ImageProcessingCemisid

1.0

Generated by Doxygen 1.7.4

Sat Jun 10 2017 15:59:16

Contents

1 Test List **1**

2 Class Index **5**

2.1 Class List 5

3 File Index **7**

3.1 File List 7

4 Class Documentation **9**

4.1 Image::h Class Reference 9

4.2 Image Class Reference 9

4.2.1 Constructor & Destructor Documentation 11

4.2.1.1 Image 11

4.2.1.2 Image 11

4.2.1.3 ∼Image 11

4.2.2 Member Function Documentation 11

4.2.2.1 Detection_Face 11

4.2.2.2 Diccionary 11

4.2.2.3 Dimension 11

4.2.2.4 Fast_awmr 12

4.2.2.5 Filtering 12

4.2.2.6 Four_windows_opencv 13

4.2.2.7 Gaussian_noise 13

4.2.2.8 Get_pixel 14

4.2.2.9 Get_pixels 14

4.2.2.10 Idtc_Robusta 14

4.2.2.11 Image_load . . . 15
4.2.2.12 Impulsive_uniform_noise . . . 16
4.2.2.13 Lost_pixels_noise . . . 16
4.2.2.14 MAE . . . 16
4.2.2.15 MSE . . . 17
4.2.2.16 Noise_remover . . . 17
4.2.2.17 Overlap . . . 18
4.2.2.18 PSNR . . . 18
4.2.2.19 Salt_and_pepper_noise . . . 18
4.2.2.20 SaveImage . . . 19
4.2.2.21 Two_windows_opencv . . . 19
4.2.2.22 wmedianf . . . 20
4.2.3 Member Data Documentation . . . 20
4.2.3.1 cols . . . 20
4.2.3.2 matrix . . . 20
4.2.3.3 rows . . . 20

5 File Documentation **21**

5.1 examples/basic_funcions.cpp File Reference . . . 21
5.1.1 Detailed Description . . . 21
5.1.2 Function Documentation . . . 21
5.1.2.1 main . . . 21
5.2 examples/detection_face.cpp File Reference . . . 21
5.2.1 Function Documentation . . . 22
5.2.1.1 main . . . 22
5.3 examples/filtering_mean.cpp File Reference . . . 22
5.3.1 Function Documentation . . . 22
5.3.1.1 main . . . 22
5.4 examples/filtering_median.cpp File Reference . . . 22
5.4.1 Function Documentation . . . 22
5.4.1.1 main . . . 22
5.5 examples/gaussian_noise.cpp File Reference . . . 22
5.5.1 Function Documentation . . . 23
5.5.1.1 main . . . 23

5.6 examples/idtc_Robusta_mean.cpp File Reference 23
5.6.1 Function Documentation 23
5.6.1.1 main 23
5.7 examples/idtc_Robusta_median.cpp File Reference 23
5.7.1 Function Documentation 23
5.7.1.1 main 23
5.8 examples/lost_Pixels_noise.cpp File Reference 23
5.8.1 Function Documentation 24
5.8.1.1 main 24
5.9 examples/noise_remover.cpp File Reference 24
5.9.1 Function Documentation 24
5.9.1.1 main 24
5.10 examples/overlap.cpp File Reference 24
5.10.1 Function Documentation 24
5.10.1.1 main 24
5.11 examples/salt_pepper.cpp File Reference 25
5.11.1 Function Documentation 25
5.11.1.1 main 25
5.12 examples/uniform_impulsive_noise.cpp File Reference 25
5.12.1 Function Documentation 25
5.12.1.1 main 25
5.13 include/Image.hpp File Reference 25
5.14 src/Detection_Face_opencv.cpp File Reference 26
5.14.1 Variable Documentation 26
5.14.1.1 face_cascade 26
5.15 src/Four_windows_opencv.cpp File Reference 26
5.16 src/Image_load.cpp File Reference 26
5.16.1 Function Documentation 27
5.16.1.1 Average_Median 27
5.17 src/main.cpp File Reference 27
5.17.1 Function Documentation 27
5.17.1.1 main 27
5.18 src/Two_windows_opencv.cpp File Reference 27

Chapter 1

Test List

Member Image::Dimension() basic_funcions.cpp

Image I;

I.Dimension();

Member Image::Filtering(arma::mat, int, int) filtering_median.cpp filtering_mean.cpp

mat matrix;

int l = 5;

int flag = 0;

