

UAVAgroState

Generated by Doxygen 1.8.15

Contents

1	Class Index	1
1.1	Class List	1
2	Class Documentation	3
2.1	Calibration Class Reference	3
2.1.1	Detailed Description	3
2.1.2	Member Function Documentation	3
2.1.2.1	calibrateImg()	3
2.1.2.2	obtenerInput()	4
2.1.2.3	readCalibrationMat()	4
2.1.2.4	storeCalibrationMat()	4
2.2	Color Class Reference	5
2.2.1	Detailed Description	8
2.2.2	Member Function Documentation	8
2.2.2.1	generarColores()	8
2.2.2.2	scalarToVec()	8
2.3	CommonFunctions Class Reference	9
2.3.1	Detailed Description	10
2.3.2	Member Function Documentation	10
2.3.2.1	addAlpha()	10
2.3.2.2	addTransparence()	11
2.3.2.3	boundingBox() [1/2]	11
2.3.2.4	boundingBox() [2/2]	12
2.3.2.5	cargarImagen()	12

2.3.2.6	cargarImágenes()	12
2.3.2.7	copyToTransparent()	13
2.3.2.8	crearCarpeta()	13
2.3.2.9	cropRectorROI()	14
2.3.2.10	escribirImagen()	14
2.3.2.11	escribirPDF()	14
2.3.2.12	existFile()	15
2.3.2.13	fToS()	15
2.3.2.14	histDraw()	16
2.3.2.15	info()	16
2.3.2.16	manejarCarpeta()	16
2.3.2.17	nameToDate()	17
2.3.2.18	nowDate()	17
2.3.2.19	obtenerFecha()	17
2.3.2.20	obtenerImágenes()	18
2.3.2.21	obtenerParametro()	18
2.3.2.22	obtenerUltimoDirectorio()	18
2.3.2.23	obtenerUltimoDirectorio2()	19
2.3.2.24	rectROI()	19
2.3.2.25	removeAlpha()	19
2.3.2.26	removerExtension()	20
2.3.2.27	showWindowNormal()	20
2.3.2.28	stringAzul()	20
2.3.2.29	stringRojo()	21
2.3.2.30	stringVerde()	21
2.3.2.31	tiempo()	21
2.4	FSManager Class Reference	22
2.4.1	Detailed Description	23
2.4.2	Constructor & Destructor Documentation	23
2.4.2.1	FSManager()	23

2.4.3	Member Function Documentation	23
2.4.3.1	appendFloat()	23
2.4.3.2	appendInt()	24
2.4.3.3	appendString()	24
2.4.3.4	appendVFloat()	24
2.4.3.5	appendVInt()	24
2.4.3.6	existeFloat()	25
2.4.3.7	existeInt()	25
2.4.3.8	existeString()	25
2.4.3.9	existeVFloat()	27
2.4.3.10	existeVInt()	27
2.4.3.11	readFloat()	28
2.4.3.12	readInt()	28
2.4.3.13	readString()	28
2.4.3.14	readVFloat()	29
2.4.3.15	readVInt()	29
2.5	IndexCalculation Class Reference	29
2.5.1	Detailed Description	30
2.5.2	Member Function Documentation	31
2.5.2.1	escribirSegmentaciones()	31
2.5.2.2	indexCalcu()	31
2.5.2.3	indexCalcuMS()	31
2.5.2.4	indexCalcuRGB()	32
2.5.2.5	ndviCalculation()	32
2.5.2.6	ngrdiCalculation()	32
2.5.2.7	obtenerInput()	32
2.5.2.8	obtenerMSInput()	33
2.5.2.9	obtenerMSOutput()	33
2.5.2.10	obtenerRGBInput()	33
2.5.2.11	obtenerRGBOutput()	34

2.5.2.12	rgCalculation()	34
2.5.2.13	rviCalculation()	34
2.6	Performance Class Reference	34
2.6.1	Detailed Description	35
2.7	Redaction Class Reference	35
2.7.1	Detailed Description	36
2.7.2	Member Function Documentation	36
2.7.2.1	generateChartText()	36
2.7.2.2	generatePDF()	36
2.7.2.3	writeChart()	36
2.7.2.4	writeImg()	37
2.7.2.5	writeIndex()	37
2.7.2.6	writeLut()	38
2.7.2.7	writeOriginal()	38
2.7.2.8	writeText()	38
2.8	Segmentation Class Reference	39
2.8.1	Detailed Description	39
2.8.2	Member Function Documentation	40
2.8.2.1	createLut()	40
2.8.2.2	drawIndexOfIndex()	40
2.8.2.3	generarGrafico()	41
2.8.2.4	normalizeMat()	41
2.8.2.5	segmentationVariation()	41
2.8.2.6	separarSuelo()	42
2.9	uav::Stitcher Class Reference	42
2.9.1	Detailed Description	45
2.9.2	Member Function Documentation	45
2.9.2.1	compareMats()	45
2.9.2.2	copyToTransparent()	45
2.9.2.3	escribirOutput()	46

2.9.2.4	escribirOutputOrto()	46
2.9.2.5	evaluateHomography()	46
2.9.2.6	getHomography()	46
2.9.2.7	obtenerInput()	47
2.9.2.8	obtenerInputOrto()	47
2.9.2.9	obtenerOutputOrto()	47
2.9.2.10	obtenerOutputRF()	48
2.9.2.11	runAll()	48
2.9.2.12	stitchImgs()	48
2.9.2.13	stitchWarp()	48
2.10	Undistort Class Reference	49
2.10.1	Detailed Description	49
2.10.2	Member Function Documentation	49
2.10.2.1	escribirOutput()	49
2.10.2.2	obtenerInput()	50
2.10.2.3	undistortImgs()	50
2.11	UtilInformation Class Reference	50
2.11.1	Detailed Description	51
2.11.2	Member Function Documentation	51
2.11.2.1	calcularHectareas()	51
Index		53

Chapter 1

Class Index

1.1 Class List

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

Calibration	Clase utilizada para la obtención de la matriz de calibración	3
Color	Genera un conjunto de scalares que representan diferentes colores	5
CommonFunctions	Realiza funciones genéricas que pueden ser utiles para cualquier proceso de este software . .	9
FSManager	Maneja la metadata de la camara e imágenes	22
IndexCalculation	Calcula los indices de vegetación	29
Performance	Calcula los indices de desempeño	34
Redaction	Redacta el PDF final	35
Segmentation	Genera las diferentes segmentaciones	39
uav::Stitcher	Realiza el pegado de un conjunto de imágenes	42
Undistort	Clase utilizada para quitar la distorsión de un conjunto de imágenes	49
UtilInformation	Calcula los indices de desempeño	50

Chapter 2

Class Documentation

2.1 Calibration Class Reference

Clase utilizada para la obtención de la matriz de calibración.

```
#include <Calibration.h>
```

Static Public Member Functions

- static void [calibrateImg](#) (int numCornersHor, int numCornersVer, string cameraName)
Obtiene matriz de calibración.
- static void [storeCalibrationMat](#) (Mat intrinsic, Mat distCoeffs, string cameraName)
Escribe matriz de calibración.
- static vector< Mat > [readCalibrationMat](#) (string cameraName)
Lee matriz de calibración.
- static vector< string > [obtenerInput](#) ()
Obtiene las ubicaciones de las imágenes de entrada.

2.1.1 Detailed Description

Clase utilizada para la obtención de la matriz de calibración.

2.1.2 Member Function Documentation

2.1.2.1 [calibrateImg\(\)](#)

```
static void Calibration::calibrateImg (  
    int numCornersHor,  
    int numCornersVer,  
    string cameraName ) [inline], [static]
```

Obtiene matriz de calibración.

Parameters

<i>numCornersHor</i>	
<i>numCornersVer</i>	
<i>cameraName</i>	

2.1.2.2 obtenerInput()

```
static vector<string> Calibration::obtenerInput ( ) [inline], [static]
```

Obtiene las ubicaciones de las imágenes de entrada.

Returns

vector<string>

2.1.2.3 readCalibrationMat()

```
static vector<Mat> Calibration::readCalibrationMat (
    string cameraName ) [inline], [static]
```

Lee matriz de calibración.

Parameters

<i>cameraName</i>	
-------------------	--

Returns

vector<Mat>

2.1.2.4 storeCalibrationMat()

```
static void Calibration::storeCalibrationMat (
    Mat intrinsic,
    Mat distCoeffs,
    string cameraName ) [inline], [static]
```

Escribe matriz de calibración.

