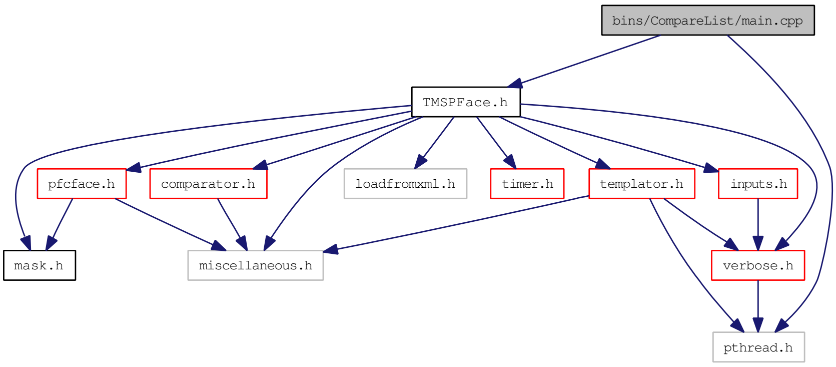
18.1.1 Function Documentation

18.1.1.1 int main (int *argc*, char ** *argv*)

18.2 bins/CompareList/main.cpp File Reference

```
#include <pthread.h>
#include "TMSPFace.h"
```

Include dependency graph for main.cpp:



Functions

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

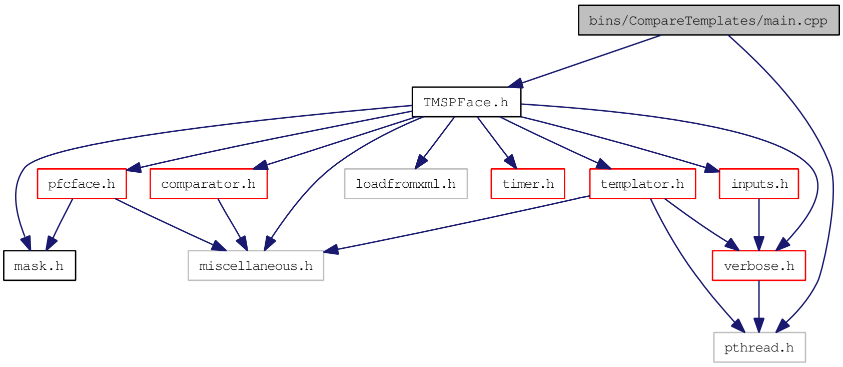
18.2.1 Function Documentation

18.2.1.1 int main (int *argc*, char ∗∗ *argv*)

18.3 bins/CompareTemplates/main.cpp File Reference

```
#include <pthread.h>
#include "TMSPFace.h"
```

Include dependency graph for main.cpp:



Functions

- int main (int argc, char ∗∗argv)

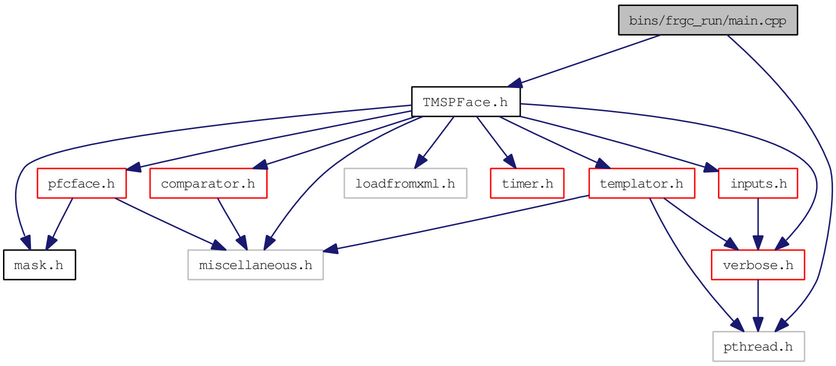
18.3.1 Function Documentation

18.3.1.1 int main (int *argc*, char ∗∗ *argv*)

18.4 bins/frgc_run/main.cpp File Reference

```
#include <pthread.h>
#include "TMSPFace.h"
```

Include dependency graph for main.cpp:



Functions

- void ∗ querytemplates (void ∗threadpass)
- void ∗ targettemplates (void ∗threadpass)
- void ∗ similarityintra (void ∗threadpass)
- void ∗ similarityinter (void ∗threadpass)
- void ∗ similarity (void ∗threadpass)
- int main (int argc, char ∗∗argv)

Variables

- Templator ∗ Pfc_temp
- struct thread_data ∗ thread_data_array
- pthread_mutex_t mutexout
- inputs Recinputs
- vector< string > ListofQuery
- vector< string > ListofTarget
- string ImageDir
- bool masking
- bool ∗ queryrun
- bool ∗ targetrun
- vector< double > ∗ DATAQuery
- vector< double > ∗ DATATarget
- int countquery = 0
- int counttarget = 0
- vector< point > intrat
- vector< point > intert
- int intra = 0
- int inter = 0
- float ∗ SimResult
- bool half = false
- verbose V (0)
- int steps = 150
- int icount = 1
- float lambda = 1.0
- int endimageQuery
- int endimageTarget

18.4.1 Function Documentation

18.4.1.1 int main (int *argc*, char ∗∗ *argv*)

18.4.1.2 void∗ querytemplates (void ∗ *threadpass*)

18.4.1.3 void∗ similarity (void ∗ *threadpass*)

18.4.1.4 void∗ similarityinter (void ∗ *threadpass*)

18.4.1.5 void∗ similarityintra (void ∗ *threadpass*)

18.4.1.6 void∗ targettemplates (void ∗ *threadpass*)