Filtering(matrix, l, flag);

Member Image::Four_windows_opencv(const char ∗, const char ∗, const char ∗, const char ∗, const ch
main.cpp

const char∗ imag1= "src/Resources/images_salt_and_pepper/lady_256_0_1.pgm";

const char∗ imag2= "src/Resources/images_salt_and_pepper/lady_256_1.pgm";

const char∗ imag3= "src/Resources/images_salt_and_pepper/lady_256_10.pgm";

const char∗ imag4= "src/Resources/images_salt_and_pepper/lady_256_100.pgm";

Four_windows_opencv(imag1, imag2, imag3, imag4, label1, label2, label3, label4);

Member Image::Gaussian_noise(double) gaussian_noise.cpp

int nivel=0.1;

Gaussian_noise(nivel1);

Member Image::Get_pixel(int, int) basic_funcions.cpp

```
Image I;
int row=5;
int col=5;
I.Get_pixel(row, col);
```

Member Image::Get_pixels() basic_funcions.cpp

```
Image I;
I.Get_pixels();
```

Member Image::Idtc_Robusta(arma::mat, int) idtc_Robusta_median.cpp idtc_Robusta_-mean.cpp

```
mat matrix;
int flag = 0;
Idtc_Robusta(matrix, flag);
```

Member Image::Image() basic_funcions.cpp

```
Image I;
I.Dimension();
```

Member Image::Image_load(arma::mat, std::string) basic_funcions.cpp

```
mat matrix;
string ruta = "src/Resources/images/house.256.pgm";
Image_load(matrix, ruta);
```

Member Image::Impulsive_uniform_noise(double) uniform_impulsive_noise.cpp

```
double nivel=0.1;
Impulsive_uniform_noise(nivel);
```

Member Image::Lost_pixels_noise(double) lost_Pixels_noise.cpp

```
double nivel=0.1;
Lost_pixels_noise(nivel);
```

Member Image::MAE(arma::mat, arma::mat) filtering_mean.cpp

```
mat matrix_r;
mat matriz_f;
MAE(matrix_r, matriz_f);
```

Member Image::MSE(arma::mat, arma::mat) filtering_mean.cpp

```
mat matrix_r;
mat matriz_f;
MSE(matrix_r, matriz_f);
```

Member Image::Noise_remover(arma::mat, int) noise_remover.cpp

```
mat matrix;
int image_tam = 256;
Noise_remover(matrix, image_tam);
```

Member Image::Overlap(arma::mat) overlap.cpp

```
mat matrix;
Overlap(matrix);
```

Member Image::PSNR(arma::mat, arma::mat) filtering_mean.cpp

```
mat matrix_r;
mat matriz_f;
PSNR(matrix_r, matriz_f);
```

Member Image::Salt_and_pepper_noise(double) salt_pepper.cpp

```
double nivel=0.1;
Salt_and_pepper_noise(nivel);
```

Member Image::SaveImage(arma::mat, std::string) gaussian_noise.cpp

```
mat matrix;
string ruta= "src/Resources/images/house.256.pgm";
SaveImage(matrix, ruta);
```

Member Image::Two_windows_opencv(const char *, const char *, std::string, std::string) main.cpp

```
const char* imag1= "src/Resources/images_salt_and_pepper/lady_256_0_1.pgm";
const char* imag2= "src/Resources/images_salt_and_pepper/lady_256_1.pgm";
string label1 = "Imagen1";
string label2 = "Imagen2";
Two_windows_opencv(imag1, imag2, label1, label2);
```

Chapter 2

Class Index

2.1 Class List

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

Image::h 9
Image 9

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

examples/basic_funcions.cpp 21
examples/detection_face.cpp 21
examples/filtering_mean.cpp 22
examples/filtering_median.cpp 22
examples/gaussian_noise.cpp 22
examples/idtc_Robusta_mean.cpp 23
examples/idtc_Robusta_median.cpp 23
examples/lost_Pixels_noise.cpp 23
examples/noise_remover.cpp 24
examples/overlap.cpp 24
examples/salt_pepper.cpp 25
examples/uniform_impulsive_noise.cpp 25
include/Image.hpp 25
src/Detection_Face_opencv.cpp 26
src/Four_windows_opencv.cpp 26
src/Image_load.cpp 26
src/main.cpp 27
src/Two_windows_opencv.cpp 27