Parameters

<i>intrinsic</i>	
<i>distCoeffs</i>	
<i>cameraName</i>	

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

- Calibration.h

2.2 Color Class Reference

Genera un conjunto de scalares que representan diferentes colores.

```
#include <Color.h>
```

Static Public Member Functions

- static vector< Vec3f > [generarColores](#) ()
Genera un vector con varios colores.
- static Vec3f [scalarToVec](#) (Scalar sclColor)
Convierte un escalar a Vec3f.

Static Public Attributes

- static cv::Scalar **aliceblue**
- static cv::Scalar **antiquewhite**
- static cv::Scalar **aqua**
- static cv::Scalar **aquamarine**
- static cv::Scalar **azure**
- static cv::Scalar **beige**
- static cv::Scalar **bisque**
- static cv::Scalar **black**
- static cv::Scalar **blanchedalmond**
- static cv::Scalar **blue**
- static cv::Scalar **blueviolet**
- static cv::Scalar **brown**
- static cv::Scalar **burlywood**
- static cv::Scalar **cadetblue**
- static cv::Scalar **chartreuse**
- static cv::Scalar **chocolate**
- static cv::Scalar **coral**
- static cv::Scalar **cornflowerblue**
- static cv::Scalar **cornsilk**
- static cv::Scalar **crimson**
- static cv::Scalar **cyan**
- static cv::Scalar **darkblue**
- static cv::Scalar **darkcyan**
- static cv::Scalar **darkgoldenrod**

- static cv::Scalar **darkgray**
- static cv::Scalar **darkgreen**
- static cv::Scalar **darkgrey**
- static cv::Scalar **darkkhaki**
- static cv::Scalar **darkmagenta**
- static cv::Scalar **darkolivegreen**
- static cv::Scalar **darkorange**
- static cv::Scalar **darkorchid**
- static cv::Scalar **darkred**
- static cv::Scalar **darksalmon**
- static cv::Scalar **darkseagreen**
- static cv::Scalar **darkslateblue**
- static cv::Scalar **darkslategray**
- static cv::Scalar **darkslategrey**
- static cv::Scalar **darkturquoise**
- static cv::Scalar **darkviolet**
- static cv::Scalar **deeppink**
- static cv::Scalar **deepskyblue**
- static cv::Scalar **dimgray**
- static cv::Scalar **dimgrey**
- static cv::Scalar **dodgerblue**
- static cv::Scalar **firebrick**
- static cv::Scalar **floralwhite**
- static cv::Scalar **forestgreen**
- static cv::Scalar **fuchsia**
- static cv::Scalar **gainsboro**
- static cv::Scalar **ghostwhite**
- static cv::Scalar **gold**
- static cv::Scalar **goldenrod**
- static cv::Scalar **gray**
- static cv::Scalar **green**
- static cv::Scalar **greenyellow**
- static cv::Scalar **grey**
- static cv::Scalar **honeydew**
- static cv::Scalar **hotpink**
- static cv::Scalar **indianred**
- static cv::Scalar **indigo**
- static cv::Scalar **ivory**
- static cv::Scalar **khaki**
- static cv::Scalar **lavender**
- static cv::Scalar **lavenderblush**
- static cv::Scalar **lawngreen**
- static cv::Scalar **lemonchiffon**
- static cv::Scalar **lightblue**
- static cv::Scalar **lightcoral**
- static cv::Scalar **lightcyan**
- static cv::Scalar **lightgoldenrodyellow**
- static cv::Scalar **lightgray**
- static cv::Scalar **lightgreen**
- static cv::Scalar **lightgrey**
- static cv::Scalar **lightpink**
- static cv::Scalar **lightsalmon**
- static cv::Scalar **lightseagreen**
- static cv::Scalar **lightskyblue**
- static cv::Scalar **lightslategray**

- static cv::Scalar **lightslategrey**
- static cv::Scalar **lightsteelblue**
- static cv::Scalar **lightyellow**
- static cv::Scalar **lime**
- static cv::Scalar **limegreen**
- static cv::Scalar **linen**
- static cv::Scalar **magenta**
- static cv::Scalar **maroon**
- static cv::Scalar **mediumaquamarine**
- static cv::Scalar **mediumblue**
- static cv::Scalar **mediumorchid**
- static cv::Scalar **mediumpurple**
- static cv::Scalar **mediumseagreen**
- static cv::Scalar **mediumslateblue**
- static cv::Scalar **mediumspringgreen**
- static cv::Scalar **mediumturquoise**
- static cv::Scalar **mediumvioletred**
- static cv::Scalar **midnightblue**
- static cv::Scalar **mintcream**
- static cv::Scalar **mistyrose**
- static cv::Scalar **moccasin**
- static cv::Scalar **navajowhite**
- static cv::Scalar **navy**
- static cv::Scalar **oldlace**
- static cv::Scalar **olive**
- static cv::Scalar **olivedrab**
- static cv::Scalar **orange**
- static cv::Scalar **orangered**
- static cv::Scalar **orchid**
- static cv::Scalar **palegoldenrod**
- static cv::Scalar **palegreen**
- static cv::Scalar **paleturquoise**
- static cv::Scalar **palevioletred**
- static cv::Scalar **papayawhip**
- static cv::Scalar **peachpuff**
- static cv::Scalar **peru**
- static cv::Scalar **pink**
- static cv::Scalar **plum**
- static cv::Scalar **powderblue**
- static cv::Scalar **purple**
- static cv::Scalar **red**
- static cv::Scalar **rosybrown**
- static cv::Scalar **royalblue**
- static cv::Scalar **saddlebrown**
- static cv::Scalar **salmon**
- static cv::Scalar **sandybrown**
- static cv::Scalar **seagreen**
- static cv::Scalar **seashell**
- static cv::Scalar **sienna**
- static cv::Scalar **silver**
- static cv::Scalar **skyblue**
- static cv::Scalar **slateblue**
- static cv::Scalar **slategray**
- static cv::Scalar **slategrey**
- static cv::Scalar **snow**

- static cv::Scalar **springgreen**
- static cv::Scalar **steelblue**
- static cv::Scalar **tan**
- static cv::Scalar **teal**
- static cv::Scalar **thistle**
- static cv::Scalar **tomato**
- static cv::Scalar **turquoise**
- static cv::Scalar **violet**
- static cv::Scalar **wheat**
- static cv::Scalar **white**
- static cv::Scalar **whitesmoke**
- static cv::Scalar **yellow**
- static cv::Scalar **yellowgreen**

2.2.1 Detailed Description

Genera un conjunto de scalares que representan diferentes colores.

2.2.2 Member Function Documentation

2.2.2.1 generarColores()

```
static vector<Vec3f> Color::generarColores ( ) [inline], [static]
```

Genera un vector con varios colores.

Returns

vector<Vec3f>

2.2.2.2 scalarToVec()

```
static Vec3f Color::scalarToVec (
    Scalar sclColor ) [inline], [static]
```

Convierte un escalar a Vec3f.

Parameters

<i>sclColor</i>	
-----------------	--

Returns

Vec3f

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

- Color.h

2.3 CommonFunctions Class Reference

Realiza funciones genéricas que pueden ser utiles para cualquier proceso de este software.

```
#include <CommonFunctions.h>
```