18.4.2 Variable Documentation

18.4.2.1 int countquery = 0

18.4.2.2 int counttarget = 0

18.4.2.3 vector< double >∗ DATAQuery

18.4.2.4 vector< double > ∗ DATATarget

18.4.2.5 int endimageQuery

18.4.2.6 int endimageTarget

18.4.2.7 bool half = false

18.4.2.8 int icount = 1

18.4.2.9 string ImageDir

18.4.2.10 int inter = 0

18.4.2.11 vector<point> intert

18.4.2.12 int intra = 0

18.4.2.13 vector<point> intrat

18.4.2.14 float lambda = 1.0

18.4.2.15 vector<string> ListofQuery

18.4.2.16 vector<string> ListofTarget

18.4.2.17 bool masking

18.4.2.18 pthread_mutex_t mutexout

18.4.2.19 Templator∗ Pfc_temp

18.4.2.20 bool ∗ queryrun

18.4.2.21 inputs Recinputs

18.4.2.22 float∗ SimResult

18.4.2.23 int steps = 150

18.4.2.24 bool ∗ targetrun

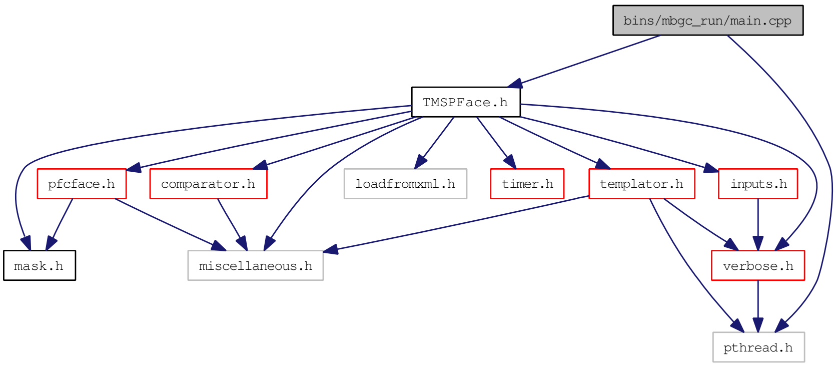
18.4.2.25 struct thread_data∗ thread_data_array

18.4.2.26 verbose V(0)

18.5 bins/mbgc_run/main.cpp File Reference

```
#include <pthread.h>
#include "TMSPFace.h"
```

Include dependency graph for main.cpp:



Functions

- int saveresselect (const char ∗filename, double ∗res, int size)
- bool findInVector (vector< int > &select, int val)
- int countnonnan (double ∗res, int size)
- int Discardmax (double ∗res, int width, int height, vector< int > &select, int keep)
- float getpseudodist (float ∗vect, int size, int method)
- void ∗ querytemplates (void ∗threadpass)
- void ∗ targettemplates (void ∗threadpass)
- void ∗ similarity (void ∗threadpass)
- void ∗ SelectStableQuery (void ∗threadpass)
- int main (int argc, char ∗∗argv)

Variables

- Templator ∗ Pfc_temp
- struct thread_data ∗ thread_data_array
- pthread_mutex_t mutexout
- inputs Recinputs
- string ImageDir
- vector< vector< int > > SelectedQuery
- vector< double > ∗ DATATarget
- vector< vector< double > > ∗ DATAQuery
- vector< XmlFace > ListofTargetStill
- vector< vector< XmlFace > > ListofQueryVideo
- int endimageTargetStill
- int endimageQueryVideo
- vector< vector< float ∗ > > SimResult
- verbose V (0)
- int icount = 0

18.5.1 Function Documentation

18.5.1.1 int countnonnan (double ∗ *res*, int *size*)

18.5.1.2 int Discardmax (double $*$ *res*, int *width*, int *height*, vector$<$ int $>$ & *select*, int *keep*)

18.5.1.3 bool findInVector (vector$<$ int $>$ & *select*, int *val*)

18.5.1.4 float getpseudodist (float $*$ *vect*, int *size*, int *method*)

18.5.1.5 int main (int *argc*, char $**$ *argv*)

18.5.1.6 void$*$ querytemplates (void $*$ *threadpass*)

18.5.1.7 int saveresselect (const char $*$ *filename*, double $*$ *res*, int *size*)

18.5.1.8 void$*$ SelectStableQuery (void $*$ *threadpass*)

18.5.1.9 void$*$ similarity (void $*$ *threadpass*)

18.5.1.10 void$*$ targettemplates (void $*$ *threadpass*)

18.5.2 Variable Documentation

18.5.2.1 vector$<$ vector $<$ double $>$ $>*$ DATAQuery

18.5.2.2 vector$<$ double $>*$ DATATarget

18.5.2.3 int endimageQueryVideo

18.5.2.4 int endimageTargetStill

18.5.2.5 int icount = 0

18.5.2.6 string ImageDir

18.5.2.7 vector<vector <XmlFace> > ListofQueryVideo

18.5.2.8 vector<XmlFace> ListofTargetStill

18.5.2.9 pthread_mutex_t mutexout

18.5.2.10 Templator∗ Pfc_temp

18.5.2.11 inputs Recinputs

18.5.2.12 vector< vector < int > > SelectedQuery

18.5.2.13 vector< vector < float ∗ > > SimResult

18.5.2.14 struct thread_data∗ thread_data_array

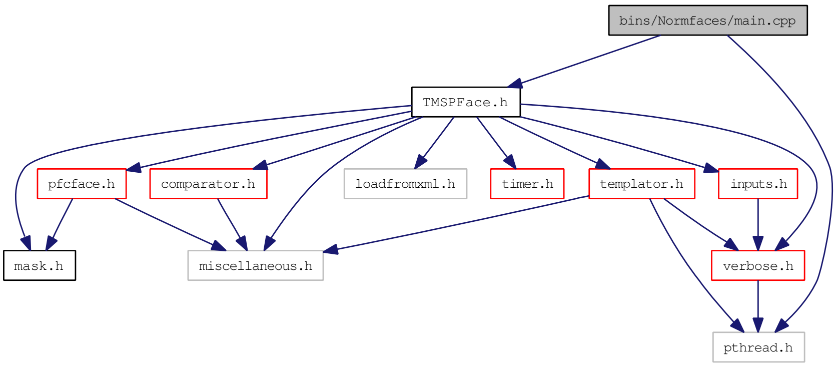
18.5.2.15 verbose V(0)

18.6 bins/Normfaces/main.cpp File Reference

`#include <pthread.h>`

`#include "TMSPFace.h"`

Include dependency graph for main.cpp:



Functions

- void ∗ normlist (void ∗threadpass)
- int main (int argc, char ∗∗argv)

