

# Imageprocessing

## 1.0

Generated by Doxygen 1.7.4

Wed Apr 5 2017 17:00:50



# Contents

<b>1</b>	<b>Test List</b>	<b>1</b>
<b>2</b>	<b>Class Index</b>	<b>5</b>
2.1	Class List . . . . .	5
<b>3</b>	<b>File Index</b>	<b>7</b>
3.1	File List . . . . .	7
<b>4</b>	<b>Class Documentation</b>	<b>9</b>
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	Diccionario . . . . .	11
4.2.2.2	Dimension . . . . .	11
4.2.2.3	Fast_awmr . . . . .	11
4.2.2.4	Filtering . . . . .	11
4.2.2.5	Four_windows_opencv . . . . .	12
4.2.2.6	Gaussian_noise . . . . .	12
4.2.2.7	Get_pixel . . . . .	13
4.2.2.8	Get_pixels . . . . .	13
4.2.2.9	Idtc_Robusta . . . . .	14
4.2.2.10	Image_load . . . . .	14

4.2.2.11	<a href="#">Impulsive_uniform_noise</a>	14
4.2.2.12	<a href="#">Lost_pixels_noise</a>	15
4.2.2.13	<a href="#">MAE</a>	15
4.2.2.14	<a href="#">MSE</a>	15
4.2.2.15	<a href="#">Noise_remover</a>	16
4.2.2.16	<a href="#">Overlap</a>	16
4.2.2.17	<a href="#">PSNR</a>	17
4.2.2.18	<a href="#">Salt_and_pepper_noise</a>	17
4.2.2.19	<a href="#">SavelImage</a>	18
4.2.2.20	<a href="#">Two_windows_opencv</a>	18
4.2.2.21	<a href="#">wmedianf</a>	19
4.2.3	<a href="#">Member Data Documentation</a>	19
4.2.3.1	<a href="#">cols</a>	19
4.2.3.2	<a href="#">matrix</a>	19
4.2.3.3	<a href="#">rows</a>	19
<b>5</b>	<b><a href="#">File Documentation</a></b>	<b>21</b>
5.1	<a href="#">examples/basic_funcions.cpp File Reference</a>	21
5.1.1	<a href="#">Function Documentation</a>	21
5.1.1.1	<a href="#">main</a>	21
5.2	<a href="#">examples/filtering_mean.cpp File Reference</a>	21
5.2.1	<a href="#">Function Documentation</a>	21
5.2.1.1	<a href="#">main</a>	22
5.3	<a href="#">examples/filtering_median.cpp File Reference</a>	22
5.3.1	<a href="#">Function Documentation</a>	22
5.3.1.1	<a href="#">main</a>	22
5.4	<a href="#">examples/gaussian_noise.cpp File Reference</a>	22
5.4.1	<a href="#">Function Documentation</a>	22
5.4.1.1	<a href="#">main</a>	22
5.5	<a href="#">examples/idtc_Robusta_mean.cpp File Reference</a>	22
5.5.1	<a href="#">Function Documentation</a>	23
5.5.1.1	<a href="#">main</a>	23
5.6	<a href="#">examples/idtc_Robusta_median.cpp File Reference</a>	23
5.6.1	<a href="#">Function Documentation</a>	23

5.6.1.1	main	23
5.7	<a href="#">examples/lost_Pixels_noise.cpp File Reference</a>	23
5.7.1	Function Documentation	23
5.7.1.1	main	23
5.8	<a href="#">examples/noise_remover.cpp File Reference</a>	23
5.8.1	Function Documentation	24
5.8.1.1	main	24
5.9	<a href="#">examples/overlap.cpp File Reference</a>	24
5.9.1	Function Documentation	24
5.9.1.1	main	24
5.10	<a href="#">examples/salt_pepper.cpp File Reference</a>	24
5.10.1	Function Documentation	24
5.10.1.1	main	24
5.11	<a href="#">examples/uniform_impulsive_noise.cpp File Reference</a>	24
5.11.1	Function Documentation	25
5.11.1.1	main	25
5.12	<a href="#">include/Image.hpp File Reference</a>	25
5.13	<a href="#">src/Four_windows_opencv.cpp File Reference</a>	25
5.14	<a href="#">src/Image_load.cpp File Reference</a>	25
5.14.1	Function Documentation	26
5.14.1.1	Average_Median	26
5.15	<a href="#">src/main.cpp File Reference</a>	26
5.15.1	Function Documentation	26
5.15.1.1	main	26
5.16	<a href="#">src/Two_windows_opencv.cpp File Reference</a>	26