Static Public Member Functions

- static void [info](#) (const cv::Mat &image, std::ostream &out=std::cout)
Imprime en pantalla el tamaño, cantidad de canales, y profundidad de una imagen.
- static Mat [cargarImagen](#) (string strImg, int tamano=4, int Tipo=int(IMREAD_UNCHANGED))
Carga y redimensiona una imagen en base a su ubicación.
- static vector< Mat > [cargarImagenes](#) (vector< string > strImgs, int tamano=4, int Tipo=int(IMREAD_UNCHANGED))
Carga y redimensiona un grupo de imágenes en base a sus ubicaciones.
- static vector< string > [obtenerImagenes](#) (const char *carpeta, bool verbose=false, bool reverse=false)
Obtiene las ubicaciones de un conjunto de imágenes que estan dentro de una carpeta.
- static bool [crearCarpeta](#) (String str)
Crea una carpeta.
- static string [obtenerUltimoDirectorio](#) (string &carpeta)
Obtiene el ultimo directorio (o archivo) de una ubicación y además lo borra de la cadena de texto.
- static string [obtenerUltimoDirectorio2](#) (string carpeta)
Obtiene el ultimo directorio (o archivo) de una ubicación sin borrarlo de la cadena de texto.
- static void [manejarCarpeta](#) (string carpeta)
Verifica si la ubicación de un archivo existe, en caso contrario crea las carpetas que sean necesarias.
- static void [escribirImagen](#) (string carpeta, Mat img)
Escribe una imagen, creando previamente las carpetas que sean necesarias.
- static void [showWindowNormal](#) (Mat img, String namewindow="img")
Muestra una imagen en una ventana.
- static timeval [tiempo](#) (timeval begin, string msg)
Devuelve la diferencia en segundos entre un timeval y el timeval actual, e imprime en pantalla esa diferencia.
- static Rect [rectROI](#) (Mat img)
Obtiene un rectangulo que indica la caja de límites del area de interes.
- static Mat [cropRectorROI](#) (Mat img, Rect rect)
En base a una imagen y a un rectangulo que indique el area de interes, corta la imagen.
- static Mat [boundingBox](#) (Mat img, int colindent, int rowindent)
Agrega a una imagen espacios iguales a la izquierda y derecha, y arriba y abajo.
- static Mat [boundingBox](#) (Mat img, int arlndent, int abldent, int izldent, int derldent)
Agrega a una imagen espacios arriba, abajo, izquierda y derecha.
- static Mat [copyToTransparent](#) (Mat obj, Mat scene)
Copia una imagen dentro de otra, pero manteniendo las transparencias.

- static bool [escribirPDF](#) (HPDF_Doc pdf, string str)
Escribe un pdf en una carpeta, creando las carpetas faltantes.
- static Mat [addTransparence](#) (Mat img)
Agrega transparencia a una imagen, en base a un umbral.
- static Mat [addAlpha](#) (Mat img, Mat trans)
Agrega transparencia a una imagen, en base a una mascara.
- static Mat [removeAlpha](#) (Mat img)
Quita el cuarto canal de una imagen.
- static void [histDraw](#) (Mat img, string imgName)
Dibuja el histograma de una imagen.
- static string [type2str](#) (Mat img)
- static string [obtenerFecha](#) (string img)
Obtener fecha (todo lo que esta antes del segundo _) de una imagen.
- static int [obtenerParametro](#) (string str, string parametro)
Obtener el parametro que viene despues del string parametro.
- long static int [cantPixeles](#) (Mat img)
- static string [removerExtension](#) (string str)
Quita la extension del string que se le pase.
- static bool [existFile](#) (string name)
Verifica si el archivo/carpeta existe.
- static string [stringVerde](#) (string info)
Realiza un string que en salida a consola ees de color verde.
- static string [stringAzul](#) (string proceso)
Realiza un string que en salida a consola ees de color Azul.
- static string [stringRojo](#) (string peligro)
Realiza un string que en salida a consola ees de color verde.
- static string [fToS](#) (float myFloat, int cant=4)
convierte un flotante a string
- static string [nowDate](#) ()
devuelve un string con la fecha de hoy en formato DD/MM/AAAA
- static string [nameToDate](#) (string strName)
pasa un string de un formato de fecha a otro AAAA_MMDD -> DD/MM/AAAA

2.3.1 Detailed Description

Realiza funciones genéricas que pueden ser utiles para cualquier proceso de este software.

2.3.2 Member Function Documentation

2.3.2.1 addAlpha()

```
static Mat CommonFunctions::addAlpha (
    Mat img,
    Mat trans ) [inline], [static]
```

Agrega transparencia a una imagen, en base a una mascara.

Parameters

<i>img</i>	
<i>trans</i>	

Returns

Mat

2.3.2.2 addTransparence()

```
static Mat CommonFunctions::addTransparence (
    Mat img ) [inline], [static]
```

Agrega transparencia a una imagen, en base a un umbral.

Parameters

<i>img</i>	
------------	--

Returns

Mat

2.3.2.3 boundingBox() [1/2]

```
static Mat CommonFunctions::boundingBox (
    Mat img,
    int colindent,
    int rowindent ) [inline], [static]
```

Agrega a una imagen espacios iguales a la izquierda y derecha, y arriba y abajo.

Parameters

<i>img</i>	
<i>colindent</i>	
<i>rowindent</i>	

Returns

Mat

2.3.2.4 boundingBox() [2/2]

```
static Mat CommonFunctions::boundingBox (
    Mat img,
    int arIndent,
    int abIndent,
    int izIndent,
    int derIndent ) [inline], [static]
```

Agrega a una imagen espacios arriba, abajo, izquierda y derecha.

Parameters

<i>img</i>	
<i>arIndent</i>	
<i>abIndent</i>	
<i>izIndent</i>	
<i>derIndent</i>	

Returns

Mat

2.3.2.5 cargarImagen()

```
static Mat CommonFunctions::cargarImagen (
    string strImg,
    int tamano = 4,
    int Tipo = int(IMREAD_UNCHANGED) ) [inline], [static]
```

Carga y redimensiona una imagen en base a su ubicación.

Parameters

<i>strImg</i>	
<i>tamano</i>	
<i>Tipo</i>	

Returns

Mat

2.3.2.6 cargarImagenes()

```
static vector<Mat> CommonFunctions::cargarImagenes (
    vector< string > strImgs,
```

```
int tamano = 4,  
int Tipo = int(IMREAD_UNCHANGED) ) [inline], [static]
```

Carga y redimensiona un grupo de imágenes en base a sus ubicaciones.

Parameters

<i>strImgs</i>	
<i>tamano</i>	
<i>Tipo</i>	

Returns

vector<Mat>

2.3.2.7 copyToTransparent()

```
static Mat CommonFunctions::copyToTransparent (  
    Mat obj,  
    Mat scene ) [inline], [static]
```

Copia una imagen dentro de otra, pero manteniendo las transparencias.

Parameters

<i>obj</i>	
<i>scene</i>	

Returns

Mat

2.3.2.8 crearCarpeta()

```
static bool CommonFunctions::crearCarpeta (  
    String str ) [inline], [static]
```

Crea una carpeta.

Parameters

<i>str</i>	
------------	--

Returns

true
false

2.3.2.9 cropRectorROI()

```
static Mat CommonFunctions::cropRectorROI (
    Mat img,
    Rect rect ) [inline], [static]
```

En base a una imagen y a un rectangulo que indique el area de interes, corta la imagen.

Parameters

<i>img</i>	
<i>rect</i>	

Returns

Mat

2.3.2.10 escribirImagen()

```
static void CommonFunctions::escribirImagen (
    string carpeta,
    Mat img ) [inline], [static]
```

Escribe una imagen, creando previamente las carpetas que sean necesarias.

Parameters

<i>carpeta</i>	
<i>img</i>	

2.3.2.11 escribirPDF()

```
static bool CommonFunctions::escribirPDF (
    HPDF_Doc pdf,
    string str ) [inline], [static]
```

Escribe un pdf en una carpeta, creando las carpetas faltantes.

Parameters

<i>pdf</i>	
<i>str</i>	

Returns

true
false

2.3.2.12 existFile()

```
static bool CommonFunctions::existFile (  
    string name ) [inline], [static]
```

Verifica si el archivo/carpeta existe.

Parameters

<i>name</i>	
-------------	--

Returns

true
false

2.3.2.13 fToS()

```
static string CommonFunctions::fToS (  
    float myFloat,  
    int cant = 4 ) [inline], [static]
```

convierte un flotante a string

Parameters

<i>myFloat</i>	
----------------	--

Returns

string

2.3.2.14 histDraw()

```
static void CommonFunctions::histDraw (
    Mat img,
    string imgName ) [inline], [static]
```

Dibuja el histograma de una imagen.

Parameters

<i>img</i>	
<i>imgName</i>	

Establish the number of bins

Set the ranges (for B,G,R)

Compute the histograms:

Normalize the result to [0, histImage.rows]

2.3.2.15 info()

```
static void CommonFunctions::info (
    const cv::Mat & image,
    std::ostream & out = std::cout ) [inline], [static]
```

Imprime en pantalla el tamaño, cantidad de canales, y profundidad de una imagen.

Parameters

<i>image</i>	
<i>out</i>	

2.3.2.16 manejarCarpeta()

```
static void CommonFunctions::manejarCarpeta (
    string carpeta ) [inline], [static]
```

Verifica si la ubicación de un archivo existe, en caso contrario crea las carpetas que sean necesarias.