Chapter 4

Class Documentation

4.1 Image::h Class Reference

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

- include/Image.hpp

4.2 Image Class Reference

```
#include <Image.hpp>
```

Classes

- class h

Public Member Functions

- arma::mat Image_load (arma::mat, std::string)

  Carga de Imagen pgm a mat.
- Image ()

  Constructor de la clase.
- Image (arma::mat)
- void Dimension ()

  Obtiene las dimensiones de la matriz.
- int Get_pixel (int, int)

  Obtiene el valor de un pixel.
- void Get_pixels ()

  Imprime todos los pixeles de la matriz imagen.

- arma::mat Gaussian_noise (double)

  Aplica Ruido Gaussiano a una matriz.
- arma::mat Lost_pixels_noise (double)

  Aplica Perdida de pixeles a una matriz.
- arma::mat Salt_and_pepper_noise (double)

  Aplica Ruido Sal y Pimienta a una matriz.
- arma::mat Impulsive_uniform_noise (double)

  Aplica Ruido Impulsivo Uniforme a una matriz.
- void SaveImage (arma::mat, std::string)

  Guarda la matriz en formato pgm.
- double PSNR (arma::mat, arma::mat)

  Calculo del PSNR.
- double MAE (arma::mat, arma::mat)

  Calculo del MAE.
- double MSE (arma::mat, arma::mat)

  Calculo del MSE.
- arma::mat Filtering (arma::mat, int, int)

  Aplica un filtrado usando el promedio o la mediana para remover ruido.
- void Detection_Face (std::string)
- void Four_windows_opencv (const char *, const char *, const char *, const char *, const char *, const char *, const char *, const char *)

  Carga cuatro imagenes en una ventana de opencv.
- void Two_windows_opencv (const char *, const char *, std::string, std::string)

  Carga dos imagenes en una ventana de opencv.
- arma::mat Noise_remover (arma::mat, int)

  Aproximación Robusta por bloques No solpados (ventana 8x8)
- arma::mat Overlap (arma::mat)

  Aproximación por bloques solpados (ventana 8x8)
- arma::mat Idtc_Robusta (arma::mat, int)

  Aproximación Robusta por bloques solpados (ventana 8x8)
- double wmedianf (arma::vec &, arma::vec &)

  Calculo de la mediana.
- arma::mat Diccionary ()

  Diccionario de la transformada del Coseno.
- arma::vec Fast_awmr (arma::vec &y, arma::mat &A, int sparsity, int itmax, double beta, double tol, double epsilon, int numcoefperiter, double Kpar)

  Funcion necesaria para idt_Robusta()
- ~Image ()

Private Attributes

- int cols
- int rows
- arma::mat matrix

4.2.1 Constructor & Destructor Documentation

4.2.1.1 Image::Image ( )

Constructor de la clase.

Descripcion: Constructor de la clase Image para acceder a sus metodos internos

Test

basic_funcions.cpp
Image I;
I.Dimension();

4.2.1.2 Image::Image (arma::mat)

4.2.1.3 Image::∼Image ( )

4.2.2 Member Function Documentation

4.2.2.1 void Image::Detection_Face (std::string)

4.2.2.2 mat Image::Diccionary ( )

Diccionario de la transformada del Coseno.

Descripcion: Funcion necesaria para idt_Robusta()

Basado en los Articulos:

Ramírez J. y Paredes J. (2014). Robust Sparse Recovery Base On Weighted Median Operator. IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP). Department of Electrical Engineering, Universidad de Los Andes, Mérida, Venezuela. Doi: 978-1-4799-2893-4/14/$31.00. Recuperado de: http://www.mirlab.org/conference_papers/International_Conference/ICASSP%202014/papers/p1050-ramirez.pdf

Ramírez J. y Paredes J. (2015). Robust Transforms Based on the Weighted Median Operator. IEEE Signal Processing Letters, 22(1), pp. 120 – 124. doi: 10.1109/LSP.2014.2349351. Recuperado de: http://ieeexplore.ieee.org/document/6880779/

4.2.2.3 void Image::Dimension ( )

Obtiene las dimensiones de la matriz.

Descripcion: Permite obtener la dimension nxn de la matriz.

Returns

Retorna la dimension de la imagen nxn.