# 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 char *, const char *)`** [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;  
Impulsive_uniform_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::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(int)`** `uniform_impulsive_noise.cpp`

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

**Member `Image::Lost_pixels_noise(int)`** `lost_Pixels_noise.cpp`

```
int 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`**(int) [salt\\_pepper.cpp](#)

```
int 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:

<a href="#">Image::h</a>	9
<a href="#">Image</a>	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/filtering_mean.cpp . . . . .	21
examples/filtering_median.cpp . . . . .	22
examples/gaussian_noise.cpp . . . . .	22
examples/idtc_Robusta_mean.cpp . . . . .	22
examples/idtc_Robusta_median.cpp . . . . .	23
examples/lost_Pixels_noise.cpp . . . . .	23
examples/noise_remover.cpp . . . . .	23
examples/overlap.cpp . . . . .	24
examples/salt_pepper.cpp . . . . .	24
examples/uniform_impulsive_noise.cpp . . . . .	24
include/Image.hpp . . . . .	25
src/Four_windows_opencv.cpp . . . . .	25
src/Image_load.cpp . . . . .	25
src/main.cpp . . . . .	26
src/Two_windows_opencv.cpp . . . . .	26



## 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](#) (int)  
*Aplica Perdida de pixeles a una matriz.*
- arma::mat [Salt\\_and\\_pepper\\_noise](#) (int)  
*Aplica Ruido Sal y Pimienta a una matriz.*
- arma::mat [Impulsive\\_uniform\\_noise](#) (int)  
*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 [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)  
*Remocion de Ruido no solapado (ventana 8x8)*
- arma::mat [Overlap](#) (arma::mat)  
*Remocion de Ruido solapado (ventana 8x8)*
- arma::mat [ldtc\\_Robusta](#) (arma::mat, int)
- double [wmedianf](#) (arma::vec &, arma::vec &)
- arma::mat [Diccionario](#) ()
- arma::vec [Fast\\_awmr](#) (arma::vec &y, arma::mat &A, int sparsity, int itmax, double beta, double tol, double epsilon, int numcoefperiter, double Kpar)
- [~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_functions.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 mat Image::Diccionario ( )

### 4.2.2.2 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_functions.cpp  
Image I;  
I.Dimension();
```

### 4.2.2.3 vec Image::Fast\_awmr ( arma::vec & y, arma::mat & A, int *sparsity*, int *itmax*, double *beta*, double *tol*, double *epsilon*, int *numcoefperiter*, double *Kpar* )

### 4.2.2.4 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  $l \times l$  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**

<i>matrix</i>	es la matriz con ruido a la cual se le aplicara el filtrado.
<i>l</i>	es el tamaño de la ventana
<i>flag</i>	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);
```

4.2.2.5 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**

<i>imag1</i>	es la ruta o ubicación de la imagen1
<i>imag2</i>	es la ruta o ubicación de la imagen2 ....Imag3 ... Imag4
<i>label1</i>	es el nombre de la imagen1
<i>label2</i>	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);
```

4.2.2.6 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

<i>nivel</i>	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;  
Impulsive_uniform_noise(nivel1);
```

##### 4.2.2.7 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

<i>row</i>	es el numero de filas de la matriz
<i>col</i>	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);
```

##### 4.2.2.8 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();
```

4.2.2.9 `mat Image::ldtc.Robusta ( arma::mat , int )`

4.2.2.10 `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**

<i>matrix</i>	es la matriz que contine los pixeles de la imagen.
<i>ruta</i>	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);
```

4.2.2.11 `mat Image::Impulsive_uniform_noise ( int nivel )`

Aplica Ruido Impulsivo Uniforme a una matriz.

Descripcion: Permite aplicarle Ruido Impulsivo Uniforme a una matriz

**Parameters**

<i>nivel</i>	es el porcentaje de ruido.
--------------	----------------------------

**Returns**

value: Retorna la matriz con el ruido aplicado.

**Test**