Parameters

<i>carpeta</i>	
----------------	--

2.3.2.17 nameToDate()

```
static string CommonFunctions::nameToDate (
    string strName ) [inline], [static]
```

pasa un string de un formato de fecha a otro AAAA_MMDD -> DD/MM/AAAA

Parameters

<i>strName</i>	
----------------	--

Returns

string

2.3.2.18 nowDate()

```
static string CommonFunctions::nowDate ( ) [inline], [static]
```

devuelve un string con la fecha de hoy en formato DD/MM/AAAA

Returns

string

2.3.2.19 obtenerFecha()

```
static string CommonFunctions::obtenerFecha (
    string img ) [inline], [static]
```

Obtener fecha (todo lo que esta antes del segundo _) de una imagen.

Parameters

<i>img</i>	
------------	--

Returns

string

2.3.2.20 obtenerImagenes()

```
static vector<string> CommonFunctions::obtenerImagenes (
    const char * carpeta,
    bool verbose = false,
    bool reverse = false ) [inline], [static]
```

Obtiene las ubicaciones de un conjunto de imágenes que estan dentro de una carpeta.

Parameters

<i>carpeta</i>	
<i>reverse</i>	

Returns

vector<string>

2.3.2.21 obtenerParametro()

```
static int CommonFunctions::obtenerParametro (
    string str,
    string parametro ) [inline], [static]
```

Obtener el parametro que viene despues del string parametro.

Parameters

<i>str</i>	
<i>parametro</i>	

Returns

string

2.3.2.22 obtenerUltimoDirectorio()

```
static string CommonFunctions::obtenerUltimoDirectorio (
    string & carpeta ) [inline], [static]
```

Obtiene el ultimo directorio (o archivo) de una ubicación y además lo borra de la cadena de texto.

Parameters

<i>carpeta</i>	
----------------	--

Returns

string

2.3.2.23 obtenerUltimoDirectorio2()

```
static string CommonFunctions::obtenerUltimoDirectorio2 (  
    string carpeta ) [inline], [static]
```

Obtiene el ultimo directorio (o archivo) de una ubicación sin borrarlo de la cadena de texto.

Parameters

<i>carpeta</i>	
----------------	--

Returns

string

2.3.2.24 rectROI()

```
static Rect CommonFunctions::rectROI (  
    Mat img ) [inline], [static]
```

Obtiene un rectangulo que indica la caja de límites del area de interes.

Parameters

<i>img</i>	
------------	--

Returns

Rect

2.3.2.25 removeAlpha()

```
static Mat CommonFunctions::removeAlpha (  
    Mat img ) [inline], [static]
```

Quita el cuarto canal de una imagen.

Parameters

<i>img</i>	
------------	--

Returns

Mat

2.3.2.26 removerExtension()

```
static string CommonFunctions::removerExtension (  
    string str ) [inline], [static]
```

Quita la extension del string que se le pase.

Parameters

<i>str</i>	
------------	--

Returns

string

2.3.2.27 showWindowNormal()

```
static void CommonFunctions::showWindowNormal (  
    Mat img,  
    String namewindow = "img" ) [inline], [static]
```

Muestra una imagen en una ventana.

Parameters

<i>img</i>	
<i>namewindow</i>	

2.3.2.28 stringAzul()

```
static string CommonFunctions::stringAzul (  
    string proceso ) [inline], [static]
```

Realiza un string que en salida a consola ees de color Azul.

Parameters

<i>proceso</i>	
----------------	--

Returns

string

2.3.2.29 stringRojo()

```
static string CommonFunctions::stringRojo (  
    string peligro ) [inline], [static]
```

Realiza un string que en salida a consola ees de color verde.

Parameters

<i>peligro</i>	
----------------	--

Returns

string

2.3.2.30 stringVerde()

```
static string CommonFunctions::stringVerde (  
    string info ) [inline], [static]
```

Realiza un string que en salida a consola ees de color verde.

Parameters

<i>info</i>	
-------------	--

Returns

string

2.3.2.31 tiempo()

```
static timeval CommonFunctions::tiempo (  
    timeval begin,  
    string msg ) [inline], [static]
```

Devuelve la diferencia en segundos entre un timeval y el timeval actual, e imprime en pantalla esa diferencia.

Parameters

<i>begin</i>	
<i>msg</i>	

Returns

timeval

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

- CommonFunctions.h

2.4 FSManager Class Reference

Maneja la metadata de la camara e imágenes.

```
#include <FSManager.h>
```

Public Member Functions

- **FSManager** (string dir, string tipo, bool sobrescribir=false)
Construye un fsmanager y para esto crea el archivo de 'dir' si es que no existe.
- void **appendInt** (string nombre, int value)
Agrega un entero al fs con el nombre y valor ingresado.
- void **appendFloat** (string nombre, float value)
Agrega un flotante al fs con el nombre y valor ingresado.
- void **appendString** (string nombre, string value)
Agrega un string al fs con el nombre y valor ingresado.
- void **appendVInt** (string nombre, vector< int > value)
Agrega un vector<int> al fs con el nombre y valor ingresado.
- void **appendVFloat** (string nombre, vector< float > value)
Agrega un vector<float> al fs con el nombre y valor ingresado.
- int **readInt** (string nombre)
Lee un entero del fs con el nombre ingresado.
- float **readFloat** (string nombre)
Lee un flotante del fs con el nombre ingresado.
- string **readString** (string nombre)
Lee un string del fs con el nombre ingresado.
- vector< int > **readVInt** (string nombre)
Lee un vector de strings del fs con el nombre ingresado.
- vector< float > **readVFloat** (string nombre)
Lee un vector de floats del fs con el nombre ingresado.
- bool **existeInt** (string nombre)
Verifica si existe un entero con ese nombre en el fs.
- bool **existeFloat** (string nombre)
Verifica si existe un flotante con ese nombre en el fs.
- bool **existeString** (string nombre)
Verifica si existe un string con ese nombre en el fs.
- bool **existeVInt** (string nombre)
Verifica si existe un vector de enteros con ese nombre en el fs.
- bool **existeVFloat** (string nombre)
Verifica si existe un vector de flotantes con ese nombre en el fs.

Public Attributes

- string **dir**

2.4.1 Detailed Description

Maneja la metadata de la camara e imágenes.

2.4.2 Constructor & Destructor Documentation

2.4.2.1 FSManager()

```
FSManager::FSManager (
    string dir,
    string tipo,
    bool sobreescribir = false ) [inline]
```

Construye un fsmanager y para esto crea el archivo de 'dir' si es que no existe.

Parameters

<i>dir</i>	
<i>tipo</i>	

2.4.3 Member Function Documentation

2.4.3.1 appendFloat()

```
void FSManager::appendFloat (
    string nombre,
    float value ) [inline]
```

Agrega un flotante al fs con el nombre y valor ingresado.

Parameters

<i>nombre</i>	
<i>value</i>	

2.4.3.2 appendInt()

```
void FSManager::appendInt (
    string nombre,
    int value ) [inline]
```

Agrega un entero al fs con el nombre y valor ingresado.

Parameters

<i>nombre</i>	
<i>value</i>	

2.4.3.3 appendString()

```
void FSManager::appendString (
    string nombre,
    string value ) [inline]
```

Agrega un string al fs con el nombre y valor ingresado.

Parameters

<i>nombre</i>	
<i>value</i>	

2.4.3.4 appendVFloat()

```
void FSManager::appendVFloat (
    string nombre,
    vector< float > value ) [inline]
```

Agrega un vector<float> al fs con el nombre y valor ingresado.

Parameters

<i>nombre</i>	
<i>value</i>	

2.4.3.5 appendVInt()

```
void FSManager::appendVInt (
    string nombre,
    vector< int > value ) [inline]
```


Agrega un vector<int> al fs con el nombre y valor ingresado.

Parameters

<i>nombre</i>	
<i>value</i>	

2.4.3.6 existeFloat()

```
bool FSManager::existeFloat (
    string nombre ) [inline]
```

Verifica si existe un flotante con ese nombre en el fs.

Parameters

<i>nombre</i>	
---------------	--

Returns

true
false

2.4.3.7 existeInt()

```
bool FSManager::existeInt (
    string nombre ) [inline]
```

Verifica si existe un entero con ese nombre en el fs.

Parameters

<i>nombre</i>	
---------------	--

Returns

true
false

2.4.3.8 existeString()

```
bool FSManager::existeString (
    string nombre ) [inline]
```

Verifica si existe un string con ese nombre en el fs.