Test

basic_funcions.cpp
Image I;
I.Dimension();

4.2.2.4 vec Image::Fast_awmr (arma::vec & *y,* arma::mat & *A,* int *sparsity,* int *itmax,* double *beta,* double *tol,* double *epsilon,* int *numcoefperiter,* double *Kpar*)

Funcion necesaria para idt_Robusta()

Basado en los Articulos:

Ramírez J. y Paredes J. (2014). Robust Sparse Recovery Base On Weighted Median Operator. IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP). Department of Electrical Engineering, Universidad de Los Andes, Mérida, Venezuela. Doi: 978-1-4799-2893-4/14/$31.00. Recuperado de: http://www.mirlab.org/conference papers/International_Conference/ICASSP%202014/papers/p1050-ramirez.pdf

Ramírez J. y Paredes J. (2015). Robust Transforms Based on the Weighted Median Operator. IEEE Signal Processing Letters, 22(1), pp. 120 – 124. doi: 10.1109/LSP.2014.2349351. Recuperado de: http://ieeexplore.ieee.org/document/6880779/

4.2.2.5 mat Image::Filtering (arma::mat , int , int)

Aplica un filtrado usando el promedio o la mediana para remover ruido.

Descripcion: Permite aplicar un filtrado a una matriz, utilizando el sparse. Se selecciona una ventana de lxl a la cual se le aplica un promedio o la mediana para restaurar la imagen. Se calcula utilizando el tamaño de la ventana l y el pixel central de la ventana, luego se aplica el promedio o mediana de los pixels adyacentes al central.

Parameters

matrix	es la matriz con ruido a la cual se le aplicara el filtrado.
l	es el tamaño de la ventana
flag	es la bandera que indica si se efectua el promedio o la mediana. Donde el promedio es igual 0 y la mediana igual a 1

Returns

filt_matrix: Retorna la matriz con el filtrado.

Test

filtering_median.cpp filtering_mean.cpp
mat matrix;
int l = 5;
int flag = 0;
Filtering(matrix, l, flag);

4.2.2.6 void Image::Four_windows_opencv (const char * *imag1,* const char * *imag2,* const char * *imag3,* const char * *imag4,* const char * *label1,* const char * *label2,* const char * *label3,* const char * *label4*)

Carga cuatro imagenes en una ventana de opencv.

Descripcion: Permite cargar cuatro imagenes en una ventana de opencv

Parameters

imag1	es la ruta o ubicación de la imagen1
imag2	es la ruta o ubicación de la imagen2 ....Imag3 ... Imag4
label1	es el nombre de la imagen1
label2	es el nombre de la imagen2 ....Imag3 ... Imag4

Returns

Abre la ventana opencv con cuatro imagenes

Test

main.cpp

```
const char* imag1= "src/Resources/images_salt_and_pepper/lady_256_0_1.pgm";
const char* imag2= "src/Resources/images_salt_and_pepper/lady_256_1.pgm";
const char* imag3= "src/Resources/images_salt_and_pepper/lady_256_10.pgm";
const char* imag4= "src/Resources/images_salt_and_pepper/lady_256_100.pgm";
Four_windows_opencv(imag1, imag2, imag3, imag4, label1, label2, label3, label4);
```

4.2.2.7 mat Image::Gaussian_noise (double *nivel*)

Aplica Ruido Gaussiano a una matriz.

Descripcion: Permite aplicarle Ruido Gaussiano a una matriz. El Ruido Gaussiano es una matriz de media 0 y con una desviacion estandar variable.

Parameters

nivel	es el nivel de varianza variable Parametros Internos: A es una matriz gaussiana de dimensiones iguales a la matriz y de media 0 y varianza 1 B es una matriz donde se almacena la matriz gaussiana de media 0 y el calculo de una varianza determinada std es la desviacion estandar

Returns

value: Retorna la matriz con el ruido aplicado

Test

gaussian_noise.cpp

int nivel=0.1;
Gaussian_noise(nivel1);

4.2.2.8 int Image::Get_pixel (int *row*, int *col*)

Obtiene el valor de un pixel.

Descripcion: Permite obtener el valor de un pixel de la matriz de la imagen original

Parameters

row	es el numero de filas de la matriz
col	es el numero de columnas de la matriz

Returns

Retorna el valor (value) de un pixel

Test

basic_funcions.cpp
Image I;
int row=5;
int col=5;
I.Get_pixel(row, col);

4.2.2.9 void Image::Get_pixels ( )

Imprime todos los pixeles de la matriz imagen.