```
uniform_impulsive_noise.cpp
int nivel=0.1;
```

```
Impulsive_uniform_noise(nivel);
```

#### 4.2.2.12 `mat Image::Lost_pixels_noise ( int nivel )`

Aplica Perdida de pixeles a una matriz.

Descripcion: Permite aplicarle Perdida de Pixeles a una matriz

##### Parameters

<i>nivel</i>	es el porcentaje de ruido
--------------	---------------------------

##### Returns

value: Retorna la matriz con el ruido aplicado

##### Test

```
lost_Pixels_noise.cpp  
int nivel=0.1;  
Lost_pixels_noise(nivel);
```

#### 4.2.2.13 `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

<i>matrix_r</i>	es la matriz con ruido.
<i>matriz_f</i>	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);
```

#### 4.2.2.14 `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

<i>matrix_r</i>	es la matriz con ruido.
<i>matriz_f</i>	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);
```

#### 4.2.2.15 mat Image::Noise\_remover ( arma::mat , int )

Remocion de Ruido no solapado (ventana 8x8)

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

#### Parameters

<i>matrix</i>	es la matriz que se le removerá el ruido.
<i>image_tam</i>	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);
```

#### 4.2.2.16 mat Image::Overlap ( arma::mat )

Remocion de Ruido solapado (ventana 8x8)

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

#### Parameters

<i>matrix</i>	es la matriz que se le removerá el ruido.
---------------	-------------------------------------------

**Returns**

removematrix: Retorna la matriz sin ruido.

**Test**

```
overlap.cpp  
mat matrix;  
Overlap(matrix);
```

**4.2.2.17 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**

<i>matrix_r</i>	es la matriz con ruido.
<i>matriz_f</i>	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);
```

**4.2.2.18 mat Image::Salt\_and\_pepper\_noise ( int nivel )**

Aplica Ruido Sal y Pimienta a una matriz.

Descripcion: Permite aplicarle Ruido Sal y Pimienta a una matriz

**Parameters**

<i>nivel</i>	es el porcentaje de ruido
--------------	---------------------------

**Returns**

Retorna la matriz con el ruido (RG)

**Test**

```
salt_pepper.cpp  
int nivel=0.1;  
Salt_and_pepper_noise(nivel);
```

#### 4.2.2.19 void Image::SaveImage ( arma::mat , std::string )

Guarda la matriz en formato pgm.

Descripcion: Permite guardar una matriz mat en formato pgm

##### Parameters

<i>matrix</i>	matriz que contine los pixeles de la imagen.
<i>ruta</i>	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);
```

#### 4.2.2.20 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

<i>imag1</i>	es la ruta o ubicación de la imagen1.
<i>imag2</i>	es la ruta o ubicación de la imagen2.
<i>label1</i>	es el nombre de la imagen1.
<i>label2</i>	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);
```



4.2.2.21 `double Image::wmedianf ( arma::vec & , arma::vec & )`

### 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/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 Function Documentation

5.1.1.1 int main ( )

### 5.2 examples/filtering\_mean.cpp File Reference

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

#### Functions

- int `main` ()

#### 5.2.1 Function Documentation

5.2.1.1 int main ( )

### 5.3 examples/filtering\_median.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/gaussian\_noise.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/idtc\_Robusta\_mean.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/ldtc\_Robusta\_median.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/lost\_Pixels\_noise.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/noise\_remover.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/overlap.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/salt\_pepper.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/uniform\_impulsive\_noise.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 include/Image.hpp File Reference

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

### Classes

- class [Image](#)

## 5.13 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>
```

## 5.14 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.*

### 5.14.1 Function Documentation

#### 5.14.1.1 `double Image::Average_Median ( vec vector, int flag )`

Calcula el promedio y la mediana.

Descripcion: Permite calcular el promedio y la mediana.

#### Parameters

<i>vector</i>	es el vector al cual se le calculará el promedio o la mediana
<i>flag</i>	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

## 5.15 src/main.cpp File Reference

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

### Functions

- `int main ()`  
*Titulo: Funcion de Inicio.*

### 5.15.1 Function Documentation

#### 5.15.1.1 `int main ( )`

Titulo: Funcion de Inicio.

Descripcion: Comienzo de codigo.