Parameters

<i>nombre</i>	
---------------	--

Returns

true
false

2.4.3.9 existeVFloat()

```
bool FSManager::existeVFloat (
    string nombre ) [inline]
```

Verifica si existe un vector de flotantes con ese nombre en el fs.

Parameters

<i>nombre</i>	
---------------	--

Returns

true
false

2.4.3.10 existeVInt()

```
bool FSManager::existeVInt (
    string nombre ) [inline]
```

Verifica si existe un vector de enteros con ese nombre en el fs.

Parameters

<i>nombre</i>	
---------------	--

Returns

true
false

2.4.3.11 readFloat()

```
float FSManager::readFloat (
    string nombre ) [inline]
```

Lee un flotante del fs con el nombre ingresado.

Parameters

<i>nombre</i>	
---------------	--

Returns

int

2.4.3.12 readInt()

```
int FSManager::readInt (
    string nombre ) [inline]
```

Lee un entero del fs con el nombre ingresado.

Parameters

<i>nombre</i>	
---------------	--

Returns

int

2.4.3.13 readString()

```
string FSManager::readString (
    string nombre ) [inline]
```

Lee un string del fs con el nombre ingresado.

Parameters

<i>nombre</i>	
---------------	--

Returns

int

2.4.3.14 readVFloat()

```
vector<float> FSManager::readVFloat (
    string nombre ) [inline]
```

Lee un vector de floats del fs con el nombre ingresado.

Parameters

<i>nombre</i>	
---------------	--

Returns

vector<float>

2.4.3.15 readVInt()

```
vector<int> FSManager::readVInt (
    string nombre ) [inline]
```

Lee un vector de strings del fs con el nombre ingresado.

Parameters

<i>nombre</i>	
---------------	--

Returns

vector<int>

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

- FSManager.h

2.5 IndexCalculation Class Reference

Calcula los indices de vegetación.

```
#include <IndexCalculation.h>
```

Public Member Functions

- **IndexCalculation** (bool outputStitching=false, bool parallel=true, bool sobrecribir=true)
- void **processManager** ()
Maneja todo el proceso para generación de los índices.
- void **indexCalcu** (string strImg, bool multispectral)
Decide que índices se van a calcular en base al tipo de imágenes que se procesen.
- void **indexCalcuRGB** (string strImg)
Calcula los índices para imágenes RGB.
- void **rgCalculation** (vector< Mat > BGRA, string strImg)
Calcula y escribe el índice RG=Rojo/Verde.
- void **ngrdiCalculation** (vector< Mat > BGRA, string strImg)
Calcula y escribe el índice NGRDI.
- void **indexCalcuMS** (string strImg)
Calcula los índices para imágenes multi-espectrales.
- void **ndviCalculation** (vector< Mat > BGRA, string strImg)
Calcula y escribe el índice NDVI=(Infrarojo-Rojo)/(Infrarojo+Rojo).
- void **rviCalculation** (vector< Mat > BGRA, string strImg)
Calcula y escribe el índice RVI=Infrarojo/Rojo.
- void **escribirSegmentaciones** (Mat indice, Mat trans, string strImg, string strIndex)
Escribe los resultados de un índice con diferentes segmentaciones.
- vector< string > **obtenerInput** (bool multiespectral, bool outputStitching)
Obtiene las ubicaciones de las imágenes de entrada en base al tipo de imagen y a la BD que la contiene.
- vector< string > **obtenerRGBInput** ()
Obtiene las ubicaciones de las imágenes RGB de entrada.
- string **obtenerRGBInputStr** ()
- vector< string > **obtenerMSInput** ()
Obtiene las ubicaciones de las imágenes multi-espectrales de entrada.
- string **obtenerMSInputStr** ()
- vector< string > **obtenerRGBOutput** ()
Obtiene las ubicaciones de las imágenes RGB de salida.
- vector< string > **obtenerMSOutput** ()
Obtiene las ubicaciones de las imágenes multi-espectrales de salida.
- string **obtenerIndex** (string strImg, string strIndex)
- string **obtenerCuant** (string strImg, string strIndex)
- string **obtenerChart** (string strImg, string strIndex)
- string **obtenerChartImg** (string strImg, string strIndex)
- string **obtenerLut** (string strImg, string strIndex)

Public Attributes

- bool **outputStitching**
- bool **parallel**
- bool **multispectral**
- bool **sobrecribir**

2.5.1 Detailed Description

Calcula los índices de vegetación.

2.5.2 Member Function Documentation

2.5.2.1 escribirSegmentaciones()

```
void IndexCalculation::escribirSegmentaciones (
    Mat indice,
    Mat trans,
    string strImg,
    string strIndex ) [inline]
```

Escribe los resultados de un indice con diferentes segmentaciones.

Parameters

<i>indice</i>	
<i>trans</i>	
<i>strImg</i>	
<i>strIndex</i>	

2.5.2.2 indexCalcu()

```
void IndexCalculation::indexCalcu (
    string strImg,
    bool multispectral ) [inline]
```

Decide que indices se van a calcular en base al tipo de imágenes que se procesen.

Parameters

<i>strImg</i>	
<i>multispectral</i>	

2.5.2.3 indexCalcuMS()

```
void IndexCalculation::indexCalcuMS (
    string strImg ) [inline]
```

Calcula los indices para imágenes multi-espectrales.

Parameters

<i>strImg</i>	
---------------	--

2.5.2.4 indexCalcuRGB()

```
void IndexCalculation::indexCalcuRGB (
    string strImg ) [inline]
```

Calcula los indices para imágenes RGB.

Parameters

<i>strImg</i>	
---------------	--

2.5.2.5 ndviCalculation()

```
void IndexCalculation::ndviCalculation (
    vector< Mat > BGRA,
    string strImg ) [inline]
```

Calcula y escribe el indice NDVI=(Infrarojo-Rojo)/(Infrarojo+Rojo).

Parameters

<i>BGRA</i>	
<i>strImg</i>	

2.5.2.6 ngrdiCalculation()

```
void IndexCalculation::ngrdiCalculation (
    vector< Mat > BGRA,
    string strImg ) [inline]
```

Calcula y escribe el indice NGRDI.

Parameters

<i>BGRA</i>	
<i>strImg</i>	

2.5.2.7 obtenerInput()

```
vector<string> IndexCalculation::obtenerInput (
```



```
bool multispectral,  
bool outputStitching ) [inline]
```

Obtiene las ubicaciones de las imágenes de entrada en base al tipo de imagen y a la BD que la contiene.

Parameters

<i>multispectral</i>	
<i>outputStitching</i>	

Returns

vector<string>

2.5.2.8 obtenerMSInput()

```
vector<string> IndexCalculation::obtenerMSInput ( ) [inline]
```

Obtiene las ubicaciones de las imágenes multi-espectrales de entrada.

Returns

vector<string>

2.5.2.9 obtenerMSOutput()

```
vector<string> IndexCalculation::obtenerMSOutput ( ) [inline]
```

Obtiene las ubicaciones de las imágenes multi-espectrales de salida.

Returns

vector<string>

2.5.2.10 obtenerRGBInput()

```
vector<string> IndexCalculation::obtenerRGBInput ( ) [inline]
```

Obtiene las ubicaciones de las imágenes RGB de entrada.

Returns

vector<string>

2.5.2.11 obtenerRGBOutput()

```
vector<string> IndexCalculation::obtenerRGBOutput ( ) [inline]
```

Obtiene las ubicaciones de las imágenes RGB de salida.

Returns

vector<string>

2.5.2.12 rgCalculation()

```
void IndexCalculation::rgCalculation (
    vector< Mat > BGRA,
    string strImg ) [inline]
```

Calcula y escribe el índice RG=Rojo/Verde.

Parameters

<i>BGRA</i>	
<i>strImg</i>	

2.5.2.13 rviCalculation()

```
void IndexCalculation::rviCalculation (
    vector< Mat > BGRA,
    string strImg ) [inline]
```

Calcula y escribe el índice RVI=Infrarojo/Rojo.

Parameters

<i>BGRA</i>	
<i>strImg</i>	

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

- IndexCalculation.h

2.6 Performance Class Reference

Calcula los índices de desempeño.

```
#include <Performance.h>
```

2.6.1 Detailed Description

Calcula los indices de desempeño.