Variables

- pthread_mutex_t mutexout
- vector< XmlFace > ListofImages
- int endimagelist
- int icount = 0
- string ImageDir
- string ImageOutdir
- inputs Recinputs
- verbose V (0)
- struct thread_data ∗ thread_data_array

18.6.1 Function Documentation

18.6.1.1 int main (int *argc*, char ∗∗ *argv*)

18.6.1.2 void∗ normlist (void ∗ *threadpass*)

18.6.2 Variable Documentation

18.6.2.1 int endimagelist

18.6.2.2 int icount = 0

18.6.2.3 string ImageDir

18.6.2.4 string ImageOutdir

18.6.2.5 vector< XmlFace > ListofImages

18.6.2.6 pthread_mutex_t mutexout

18.6.2.7 inputs Recinputs

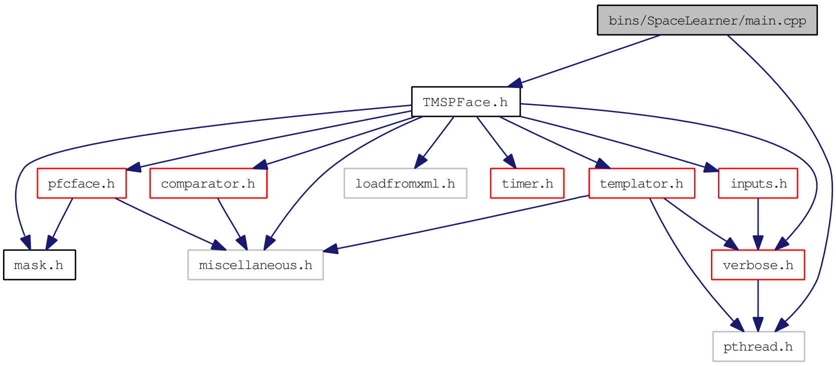
18.6.2.8 struct thread_data∗ thread_data_array

18.6.2.9 verbose V(0)

18.7 bins/SpaceLearner/main.cpp File Reference

```
#include <pthread.h>
#include "TMSPFace.h"
```

Include dependency graph for main.cpp:



Functions

- void ∗ boostcreation (void ∗threadpass)
- void ∗ boostloading (void ∗threadpass)
- int main (int argc, char ∗∗argv)

Variables

- struct thread_data ∗ thread_data_array
- pthread_mutex_t mutexout
- inputs Recinputs
- Matrix DATA
- string ImageDir
- int steps = 150
- int icount = 1
- int startimage
- int endimage
- float lambda = 1.0
- vector< XmlFace > ListofFile
- Gaborate ∗ Pfc_Gab
- verbose V (2)

18.7.1 Function Documentation

18.7.1.1 void∗ boostcreation (void ∗ *threadpass*)

18.7.1.2 void∗ boostloading (void ∗ *threadpass*)

18.7.1.3 int main (int *argc*, char ∗∗ *argv*)

18.7.2 Variable Documentation

18.7.2.1 Matrix DATA

18.7.2.2 int endimage

18.7.2.3 int icount = 1

18.7.2.4 string ImageDir

18.7.2.5 float lambda = 1.0

18.7.2.6 vector<XmlFace> ListofFile

18.7.2.7 pthread_mutex_t mutexout

18.7.2.8 Gaborate* Pfc_Gab

18.7.2.9 inputs Recinputs

18.7.2.10 int startimage

18.7.2.11 int steps = 150

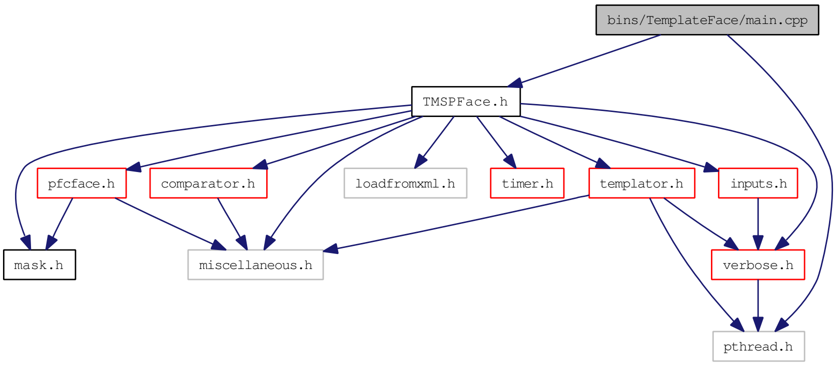
18.7.2.12 struct thread_data∗ thread_data_array

18.7.2.13 verbose V(2)

18.8 bins/TemplateFace/main.cpp File Reference

```
#include <pthread.h>
#include "TMSPFace.h"
```

Include dependency graph for main.cpp:



Functions

- int main (int argc, char ∗∗argv)

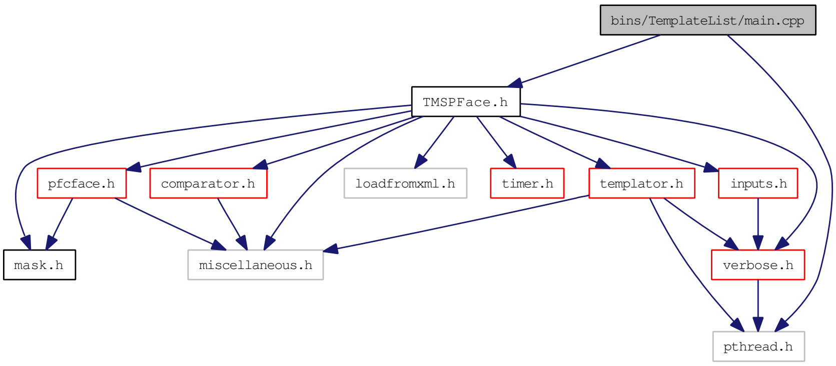
18.8.1 Function Documentation

18.8.1.1 int main (int *argc*, char ∗∗ *argv*)

18.9 bins/TemplateList/main.cpp File Reference

```
#include <pthread.h>
#include "TMSPFace.h"
```

Include dependency graph for main.cpp:



Functions

- void ∗ querytemplates (void ∗threadpass)
- int main (int argc, char ∗∗argv)