FA = I.Filtering(SC, 5);

## 5.16 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, 26

basic\_funcions.cpp  
    main, 21

cols  
    Image, 19

Diccionario  
    Image, 11

Dimension  
    Image, 11

examples/basic\_funcions.cpp, 21  
examples/filtering\_mean.cpp, 21  
examples/filtering\_median.cpp, 22  
examples/gaussian\_noise.cpp, 22  
examples/ldtc\_Robusta\_mean.cpp, 22  
examples/ldtc\_Robusta\_median.cpp, 23  
examples/lost\_Pixels\_noise.cpp, 23  
examples/noise\_remover.cpp, 23  
examples/overlap.cpp, 24  
examples/salt\_pepper.cpp, 24  
examples/uniform\_impulsive\_noise.cpp, 24

Fast\_awmr  
    Image, 11

Filtering  
    Image, 11

filtering\_mean.cpp  
    main, 21

filtering\_median.cpp  
    main, 22

Four\_windows\_opencv  
    Image, 12

Gaussian\_noise  
    Image, 12

gaussian\_noise.cpp  
    main, 22

Get\_pixel  
    Image, 13

Get\_pixels  
    Image, 13

ldtc\_Robusta  
    Image, 14

ldtc\_Robusta\_mean.cpp  
    main, 23

ldtc\_Robusta\_median.cpp  
    main, 23

Image, 9  
    ~Image, 11  
    cols, 19  
    Diccionario, 11  
    Dimension, 11  
    Fast\_awmr, 11  
    Filtering, 11  
    Four\_windows\_opencv, 12  
    Gaussian\_noise, 12  
    Get\_pixel, 13  
    Get\_pixels, 13  
    ldtc\_Robusta, 14  
    Image, 11  
    Image\_load, 14  
    Impulsive\_uniform\_noise, 14  
    Lost\_pixels\_noise, 15  
    MAE, 15  
    matrix, 19  
    MSE, 15  
    Noise\_remover, 16  
    Overlap, 16  
    PSNR, 17  
    rows, 19  
    Salt\_and\_pepper\_noise, 17  
    Savelmage, 17  
    Two\_windows\_opencv, 18  
    wmedianf, 18

Image::h, 9

Image\_load  
  Image, [14](#)  
Image\_load.cpp  
  Average\_Median, [26](#)  
Impulsive\_uniform\_noise  
  Image, [14](#)  
include/Image.hpp, [25](#)  
  
Lost\_pixels\_noise  
  Image, [15](#)  
lost\_Pixels\_noise.cpp  
  main, [23](#)  
  
MAE  
  Image, [15](#)  
main  
  basic\_fuctions.cpp, [21](#)  
  filtering\_mean.cpp, [21](#)  
  filtering\_median.cpp, [22](#)  
  gaussian\_noise.cpp, [22](#)  
  idtc\_Robusta\_mean.cpp, [23](#)  
  idtc\_Robusta\_median.cpp, [23](#)  
  lost\_Pixels\_noise.cpp, [23](#)  
  main.cpp, [26](#)  
  noise\_remover.cpp, [24](#)  
  overlap.cpp, [24](#)  
  salt\_pepper.cpp, [24](#)  
  uniform\_impulsive\_noise.cpp, [25](#)  
main.cpp  
  main, [26](#)  
matrix  
  Image, [19](#)  
MSE  
  Image, [15](#)  
  
Noise\_remover  
  Image, [16](#)  
noise\_remover.cpp  
  main, [24](#)  
  
Overlap  
  Image, [16](#)  
overlap.cpp  
  main, [24](#)  
  
PSNR  
  Image, [17](#)  
  
rows  
  Image, [19](#)  
  
Salt\_and\_pepper\_noise  
  Image, [17](#)  
salt\_pepper.cpp  
  main, [24](#)  
SavelImage  
  Image, [17](#)  
src/Four\_windows\_opencv.cpp, [25](#)  
src/Image\_load.cpp, [25](#)  
src/main.cpp, [26](#)  
src/Two\_windows\_opencv.cpp, [26](#)  
  
Two\_windows\_opencv  
  Image, [18](#)  
  
uniform\_impulsive\_noise.cpp  
  main, [25](#)  
  
wmedianf  
  Image, [18](#)