Descripcion: Permite obtener e imprimir todos los pixeles de la imagen.

Returns

value: Retorna todos los pixeles.

Test

basic_funcions.cpp
Image I;
I.Get_pixels();

4.2.2.10 mat Image::Idtc_Robusta (arma::mat , int)

Aproximación Robusta por bloques solpados (ventana 8x8)

Descripcion: Permite remover ruido mediante el uso de una ventana 8x8 solapado por media (0) y mediana ponderada (1).

Basado en los Articulos:

Ramírez J. y Paredes J. (2014). Robust Sparse Recovery Base On Weighted Median Operator. IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP). Department of Electrical Engineering, Universidad de Los Andes, Mérida, Venezuela. Doi: 978-1-4799-2893-4/14/$31.00. Recuperado de: http://www.mirlab.org/conference_papers/International_Conference/ICASSP%202014/papers/p1050-ramirez.pdf

Ramírez J. y Paredes J. (2015). Robust Transforms Based on the Weighted Median Operator. IEEE Signal Processing Letters, 22(1), pp. 120 – 124. doi: 10.1109/LSP.2014.2349351. Recuperado de: http://ieeexplore.ieee.org/document/6880779/

Parameters

matrix_r	es la matriz con ruido.
flag	es la bandera que indica si se efectua el promedio o la mediana. Donde el promedio es igual 0 y la mediana igual a 1

Returns

Retorna la matriz sin ruido.

Test

idtc_Robusta_median.cpp idtc_Robusta_mean.cpp

```
mat matrix;
int flag = 0;
Idtc_Robusta(matrix, flag);
```

4.2.2.11 mat Image::Image_load (arma::mat , std::string)

Carga de Imagen pgm a mat.

Descripcion: Permite cargar la imagen en formato pgm a mat

Parameters

matrix	es la matriz que contine los pixeles de la imagen.
ruta	es la ubicacion o localizacion de la imagen.

Returns

Retorna si la matriz fue cargada o si hubo problemas en la carga.

Test

basic_funcions.cpp

```
mat matrix;
string ruta = "src/Resources/images/house.256.pgm";
Image_load(matrix, ruta);
```

4.2.2.12 mat Image::Impulsive_uniform_noise (double *nivel*)

Aplica Ruido Impulsivo Uniforme a una matriz.

Descripcion: Permite aplicarle Ruido Impulsivo Uniforme a una matriz

Parameters

nivel	es el porcentaje de ruido.

Returns

value: Retorna la matriz con el ruido aplicado.

Test

uniform_impulsive_noise.cpp
double nivel=0.1;
Impulsive_uniform_noise(nivel);

4.2.2.13 mat Image::Lost_pixels_noise (double *nivel*)

Aplica Perdida de pixeles a una matriz.

Descripcion: Permite aplicarle Perdida de Pixeles a una matriz

Parameters

nivel	es el porcentaje de ruido

Returns

value: Retorna la matriz con el ruido aplicado

Test

lost_Pixels_noise.cpp
double nivel=0.1;
Lost_pixels_noise(nivel);

4.2.2.14 double Image::MAE (arma::mat , arma::mat)

Calculo del MAE.

Descripcion: Permite calcular el MAE (Error promedio absoluto) de una imagen con ruido y una imagen filtrada

Parameters

matrix_r	es la matriz con ruido.
matriz_f	es la matriz filtrada o recuperada

Returns

Retorna el valor del MAE

Test

filtering_mean.cpp
mat matrix_r;
mat matriz_f;
MAE(matrix_r, matriz_f);

4.2.2.15 double Image::MSE (arma::mat , arma::mat)

Calculo del MSE.

Descripcion: Permite calcular el MSE (Error cuadratico medio) de una imagen con ruido y una imagen filtrada

Parameters

matrix_r	es la matriz con ruido.
matriz_f	es la matriz filtrada o recuperada

Returns

Retorna el valor del MSE

Test

filtering_mean.cpp
mat matrix_r;
mat matriz_f;
MSE(matrix_r, matriz_f);

4.2.2.16 mat Image::Noise_remover (arma::mat , int)

Aproximación Robusta por bloques No solpados (ventana 8x8)

Descripcion: Permite remover ruido mediante el uso de una ventana 8x8 no solapado

Parameters

matrix	es la matriz que se le removerá el ruido.
image_tam	es el tamaño de la imagen.