Variables

- Templator ∗ Pfc_temp
- struct thread_data ∗ thread_data_array
- pthread_mutex_t mutexout
- inputs Recinputs
- vector< string > ListofQuery
- vector< string > ListofCode
- string ImageDir
- string Outdir
- bool masking
- bool ∗ queryrun
- int countquery = 0
- vector< point > intrat
- vector< point > intert
- verbose V (0)
- int icount = 1
- int endimageQuery

18.9.1 Function Documentation

18.9.1.1 int main (int *argc*, char ∗∗ *argv*)

18.9.1.2 void∗ querytemplates (void ∗ *threadpass*)

18.9.2 Variable Documentation

18.9.2.1 int countquery = 0

18.9.2.2 int endimageQuery

18.9.2.3 int icount = 1

18.9.2.4 string ImageDir

18.9.2.5 vector<point> intert

18.9.2.6 vector<point> intrat

18.9.2.7 vector<string> ListofCode

18.9.2.8 vector<string> ListofQuery

18.9.2.9 bool masking

18.9.2.10 pthread_mutex_t mutexout

18.9.2.11 string Outdir

18.9.2.12 Templator∗ Pfc_temp

18.9.2.13 bool ∗ queryrun

18.9.2.14 inputs Recinputs

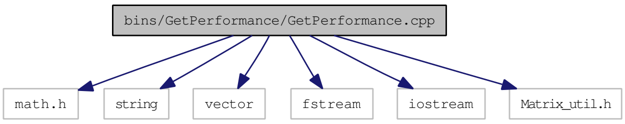
18.9.2.15 struct thread_data∗ thread_data_array

18.9.2.16 verbose V(0)

18.10 bins/GetPerformance/GetPerformance.cpp File Reference

```
#include <math.h>
#include <string>
#include <vector>
#include <fstream>
#include <iostream>
#include "Matrix_util.h"
```

Include dependency graph for GetPerformance.cpp:



Data Structures

- class inputarg
- struct point

Defines

- #define INTRA 255
- #define INTER 127

Functions

- void GetConfidence (double ∗Far, double ∗Frr, int histsize, int numimp, int numclient, double ∗&FarC, double ∗&FrrC, double confid)
- float min (float a, float b)
- float max (float a, float b)
- int Getproblemtype (vector< float > &inter, vector< float > &intra)
- int norm_dev (double ∗Fr, int histsize, double ∗&ndf)
- int IfFileExists (char ∗file)
- int main (int argc, char ∗argv[ ])
- vector< float > loadscores (char ∗filename)

18.10.1 Define Documentation

18.10.1.1 #define INTER 127

18.10.1.2 #define INTRA 255

18.10.2 Function Documentation

18.10.2.1 void GetConfidence (double ∗ *Far*, double ∗ *Frr*, int *histsize*, int *numimp*, int *numclient*, double ∗& *FarC*, double ∗& *FrrC*, double *confid*)

18.10.2.2 int Getproblemtype (vector< float > & *inter*, vector< float > & *intra*)

18.10.2.3 int IfFileExists (char ∗ *file*)

18.10.2.4 vector< float > loadscores (char ∗ *filename*)

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

18.10.2.6 float max (float *a*, float *b*)

18.10.2.7 float min (float *a*, float *b*)

18.10.2.8 int norm_dev (double ∗ *Fr*, int *histsize*, double ∗& *ndf*)

18.11 Examples/Learning.xml File Reference

18.12 Examples/metadata.xml File Reference

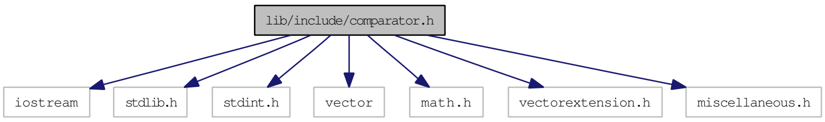
18.13 Examples/PFC_param.xml File Reference

18.14 Examples/TestList.lst File Reference

18.15 lib/include/comparator.h File Reference

```
#include <iostream>
#include <stdlib.h>
#include <stdint.h>
#include <vector>
#include <math.h>
#include <vectorextension.h>
#include "miscellaneous.h"
```

Include dependency graph for comparator.h:



Data Structures

- class TMSP_Face_Space::Comparator

  Class that compares two given templates based on the chosen Distance.

Namespaces

- namespace TMSP_Face_Space

Enumerations

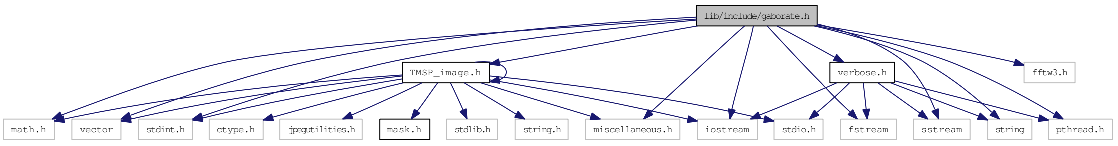
- enum TMSP_Face_Space::DISTANCE { TMSP_Face_Space::C_L1 = 1, TMSP_Face_Space::C_L2, TMSP_Face_Space::C_Angle, TMSP_Face_Space::C_NormDist }

Supported measure distances between templates.

18.16 lib/include/gaborate.h File Reference

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <string>
#include <math.h>
#include <vector>
#include <stdint.h>
#include <fftw3.h>
#include "miscellaneous.h"
#include "TMSP_image.h"
#include "verbose.h"
#include <pthread.h>
```

Include dependency graph for gaborate.h:



Data Structures

- struct GaborParams

  Structure that stores the gabor filters parameters and the returned complexe part (real, imaginary, magnitude, phase).

- class TMSP_Face_Space::Gaborate

  Class that computes the gabor filtering of an input image.

Namespaces

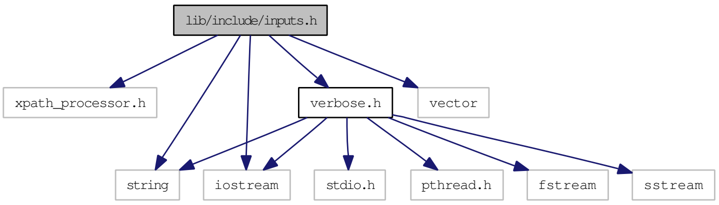
- namespace TMSP_Face_Space

18.17 lib/include/inputs.h File Reference

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