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

- Performance.h

2.7 Redaction Class Reference

Redacta el PDF final.

```
#include <Redaction.h>
```

Public Member Functions

- int [generatePDF](#) ()
Crea un PDF con todas las imágenes.
- void [writePortada](#) (HPDF_Doc pdf, string strName)
- void [writeOriginal](#) (string strImgName, HPDF_Doc pdf)
Se escribe el ortomosaico y un texto que lo acompaña.
- void [writeIndex](#) (string strImgName, string strIndex, HPDF_Doc pdf)
Se escribe el indice y un texto que lo acompaña.
- void [writeCuantizado](#) (string strImgName, string strIndex, HPDF_Doc pdf)
- void [writeLut](#) (string strImgName, string strIndex, HPDF_Doc pdf)
Se escribe el mapa de colores y un texto que lo acompaña.
- void [writeChart](#) (string strImgName, string strIndex, HPDF_Doc pdf)
Se escribe el gráfico de porcentajes y un texto que lo acompaña.
- string [generateChartText](#) (string strImgName, string strIndex)
Genera el texto para el gráfico, utilizando metadatos manejador por [FSManager](#).
- void [writeText](#) (string text, HPDF_Doc pdf, HPDF_Page page_1, int sizeFont, int left, int top, int right, int bottom, HPDF_TextAlignment textAlign=HPDF_TALIGN_JUSTIFY)
Escribe un texto en el rectangulo indicado por las coordenadas que se pasan por parametros.
- int [writeImg](#) (HPDF_Doc pdf, HPDF_Page page, const char *file, int maxWidth=600, int maxHeight=840, int xSpace=50, int ySpace=50)
Escribe una imagen dentro de un pdf, con su relación ancho/largo mantenida pero redimensionada.
- int [writeImg2](#) (HPDF_Doc pdf, HPDF_Page page, const char *file, int maxWidth=600, int maxHeight=840, int xSpace=50, int ySpace=50)
- void [writeContext](#) (string strText, HPDF_Doc pdf, HPDF_Page page_1, bool writeTitle=true)
- void [draw_line](#) (HPDF_Page page, float x0, float x1, float y0, float y1)

Public Attributes

- int **altura**
- int **page** = 0
- int **x0** = 80
- int **x1** = 520
- int **lineSpacing** = 17
- string **fonts** = "cmunbsr.ttf"
- vector< int > **sizeImg**

2.7.1 Detailed Description

Redacta el PDF final.

2.7.2 Member Function Documentation

2.7.2.1 generateChartText()

```
string Redaction::generateChartText (
    string strImgName,
    string strIndex ) [inline]
```

Genera el texto para el gráfico, utilizando metadatos manejador por [FSManager](#).

Parameters

<i>strImgName</i>	
<i>strIndex</i>	

Returns

string

2.7.2.2 generatePDF()

```
int Redaction::generatePDF ( ) [inline]
```

Crea un PDF con todas las imágenes.

Returns

int

2.7.2.3 writeChart()

```
void Redaction::writeChart (
    string strImgName,
    string strIndex,
    HPDF_Doc pdf ) [inline]
```

Se escribe el gráfico de porcentajes y un texto que lo acompaña.

Parameters

<i>strImgName</i>	
<i>strIndex</i>	
<i>pdf</i>	

2.7.2.4 writeImg()

```
int Redaction::writeImg (
    HPDF_Doc pdf,
    HPDF_Page page,
    const char * file,
    int maxWidth = 600,
    int maxHeight = 840,
    int xSpace = 50,
    int ySpace = 50 ) [inline]
```

Escribe una imagen dentro de un pdf, con su relación ancho/largo mantenida pero redimensionada.

Parameters

<i>pdf</i>	
<i>page</i>	
<i>file</i>	
<i>maxWidth</i>	
<i>maxHeight</i>	
<i>xSpace</i>	
<i>ySpace</i>	

Returns

int

2.7.2.5 writeIndex()

```
void Redaction::writeIndex (
    string strImgName,
    string strIndex,
    HPDF_Doc pdf ) [inline]
```

Se escribe el índice y un texto que lo acompaña.

Parameters

<i>strImgName</i>	
<i>strIndex</i>	
<i>pdf</i>	

2.7.2.6 writeLut()

```
void Redaction::writeLut (
    string strImgName,
    string strIndex,
    HPDF_Doc pdf ) [inline]
```

Se escribe el mapa de colores y un texto que lo acompaña.

Parameters

<i>strImgName</i>	
<i>strIndex</i>	
<i>pdf</i>	

2.7.2.7 writeOriginal()

```
void Redaction::writeOriginal (
    string strImgName,
    HPDF_Doc pdf ) [inline]
```

Se escribe el ortomosaico y un texto que lo acompaña.

Parameters

<i>strImgName</i>	
<i>pdf</i>	

2.7.2.8 writeText()

```
void Redaction::writeText (
    string text,
    HPDF_Doc pdf,
    HPDF_Page page_1,
    int sizeFont,
    int left,
    int top,
    int right,
    int bottom,
    HPDF_TextAlignment textAlign = HPDF_TALIGN_JUSTIFY ) [inline]
```

Escribe un texto en el rectangulo indicado por las coordenadas que se pasan por parametros.

Parameters

<i>text</i>	
<i>pdf</i>	
<i>page_1</i>	
<i>sizeFont</i>	
<i>left</i>	
<i>top</i>	
<i>right</i>	
<i>bottom</i>	
<i>textAlign</i>	

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

- Redaction.h

2.8 Segmentation Class Reference

Genera las diferentes segmentaciones.

```
#include <Segmentation.h>
```

Static Public Member Functions

- static Mat [segmentationVariation](#) (Mat img, Mat trans, int cantColores)
Realiza una normalizacion de la imagen (gris) y luego cuantiza sus valores para reducir las escalas de grises.
- static Mat [createLut](#) (Mat temp, Mat trans)
Crea y aplica un mapa de colores a una imagen y luego genera y pega un indice para que se pueda ver que valor tiene cada color en una imagen.
- static Mat [drawIndexOfIndex](#) (Mat img, Mat Lut, Mat trans, double min, double max)
Genera y pega un indice para que se pueda ver que valor tiene cada color en una imagen.
- static Mat [normalizateMat](#) (Mat img, Mat mask, double &min, double &max)
Normaliza los pixeles de una imagen.
- static vector< Mat > [generarGrafico](#) (Mat img, int cantidad, Mat trans, string strImg, string strIndex)
Aplica una cuantización de una imagen y a cada cuantizado le asigna su porcentaje de aparición en la imagen, con esto se realiza un gráfico.
- static Mat [separarSuelo](#) (vector< Mat > BGRA)
Separa el suelo de la vegetación.

Public Attributes

- double **min** =0
- double **max** =0

2.8.1 Detailed Description

Genera las diferentes segmentaciones.

2.8.2 Member Function Documentation

2.8.2.1 createLut()

```
static Mat Segmentation::createLut (  
    Mat temp,  
    Mat trans ) [inline], [static]
```

Crea y aplica un mapa de colores a una imagen y luego genera y pega un indice para que se pueda ver que valor tiene cada color en una imagen.

Parameters

<i>temp</i>	
<i>trans</i>	

Returns

Mat

2.8.2.2 drawIndexOfIndex()

```
static Mat Segmentation::drawIndexOfIndex (  
    Mat img,  
    Mat Lut,  
    Mat trans,  
    double min,  
    double max ) [inline], [static]
```

Genera y pega un indice para que se pueda ver que valor tiene cada color en una imagen.

Parameters

<i>img</i>	
<i>Lut</i>	
<i>trans</i>	
<i>min</i>	
<i>max</i>	

Returns

Mat

2.8.2.3 generarGrafico()

```
static vector<Mat> Segmentation::generarGrafico (
    Mat img,
    int cantidad,
    Mat trans,
    string strImg,
    string strIndex ) [inline], [static]
```

Aplica una cuantización de una imagen y a cada cuantizado le asigna su porcentaje de aparición en la imagen, con esto se realiza un gráfico.

Parameters

<i>img</i>	
<i>cantidad</i>	
<i>trans</i>	

Returns

vector<Mat>

2.8.2.4 normalizeMat()

```
static Mat Segmentation::normalizeMat (
    Mat img,
    Mat mask,
    double & min,
    double & max ) [inline], [static]
```

Normaliza los pixeles de una imagen.

Parameters

<i>img</i>	
<i>mask</i>	
<i>min</i>	
<i>max</i>	

Returns

Mat

2.8.2.5 segmentationVariation()

```
static Mat Segmentation::segmentationVariation (
    Mat img,
```

```
Mat trans,
int cantColores ) [inline], [static]
```

Realiza una normalizacion de la imagen (gris) y luego cuantiza sus valores para reducir las escalas de grises.