Returns

removermatrix: Retorna la matriz sin ruido.

Test

noise_remover.cpp

```
mat matrix;
int image_tam = 256;
Noise_remover(matrix, image_tam);
```

4.2.2.17 mat Image::Overlap (arma::mat)

Aproximación por bloques solpados (ventana 8x8)

Descripcion: Permite remover ruido mediante el uso de una ventana 8x8 solapado

Parameters

matrix	es la matriz que se le removerá el ruido.

Returns

removermatrix: Retorna la matriz sin ruido.

Test

overlap.cpp

```
mat matrix;
Overlap(matrix);
```

4.2.2.18 double Image::PSNR (arma::mat , arma::mat)

Calculo del PSNR.

Descripcion: Permite calcular el PSNR de una imagen con ruido y una imagen filtrada

Parameters

matrix_r	es la matriz con ruido.
matriz_f	es la matriz filtrada o recuperada

Returns

Retorna el valor del PSNR

Test

filtering_mean.cpp

```
mat matrix_r;
mat matriz_f;
PSNR(matrix_r, matriz_f);
```

4.2.2.19 mat Image::Salt_and_pepper_noise (double *nivel*)

Aplica Ruido Sal y Pimienta a una matriz.

Descripcion: Permite aplicarle Ruido Sal y Pimienta a una matriz

Parameters

nivel	es el porcentaje de ruido

Returns

Retorna la matriz con el ruido (RG)

Test

salt_pepper.cpp
double nivel=0.1;
Salt_and_pepper_noise(nivel);

4.2.2.20 void Image::SaveImage (arma::mat , std::string)

Guarda la matriz en formato pgm.

Descripcion: Permite guardar una matriz mat en formato pgm

Parameters

matrix	matriz que contine los pixeles de la imagen.
ruta	ubicacion o localizacion de la imagen.

Returns

Retorna si la matriz fue guardada o si hubo problemas al guardar.

Test

gaussian_noise.cpp
mat matrix;
string ruta= "src/Resources/images/house.256.pgm";
SaveImage(matrix, ruta);

4.2.2.21 void Image::Two_windows_opencv (const char * , const char * , std::string , std::string)

Carga dos imagenes en una ventana de opencv.

Descripcion: Permite cargar dos imagenes en una ventana de opencv con sus respectivas etiquetas (label).

Parameters

imag1	es la ruta o ubicación de la imagen1.
imag2	es la ruta o ubicación de la imagen2.
label1	es el nombre de la imagen1.
label2	es el nombre de la imagen2.

Returns

Abre la ventana opencv con dos imagenes.

Test

main.cpp
const char* imag1= "src/Resources/images_salt_and_pepper/lady_256_0_1.pgm";
const char* imag2= "src/Resources/images_salt_and_pepper/lady_256_1.pgm";
string label1 = "Imagen1";
string label2 = "Imagen2";
Two_windows_opencv(imag1, imag2, label1, label2);

4.2.2.22 double Image::wmedianf (arma::vec & , arma::vec &)

Calculo de la mediana.

Descripcion: Funcion necesaria para idt_Robusta()

Basado en los Articulos:

Ramírez J. y Paredes J. (2014). Robust Sparse Recovery Base On Weighted Median Operator. IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP). Department of Electrical Engineering, Universidad de Los Andes, Mérida, Venezuela. Doi: 978-1-4799-2893-4/14/$31.00. Recuperado de: http://www.mirlab.org/conference papers/International_Conference/ICASSP%202014/papers/p1050-ramirez.pdf

Ramírez J. y Paredes J. (2015). Robust Transforms Based on the Weighted Median Operator. IEEE Signal Processing Letters, 22(1), pp. 120 – 124. doi: 10.1109/LSP.2014.2349351. Recuperado de: http://ieeexplore.ieee.org/document/6880779/

4.2.3 Member Data Documentation

4.2.3.1 int Image::cols [private]

4.2.3.2 arma::mat Image::matrix [private]

4.2.3.3 int Image::rows [private]

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

- include/Image.hpp
- src/Detection_Face_opencv.cpp
- src/Four_windows_opencv.cpp
- src/Image_load.cpp
- src/Two_windows_opencv.cpp