```
#include <vector>
#include <iostream>
#include "verbose.h"
```

Include dependency graph for inputs.h:



Data Structures

- class TMSP_Face_Space::inputs

  Class that reads framework parameters from the xml parameters file paramfile.xml .

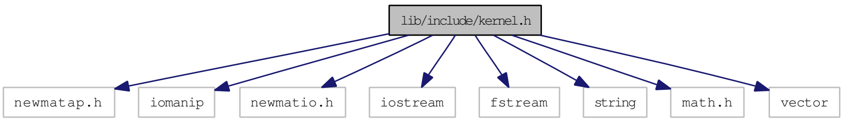
Namespaces

- namespace TMSP_Face_Space

18.18 lib/include/kernel.h File Reference

```
#include "newmatap.h"
#include <iomanip>
#include "newmatio.h"
#include <iostream>
#include <fstream>
#include <string>
#include <math.h>
#include <vector>
```

Include dependency graph for kernel.h:



Data Structures

- struct TMSP_Face_Space::kerlin

  *structure that stores a Linear kernel parameters Y = X*alpha + decal;*

- struct TMSP_Face_Space::kerpoly

  structure that stores a Polynomial kernel parameters Y = pow ((X + decal),power)

- struct TMSP_Face_Space::kerrbf

  structure that stores a Radial Basis Function kernel parameters Y= exp (-X∗power) ;

- struct TMSP_Face_Space::kersigmoid

  structure that stores a Sigmoid kernel parameters Y=tanh (X∗gamma + decal);

- class TMSP_Face_Space::kernel

  Class that create kernels and implement kernel methods.

Namespaces

- namespace TMSP_Face_Space

Defines

- #define ONE 1
- #define TWO 2
- #define LINEAR 0
- #define POLY 1
- #define RBF 2
- #define SIGMOID 3

18.18.1 Define Documentation

18.18.1.1 #define LINEAR 0

18.18.1.2 #define ONE 1

18.18.1.3 #define POLY 1

18.18.1.4 #define RBF 2

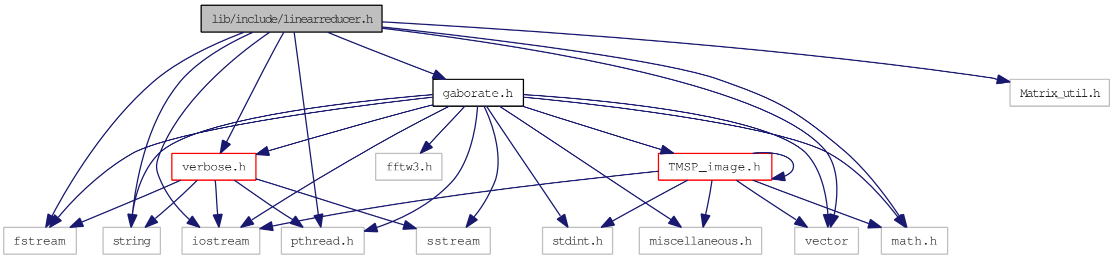
18.18.1.5 #define SIGMOID 3

18.18.1.6 #define TWO 2

18.19 lib/include/linearreducer.h File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <math.h>
#include <pthread.h>
#include "Matrix_util.h"
#include "gaborate.h"
#include "verbose.h"
```

Include dependency graph for linearreducer.h:



Data Structures

- class TMSP_Face_Space::LinearReducer

  Class used to reduce the input space by linear methods (PCA,LDA,DLDA).

Namespaces

- namespace TMSP_Face_Space

Enumerations

- enum TMSP_Face_Space::LProblem { TMSP_Face_Space::PCA = 0, TMSP_Face_Space::LDA, TMSP_Face_Space::DLDA }

  the supported linear problems

18.20 lib/include/mask.h File Reference

Data Structures

- class TMSP_Face_Space::Mask

  Class that creates an elliptic mask to be applied to faces.

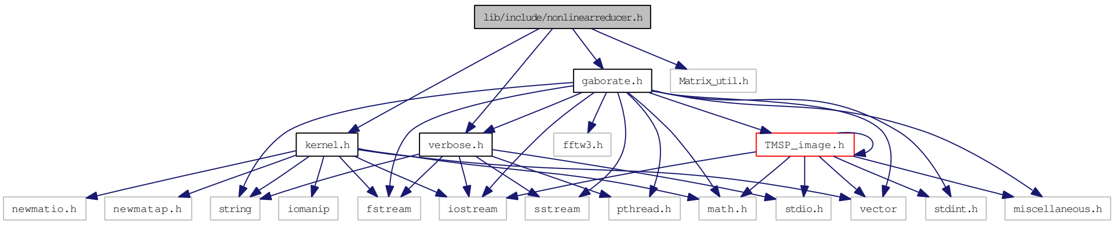
Namespaces

- namespace TMSP_Face_Space

18.21 lib/include/nonlinearreducer.h File Reference

```
#include "kernel.h"
#include "Matrix_util.h"
#include "verbose.h"
#include "gaborate.h"
```

Include dependency graph for nonlinearreducer.h:



Data Structures

- class TMSP_Face_Space::NonLinearReducer

  Class used to reduce the input space by nonlinear methods using kernel approaches (KFA,GDA).

Namespaces

- namespace TMSP_Face_Space

Defines

- #define reg 0.001

Enumerations

- enum TMSP_Face_Space::NProblem { TMSP_Face_Space::KFA = 0, TMSP_Face_Space::GDA }

  the supported Nonlinear problems

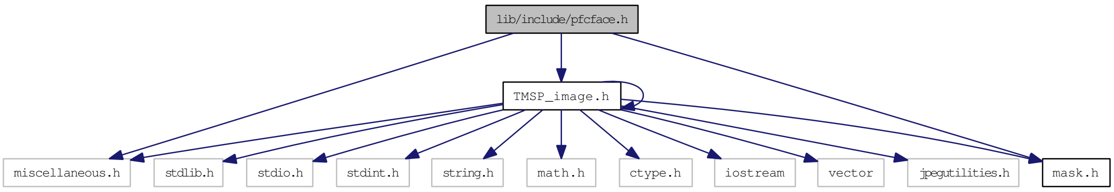
18.21.1 Define Documentation

18.21.1.1 #define reg 0.001

18.22 lib/include/pfcface.h File Reference

```
#include "miscellaneous.h"
#include "TMSP_image.h"
#include "mask.h"
```

Include dependency graph for pfcface.h:



Data Structures

- class TMSP_Face_Space::PFCface

  Class that stores all the data we need to do a verification.