Parameters

<i>img</i>	
<i>trans</i>	
<i>cantColores</i>	

Returns

Mat

2.8.2.6 separarSuelo()

```
static Mat Segmentation::separarSuelo (
    vector< Mat > BGRA ) [inline], [static]
```

Separa el suelo de la vegetación.

Parameters

<i>BGRA</i>	
-------------	--

Returns

Mat

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

- Segmentation.h

2.9 uav::Stitcher Class Reference

Realiza el pegado de un conjunto de imágenes.

```
#include <Stitcher.h>
```

Public Member Functions

- **Stitcher** (int [tamano](#)=4, int [minKeypoints](#)=5000, bool [originalSize](#)=false, bool multispectral=true, int [altura](#)=120)
- vector< Mat > [stitchWarp](#) (Mat scene, Mat obj, Mat homoMatrix)

funcion para pegar una imagen transformada por una homografia en otra imagen. En el caso de q tenga 4 canales (o sea el cuarto sea alpha [transparente]) hace un pegado especial para que no se pierda la transparencia, y en el caso contrario la pega de una manera q no se note el paso de una imagen a otra
- Mat [copyToTransparent](#) (Mat obj, Mat scene, Mat mask)

Aplica un blending especial donde, en base a una mascara, decide que valor van a aportar los pixeles del objeto y la escena, en la escena final.
- double [compareMats](#) (int numHomo, Mat homoMatrix)

Compara una imagen(escena) con la consecuente(objeto) transformada.
- void [detectAndDescript](#) ()

Obtiene los keypoints y descriptores de cada imagen y los escribe en vecKp y vecDesc.
- void [matchKp](#) ()

Realiza emparejamientos entre los keypoints de 2 imagenes, en base a sus descriptores(vecDesc), y escribe los resultados en vecMatch.
- Mat [rigidToHomography](#) (Mat R)

convierte una matriz de transformada rigida a una homografia
- vector< DMatch > [removeOutliers](#) (vector< DMatch > gm, int numHomo, int numHomo2)

Elimina emparejamientos erroneos por medio de la media y varianza de las posiciones en X e Y de los keypoints con y sin desplazamientos. Los emparejamientos con desplazamiento muy lejos de la media, se consideran erroneos.
- void [getHomography](#) (int numHomo)

Obtengo varias homografias modificando ciertos parametros, elijo la que sea mas adecuada y la escribo en homo↔NoMultiplied.
- void [getHomographies](#) ()

Obtengo las homografias entre cada par de imágenes consecutivas, las multiplico para adaptar las transformaciones al espacio que se use como marco de referencia y las guardo en el vector H.
- void [findBoundingBoxLimits](#) ()

En base a las homografias, se obtienen los valores que van a delimitar al bound box.
- bool [evaluateHomography](#) ()

Evalua que los limites del boundingbox no sean de una homografia mal calculada.
- void [generateBoundingBox](#) ()

Usando los boundingboxlimits obtenidos en la funcion findBoundingBoxLimits, se genera el boundingbox.
- void [rescaleHomographies](#) ()

En el caso de que las homografias se hayan calculado en base a imagenes a las cuales se les cambio el tamaño para que sea mas rapido el procesamiento, se les modifica la homografia para adaptarlas a su tamaño.
- void [eraseFromVectors](#) ()

Elimina las imágenes que ya hayan sido pegadas.
- Mat [stitchImgs](#) ()

En base a las homografias realiza el pegado de las imagenes.
- void [removeCorners](#) ()

quito las esquinas para remover el vignetting
- void [compensateBright](#) ()

Hago que todas las imágenes tengan la misma media en cada canal.
- Mat [runAll](#) ()

Utilizo todas las funciones anteriores para realizar el stitching, siguiendo el siguiente proceso:
- bool [processManager](#) ()

Realiza todo el proceso para pegar las imágenes.
- bool [existResult](#) ()
- void [saveMetadata](#) (bool sobrecribir)

Guarda tamaño y altura a la metadata de la imagen.

- vector< string > **obtenerInput** ()
Obtiene las ubicaciones las imágenes a pegar.
- vector< string > **obtenerInputOrto** ()
Obtiene las ubicaciones de los resultados intermedios, para pegarlos y generar el resultado final.
- string **obtenerOutputOrto** (int num)
Obtiene la ubicación donde se escribirán los resultados intermedios.
- string **obtenerOutputRF** ()
Obtiene la ubicación donde se escribirá el resultado final.
- void **escribirOutput** (int numRes)
Escribe los resultados intermedios o final dependiendo de finalResult.
- void **escribirOutputRF** ()
Escribe resultado final.
- void **escribirOutputOrto** (int numRes)
Escribe resultados intermedios.

Public Attributes

- int **altura**
Altura a la que se obtuvieron las fotos.
- int **tamano**
Tamaño con el que se redimensionan las imágenes.
- bool **originalSize** =false
Booleano que decide si recuperar tamaño original en el resultado final.
- bool **multispectral** =true
- double **yMin** =0
Limite superior de la caja de limites.
- double **yMax** =0
Limite inferior de la caja de limites.
- double **xMin** =0
Limite izquierdo de la caja de limites.
- double **xMax** =0
Limite derecho de la caja de limites.
- int **minKeypoints**
Cantidad minima de keypoints a calcular.
- bool **finalResult**
Booleano que decide si se esta procesando los resultados intermedios o el resultado final.
- string **resultName**
Nombre del resultado.
- vector< Mat > **imgs**
Imágenes a pegar.
- vector< Mat > **vecDesc**
Descriptores de las imágenes.
- vector< Mat > **H**
Transformaciones entre cada par de imágenes adaptadas al marco de referencia inicial.
- vector< Mat > **homoNoMultiplicated**
Transformaciones entre cada par de imágenes no adaptadas.
- vector< string > **strImgs**
Ubicación de cada imagen.
- vector< vector< KeyPoint > > **vecKp**
Puntos claves de las imágenes.

- `vector< vector< DMatch > > vecMatch`
Emparejamiento entre cada punto clave de imágenes consecutivas.
- `vector< vector< DMatch > > best_inliers`
Emparejamientos correctos.
- `Mat boundBox`
Caja de limites.

2.9.1 Detailed Description

Realiza el pegado de un conjunto de imágenes.

2.9.2 Member Function Documentation

2.9.2.1 `compareMats()`

```
double uav::Stitcher::compareMats (
    int numHomo,
    Mat homoMatrix ) [inline]
```

Compara una imagen(escena) con la consecuente(objeto) transformada.

Parameters

<i>numHomo</i>	
<i>homoMatrix</i>	

Returns

double

2.9.2.2 `copyToTransparent()`

```
Mat uav::Stitcher::copyToTransparent (
    Mat obj,
    Mat scene,
    Mat mask ) [inline]
```

Aplica un blending especial donde, en base a una mascara, decide que valor van a aportar los pixeles del objeto y la escena, en la escena final.

Parameters

<i>obj</i>	
<i>scene</i>	
<i>mask</i>	

Returns

Mat

2.9.2.3 escribirOutput()

```
void uav::Stitcher::escribirOutput (
    int numRes ) [inline]
```

Escribe los resultados intermedios o final dependiendo de finalResult.

Parameters

<i>numRes</i>	
---------------	--

2.9.2.4 escribirOutputOrto()

```
void uav::Stitcher::escribirOutputOrto (
    int numRes ) [inline]
```

Escribe resultados intermedios.

Parameters

<i>numRes</i>	
---------------	--

2.9.2.5 evaluateHomography()

```
bool uav::Stitcher::evaluateHomography ( ) [inline]
```

Evalua que los limites del bbox no sean de una homografia mal calculada.

Returns

true
false

2.9.2.6 getHomography()

```
void uav::Stitcher::getHomography (
    int numHomo ) [inline]
```

Obtengo varias homografias modificando ciertos parametros, elijo la que sea mas adecuada y la escribo en homo↔
NoMultiplicated.

Parameters

<i>numHomo</i>	
----------------	--

2.9.2.7 obtenerInput()

```
vector<string> uav::Stitcher::obtenerInput ( ) [inline]
```

Obtiene las ubicaciones las imágenes a pegar.

Returns

vector<string>

2.9.2.8 obtenerInputOrto()

```
vector<string> uav::Stitcher::obtenerInputOrto ( ) [inline]
```

Obtiene las ubicaciones de los resultados intermedios, para pegarlos y generar el resultado final.