Chapter 5

File Documentation

5.1 examples/basic_funcions.cpp File Reference

```
#include "../include/Image.hpp"
#include <iostream>
#include <armadillo>
```

Functions

- int main ()

5.1.1 Detailed Description

5.1.2 Function Documentation

5.1.2.1 int main ( )

5.2 examples/detection_face.cpp File Reference

```
#include "../include/Image.hpp"
#include <iostream>
#include <armadillo>
```

Functions

- int main ()

5.2.1 Function Documentation

5.2.1.1 int main ( )

5.3 examples/filtering_mean.cpp File Reference

```
#include <iostream>
#include <armadillo>
#include "Image.hpp"
```

Functions

- int main ()

5.3.1 Function Documentation

5.3.1.1 int main ( )

5.4 examples/filtering_median.cpp File Reference

```
#include <iostream>
#include <armadillo>
#include "Image.hpp"
```

Functions

- int main ()

5.4.1 Function Documentation

5.4.1.1 int main ( )

5.5 examples/gaussian_noise.cpp File Reference

```
#include <iostream>
#include <armadillo>
#include "Image.hpp"
```

Functions

- int main ()

5.5.1 Function Documentation

5.5.1.1 int main ( )

5.6 examples/idtc_Robusta_mean.cpp File Reference

```
#include <iostream>
#include <armadillo>
#include "Image.hpp"
```

Functions

- int main ()

5.6.1 Function Documentation

5.6.1.1 int main ( )

5.7 examples/idtc_Robusta_median.cpp File Reference

```
#include <iostream>
#include <armadillo>
#include "Image.hpp"
```

Functions

- int main ()

5.7.1 Function Documentation

5.7.1.1 int main ( )

5.8 examples/lost_Pixels_noise.cpp File Reference

```
#include <iostream>
#include <armadillo>
```

```
#include "Image.hpp"
```

Functions

- int main ()

5.8.1 Function Documentation

5.8.1.1 int main ( )

5.9 examples/noise_remover.cpp File Reference

```
#include <iostream>
#include <armadillo>
#include "Image.hpp"
```

Functions

- int main ()

5.9.1 Function Documentation

5.9.1.1 int main ( )

5.10 examples/overlap.cpp File Reference

```
#include <iostream>
#include <armadillo>
#include "Image.hpp"
```

Functions

- int main ()

5.10.1 Function Documentation

5.10.1.1 int main ( )

5.11 examples/salt_pepper.cpp File Reference

```
#include <iostream>
#include <armadillo>
#include "Image.hpp"
```

Functions

- int main ()

5.11.1 Function Documentation

5.11.1.1 int main ( )

5.12 examples/uniform_impulsive_noise.cpp File Reference

```
#include <iostream>
#include <armadillo>
#include "Image.hpp"
```

Functions

- int main ()

5.12.1 Function Documentation

5.12.1.1 int main ( )

5.13 include/Image.hpp File Reference

```
#include <iostream>
#include <armadillo>
```

Classes

- class Image

5.14 src/Detection_Face_opencv.cpp File Reference

```
#include "../include/Image.hpp"
#include <opencv2/imgproc/imgproc.hpp>
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/core/utility.hpp>
#include <opencv2/opencv.hpp>
#include <iostream>
```

Variables

- CascadeClassifier face_cascade

5.14.1 Variable Documentation

5.14.1.1 CascadeClassifier face_cascade

5.15 src/Four_windows_opencv.cpp File Reference

```
#include "../include/Image.hpp"
#include <opencv2/imgproc/imgproc.hpp>
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/core/utility.hpp>
#include <iostream>
```

5.16 src/Image_load.cpp File Reference

```
#include "../include/Image.hpp"
#include <iostream>
#include <armadillo>
```

Functions

- double Average_Median (vec vector, int flag)

  Calcula el promedio y la mediana.

5.16.1 Function Documentation

5.16.1.1 double Image::Average_Median (vec *vector,* int *flag*)

Calcula el promedio y la mediana.

Descripcion: Permite calcular el promedio y la mediana.

Parameters

vector	es el vector al cual se le calculará el promedio o la mediana
flag	cuando flag = 0 se calcula el promedio y cuando flag = 1 se calcula la mediana

Returns

value: Retorna la matriz con el ruido
vec B;
Average_Median(B, 0); //Promedio-Media
Average_Median(B, 1); //Mediana

5.17 src/main.cpp File Reference

```
#include <iostream>
#include <armadillo>
#include "../include/Image.hpp"
```

Functions

- int main ()

  Titulo: Funcion de Inicio.