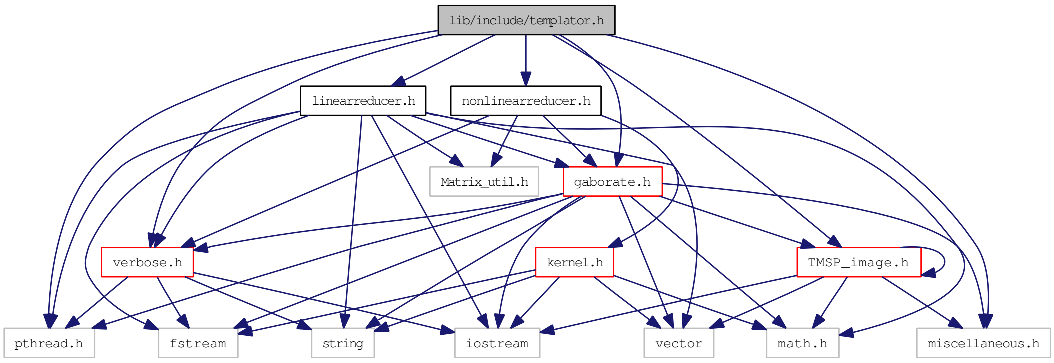
Namespaces

- namespace TMSP_Face_Space

18.23 lib/include/templator.h File Reference

```
#include <pthread.h>
#include "linearreducer.h"
#include "nonlinearreducer.h"
#include "gaborate.h"
#include "TMSP_image.h"
#include "miscellaneous.h"
#include "verbose.h"
```

Include dependency graph for templator.h:



Data Structures

- class TMSP_Face_Space::Templator

  Class that extracts template from a given, geometrically normalized and corrected illumination, face.

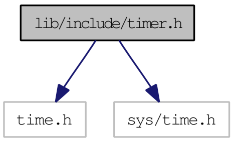
Namespaces

- namespace TMSP_Face_Space

18.24 lib/include/timer.h File Reference

```
#include <time.h>
#include <sys/time.h>
```

Include dependency graph for timer.h:



Data Structures

- class TMSP_Face_Space::Timer

  Class that allows to get time performance.

Namespaces

- namespace TMSP_Face_Space

Defines

- #define TIKCS 1000000.0

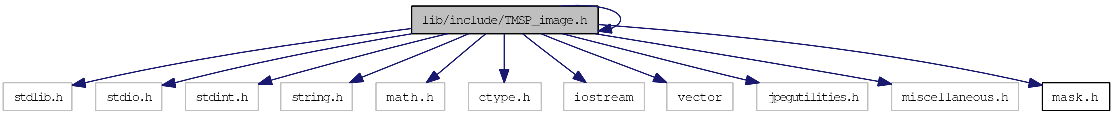
18.24.1 Define Documentation

18.24.1.1 #define TIKCS 1000000.0

18.25 lib/include/TMSP_image.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <stdint.h>
#include <string.h>
#include <math.h>
#include <ctype.h>
#include <iostream>
#include <vector>
#include "jpegutilities.h"
#include "miscellaneous.h"
#include "TMSP_image.h"
#include "mask.h"
```

Include dependency graph for TMSP_image.h:



Data Structures

- struct TMSP_Face_Space::Face_Coordinates

  Structure for storing the coordinates of Eyes, Nose and Mouth location in the original image.

- struct TMSP_Face_Space::PfcPoint

  structure for storing a point

- class TMSP_Face_Space::Pfc_Image

  Class that allows the manipulation of image for the baseline.

Namespaces

- namespace TMSP_Face_Space

Defines

- #define CENTER 0
- #define LEFT 1
- #define RIGHT 2
- #define UP 3
- #define DOWN 4
- #define Between(var, value, delta) ((var >= (value - delta)) && (var <= (value + delta)))

Enumerations

- enum TMSP_Face_Space::LightEnhance {

  TMSP_Face_Space::Im_NoEnhance = 0, TMSP_Face_Space::Im_Histogram, TMSP_Face_Space::Im_Gamma, TMSP_Face_Space::Im_Log,

  TMSP_Face_Space::Im_AnisSmooth, TMSP_Face_Space::Im_MultiRetinex, TMSP_Face_Space::Im_Pers }

  Supported Light correction.

- enum TMSP_Face_Space::ImFormat { TMSP_Face_Space::Im_8 = 1, TMSP_Face_Space::Im_16, TMSP_Face_Space::Im_24, TMSP_Face_Space::Im_32 }

  Image pixels format.

Functions

- Pfc_Image TMSP_Face_Space::PFCImageFromMat (Matrix &A)

  return a pointer to a Pfc_Image from Matrix data

- int TMSP_Face_Space::ASNorm (Pfc_Image &InputImage, int steps, float lambda, Pfc_Image &ReflectImage, Pfc_Image &LightImage)

  apply the Anisotropic smoothing

- int TMSP_Face_Space::ASNorm (Pfc_Image &InputImage, int steps, float lambda, Pfc_Image &ReflectImage, Pfc_Image &LightImage, Mask ROI, int meanref, float std)

  apply the Anisotropic smoothing with histogramm correction using a ROI and mean and std

18.25.1 Define Documentation

18.25.1.1 #define Between(var, value, delta) ((var >= (value - delta)) && (var <= (value + delta)))

18.25.1.2 #define CENTER 0

18.25.1.3 #define DOWN 4

18.25.1.4 #define LEFT 1

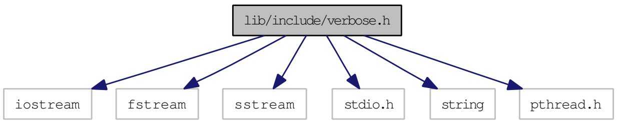
18.25.1.5 #define RIGHT 2

18.25.1.6 #define UP 3

18.26 lib/include/verbose.h File Reference

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <stdio.h>
#include <string>
#include <pthread.h>
```

Include dependency graph for verbose.h:



Data Structures

- class TMSP_Face_Space::verbose

  Class that controls the verbozing of different classes.