5.17.1 Function Documentation

5.17.1.1 int main ( )

Titulo: Funcion de Inicio.

Descripcion: Comienzo de codigo.

FA = I.Filtering(SC, 5);

5.18 src/Two_windows_opencv.cpp File Reference

```
#include "../include/Image.hpp"
```

```
#include <opencv2/imgproc/imgproc.hpp>
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/core/utility.hpp>
#include <iostream>
```

Index

~Image
    Image, 11

Average_Median
    Image_load.cpp, 27

basic_funcions.cpp
    main, 21

cols
    Image, 20

Detection_Face
    Image, 11
detection_face.cpp
    main, 22
Detection_Face_opencv.cpp
    face_cascade, 26
Diccionary
    Image, 11
Dimension
    Image, 11

examples/basic_funcions.cpp, 21
examples/detection_face.cpp, 21
examples/filtering_mean.cpp, 22
examples/filtering_median.cpp, 22
examples/gaussian_noise.cpp, 22
examples/idtc_Robusta_mean.cpp, 23
examples/idtc_Robusta_median.cpp, 23
examples/lost_Pixels_noise.cpp, 23
examples/noise_remover.cpp, 24
examples/overlap.cpp, 24
examples/salt_pepper.cpp, 25
examples/uniform_impulsive_noise.cpp, 25

face_cascade
    Detection_Face_opencv.cpp, 26
Fast_awmr
    Image, 12
Filtering
    Image, 12
filtering_mean.cpp
    main, 22
filtering_median.cpp
    main, 22
Four_windows_opencv
    Image, 12

Gaussian_noise
    Image, 13
gaussian_noise.cpp
    main, 23
Get_pixel
    Image, 14
Get_pixels
    Image, 14

Idtc_Robusta
    Image, 14
idtc_Robusta_mean.cpp
    main, 23
idtc_Robusta_median.cpp
    main, 23
Image, 9
    ~Image, 11
    cols, 20
    Detection_Face, 11
    Diccionary, 11
    Dimension, 11
    Fast_awmr, 12
    Filtering, 12
    Four_windows_opencv, 12
    Gaussian_noise, 13
    Get_pixel, 14
    Get_pixels, 14
    Idtc_Robusta, 14
    Image, 11
    Image_load, 15
    Impulsive_uniform_noise, 15
    Lost_pixels_noise, 16
    MAE, 16
    matrix, 20

MSE, 17
Noise_remover, 17
Overlap, 18
PSNR, 18
rows, 20
Salt_and_pepper_noise, 18
SaveImage, 19
Two_windows_opencv, 19
wmedianf, 20
Image::h, 9
Image_load
Image, 15
Image_load.cpp
Average_Median, 27
Impulsive_uniform_noise
Image, 15
include/Image.hpp, 25

Lost_pixels_noise
Image, 16
lost_Pixels_noise.cpp
main, 24

MAE
Image, 16
main
basic_funcions.cpp, 21
detection_face.cpp, 22
filtering_mean.cpp, 22
filtering_median.cpp, 22
gaussian_noise.cpp, 23
idtc_Robusta_mean.cpp, 23
idtc_Robusta_median.cpp, 23
lost_Pixels_noise.cpp, 24
main.cpp, 27
noise_remover.cpp, 24
overlap.cpp, 24
salt_pepper.cpp, 25
uniform_impulsive_noise.cpp, 25
main.cpp
main, 27
matrix
Image, 20
MSE
Image, 17

Noise_remover
Image, 17
noise_remover.cpp
main, 24

Overlap
Image, 18
overlap.cpp
main, 24

PSNR
Image, 18

rows
Image, 20

Salt_and_pepper_noise
Image, 18
salt_pepper.cpp
main, 25
SaveImage
Image, 19
src/Detection_Face_opencv.cpp, 26
src/Four_windows_opencv.cpp, 26
src/Image_load.cpp, 26
src/main.cpp, 27
src/Two_windows_opencv.cpp, 27

Two_windows_opencv
Image, 19

uniform_impulsive_noise.cpp
main, 25

wmedianf
Image, 20