Namespaces

- namespace TMSP_Face_Space

Defines

- #define CLEARLINE true

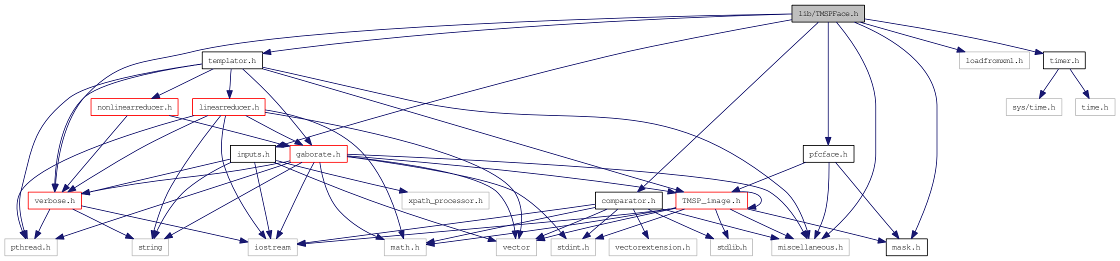
18.26.1 Define Documentation

18.26.1.1 #define CLEARLINE true

18.27 lib/TMSPFace.h File Reference

```
#include "templator.h"
#include "comparator.h"
#include "loadfromxml.h"
#include "miscellaneous.h"
#include "mask.h"
#include "inputs.h"
#include "verbose.h"
#include "timer.h"
#include "pfcface.h"
```

Include dependency graph for TMSPFace.h:



Data Structures

- struct thread_data
- struct point

Defines

- #define NTHREAD 16
- #define INTRA 255
- #define INTER 127

18.27.1 Define Documentation

18.27.1.1 #define INTER 127

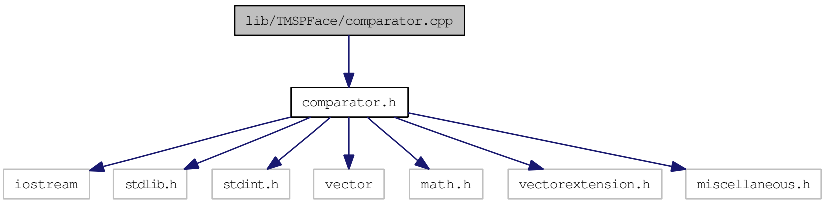
18.27.1.2 #define INTRA 255

18.27.1.3 #define NTHREAD 16

18.28 lib/TMSPFace/comparator.cpp File Reference

```
#include "comparator.h"
```

Include dependency graph for comparator.cpp:



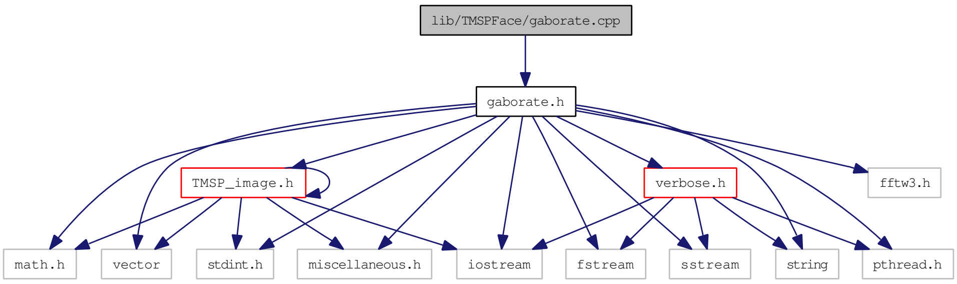
Namespaces

- namespace TMSP_Face_Space

18.29 lib/TMSPFace/gaborate.cpp File Reference

```
#include "gaborate.h"
```

Include dependency graph for gaborate.cpp:



Namespaces

- namespace TMSP_Face_Space

Defines

- #define pi 3.14159265

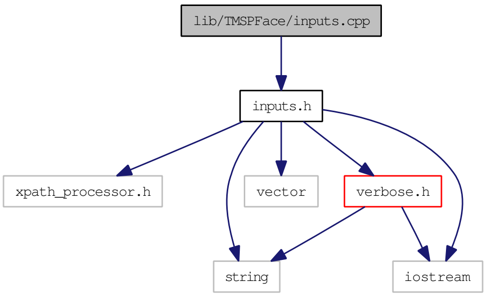
18.29.1 Define Documentation

18.29.1.1 #define pi 3.14159265

18.30 lib/TMSPFace/inputs.cpp File Reference

```
#include "inputs.h"
```

Include dependency graph for inputs.cpp:



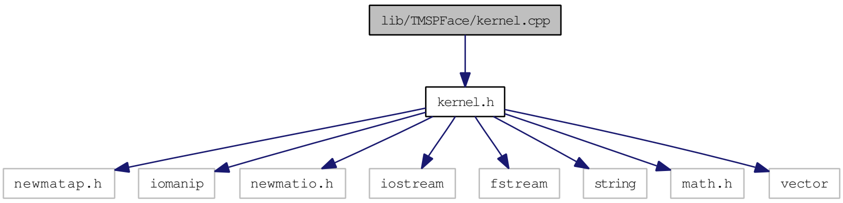
Namespaces

- namespace TMSP_Face_Space

18.31 lib/TMSPFace/kernel.cpp File Reference

```
#include "kernel.h"
```

Include dependency graph for kernel.cpp:



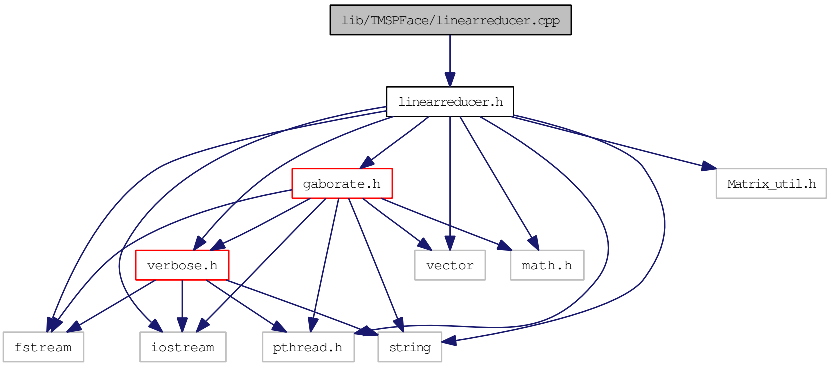
Namespaces

- namespace TMSP_Face_Space

18.32 lib/TMSPFace/linearreducer.cpp File Reference

```
#include "linearreducer.h"
```

Include dependency graph for linearreducer.cpp:



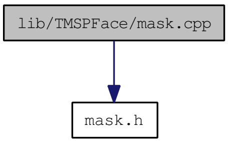
Namespaces

- namespace TMSP_Face_Space

18.33 lib/TMSPFace/mask.cpp File Reference

```
#include "mask.h"
```

Include dependency graph for mask.cpp:



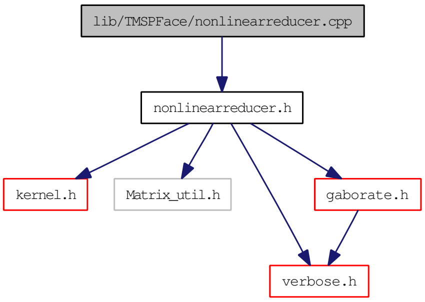
Namespaces

- namespace TMSP_Face_Space

18.34 lib/TMSPFace/nonlinearreducer.cpp File Reference

```
#include "nonlinearreducer.h"
```

Include dependency graph for nonlinearreducer.cpp:



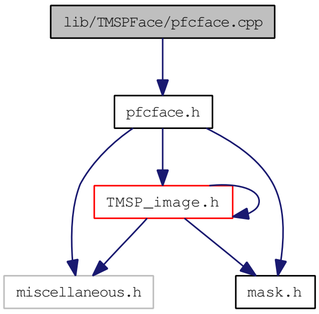
Namespaces

- namespace TMSP_Face_Space

18.35 lib/TMSPFace/pfcface.cpp File Reference

```
#include "pfcface.h"
```

Include dependency graph for pfcface.cpp:



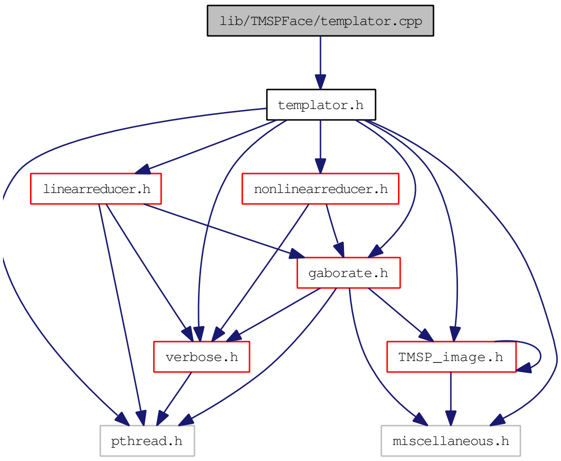
Namespaces

- namespace TMSP_Face_Space

18.36 lib/TMSPFace/templator.cpp File Reference

```
#include "templator.h"
```

Include dependency graph for templator.cpp:



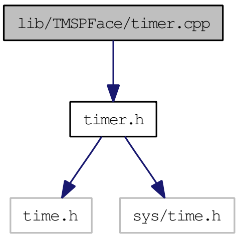
Namespaces

- namespace TMSP_Face_Space

18.37 lib/TMSPFace/timer.cpp File Reference

```
#include "timer.h"
```

Include dependency graph for timer.cpp:



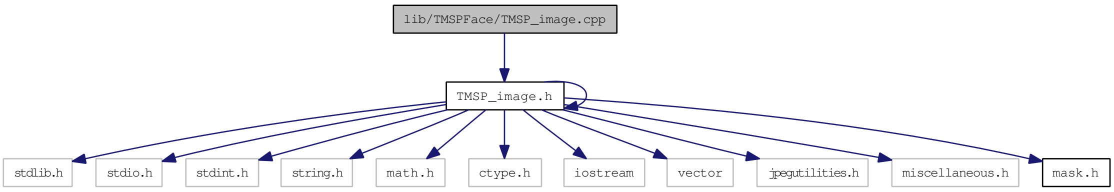
Namespaces

- namespace TMSP_Face_Space

18.38 lib/TMSPFace/TMSP_image.cpp File Reference

```
#include "TMSP_image.h"
```

Include dependency graph for TMSP_image.cpp:



Namespaces

- namespace TMSP_Face_Space

Defines

- #define swap(a, b)
- #define MEANREF 128
- #define STDREF 80.0

Functions

- Pfc_Image TMSP_Face_Space::PFCImageFromMat (Matrix &A)

  return a pointer to a Pfc_Image from Matrix data

18.38.1 Define Documentation

18.38.1.1 #define MEANREF 128

18.38.1.2 #define STDREF 80.0

18.38.1.3 #define swap(a, b)

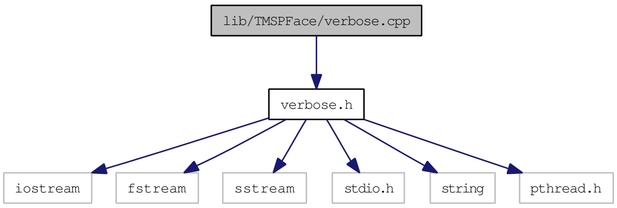
Value:

```
{ \
        int c = (a);    \
        (a) = (b);      \
        (b) = c;        \
    }
```

18.39 lib/TMSPFace/verbose.cpp File Reference

```
#include "verbose.h"
```

Include dependency graph for verbose.cpp:



Namespaces

- namespace TMSP_Face_Space

Index