Returns

vector<string>

2.9.2.9 obtenerOutputOrto()

```
string uav::Stitcher::obtenerOutputOrto (
    int num ) [inline]
```

Obtiene la ubicación donde se escribieran los resultados intermedios.

Parameters

<i>num</i>	
------------	--

Returns

string

2.9.2.10 obtenerOutputRF()

```
string uav::Stitcher::obtenerOutputRF ( ) [inline]
```

Obtiene la ubicación donde se escribirá el resultado final.

Returns

string

2.9.2.11 runAll()

```
Mat uav::Stitcher::runAll ( ) [inline]
```

Utilizo todas las funciones anteriores para realizar el stitching, siguiendo el siguiente proceso:

- Quito esquinas.
- Obtengo keypoints y descriptores.
- Realizo emparejamiento.
- Obtengo homografías.
- Genero caja de límites.
- Adapto homografías al tamaño original.
- Pego las imágenes.

2.9.2.12 stitchImgs()

```
Mat uav::Stitcher::stitchImgs ( ) [inline]
```

En base a las homografías realiza el pegado de las imágenes.

Returns

Mat

2.9.2.13 stitchWarp()

```
vector<Mat> uav::Stitcher::stitchWarp (
    Mat scene,
    Mat obj,
    Mat homoMatrix ) [inline]
```

funcion para pegar una imagen transformada por una homografía en otra imagen. En el caso de q tenga 4 canales (o sea el cuarto sea alpha [transparente]) hace un pegado especial para que no se pierda la transparencia, y en el caso contrario la pega de una manera q no se note el paso de una imagen a otra

Parameters

<i>scene</i>	asd
<i>obj</i>	
<i>homoMatrix</i>	

Returns

vector<Mat>

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

- `Stitcher.h`

2.10 Undistort Class Reference

Clase utilizada para quitar la distorsión de un conjunto de imágenes.

```
#include <Undistort.h>
```

Static Public Member Functions

- static void `undistortImgs` (string cameraName)
Quita la distorsión de un conjunto de imágenes utilizando una matriz de transformación.
- static vector< string > `obtenerInput` ()
Obtiene la ubicación de las imágenes a las que se le quitará la distorsión.
- static void `escribirOutput` (Mat frameUndistorted, string strImg)
Escribe las imágenes sin distorsión.

2.10.1 Detailed Description

Clase utilizada para quitar la distorsión de un conjunto de imágenes.

2.10.2 Member Function Documentation

2.10.2.1 `escribirOutput()`

```
static void Undistort::escribirOutput (
    Mat frameUndistorted,
    string strImg ) [inline], [static]
```

Escribe las imágenes sin distorsión.

Parameters

<i>frameUndistorted</i>	
<i>strImg</i>	

2.10.2.2 obtenerInput()

```
static vector<string> Undistort::obtenerInput ( ) [inline], [static]
```

Obtiene la ubicación de las imágenes a las que se le quitará la distorsión.

Returns

vector<string>

2.10.2.3 undistortImgs()

```
static void Undistort::undistortImgs (
    string cameraName ) [inline], [static]
```

Quita la distorsión de un conjunto de imágenes utilizando una matriz de transformación.

Parameters

<i>cameraName</i>	
-------------------	--

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

- Undistort.h

2.11 UtilInformation Class Reference

Calcula los indices de desempeño.

```
#include <UtilInformation.h>
```

Public Member Functions

- void [selectCamera](#) ()
Pide un nombre de camara y verifica si existe para leerla o crearla.
- void [writeCameraProperties](#) ()

- Pide que se entre por command-line algunos parametros y los guarda en un yml.*
 - void [readCameraProperties](#) ()
Lee los parametros escritos en un yml.
 - void [calcularHectareas](#) ()
Calcula las hectareas que tiene una imagen.
 - void [dibujarLineasMetro](#) (int cantMetros=50)
Dibuja lineas cada cantMetros.
 - void [dibujarBGR](#) ()
separa las canalesw BGR, los separa y hace la substraccion para corregir el problema de NIR

Public Attributes

- string **cameraName**
- float **width**
- float **height**
- float **focal**
- float **mmpx**

2.11.1 Detailed Description

Calcula los indices de desempeño.

2.11.2 Member Function Documentation

2.11.2.1 [calcularHectareas\(\)](#)

```
void UtilInformation::calcularHectareas ( ) [inline]
```

Calcula las hectareas que tiene una imagen.

Returns

float

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

- UtilInformation.h

Index

- addAlpha
 - CommonFunctions, [10](#)
- addTransparence
 - CommonFunctions, [11](#)
- appendFloat
 - FSManager, [23](#)
- appendInt
 - FSManager, [23](#)
- appendString
 - FSManager, [24](#)
- appendVFloat
 - FSManager, [24](#)
- appendVInt
 - FSManager, [24](#)
- boundingBox
 - CommonFunctions, [11](#)
- calcularHectareas
 - UtilInformation, [51](#)
- calibratImg
 - Calibration, [3](#)
- Calibration, [3](#)
 - calibratImg, [3](#)
 - obtenerInput, [4](#)
 - readCalibrationMat, [4](#)
 - storeCalibrationMat, [4](#)
- cargarImagen
 - CommonFunctions, [12](#)
- cargarImagenes
 - CommonFunctions, [12](#)
- Color, [5](#)
 - generarColores, [8](#)
 - scalarToVec, [8](#)
- CommonFunctions, [9](#)
 - addAlpha, [10](#)
 - addTransparence, [11](#)
 - boundingBox, [11](#)
 - cargarImagen, [12](#)
 - cargarImagenes, [12](#)
 - copyToTransparent, [13](#)
 - crearCarpeta, [13](#)
 - cropRectorROI, [14](#)
 - escribirImagen, [14](#)
 - escribirPDF, [14](#)
 - existFile, [15](#)
 - fToS, [15](#)
 - histDraw, [15](#)
 - info, [16](#)
 - manejarCarpeta, [16](#)
 - nameToDate, [16](#)
 - nowDate, [17](#)
 - obtenerFecha, [17](#)
 - obtenerImagenes, [17](#)
 - obtenerParametro, [18](#)
 - obtenerUltimoDirectorio, [18](#)
 - obtenerUltimoDirectorio2, [19](#)
 - rectROI, [19](#)
 - removeAlpha, [19](#)
 - removerExtension, [20](#)
 - showWindowNormal, [20](#)
 - stringAzul, [20](#)
 - stringRojo, [21](#)
 - stringVerde, [21](#)
 - tiempo, [21](#)
- compareMats
 - uav::Stitcher, [45](#)
- copyToTransparent
 - CommonFunctions, [13](#)
 - uav::Stitcher, [45](#)
- crearCarpeta
 - CommonFunctions, [13](#)
- createLut
 - Segmentation, [40](#)
- cropRectorROI
 - CommonFunctions, [14](#)
- drawIndexOfIndex
 - Segmentation, [40](#)
- escribirImagen
 - CommonFunctions, [14](#)
- escribirOutput
 - uav::Stitcher, [46](#)
 - Undistort, [49](#)
- escribirOutputOrto
 - uav::Stitcher, [46](#)
- escribirPDF
 - CommonFunctions, [14](#)
- escribirSegmentaciones
 - IndexCalculation, [31](#)
- evaluateHomography
 - uav::Stitcher, [46](#)
- existFile
 - CommonFunctions, [15](#)
- existeFloat
 - FSManager, [25](#)
- existeInt
 - FSManager, [25](#)
- existeString

- FSManager, 25
- existeVFloat
 - FSManager, 27
- existeVInt
 - FSManager, 27
- FSManager, 22
 - appendFloat, 23
 - appendInt, 23
 - appendString, 24
 - appendVFloat, 24
 - appendVInt, 24
 - existeFloat, 25
 - existeInt, 25
 - existeString, 25
 - existeVFloat, 27
 - existeVInt, 27
 - FSManager, 23
 - readFloat, 27
 - readInt, 28
 - readString, 28
 - readVFloat, 29
 - readVInt, 29
- fToS
 - CommonFunctions, 15
- generarColores
 - Color, 8
- generarGrafico
 - Segmentation, 40
- generateChartText
 - Redaction, 36
- generatePDF
 - Redaction, 36
- getHomography
 - uav::Stitcher, 46
- histDraw
 - CommonFunctions, 15
- indexCalcu
 - IndexCalculation, 31
- indexCalcuMS
 - IndexCalculation, 31
- indexCalcuRGB
 - IndexCalculation, 32
- IndexCalculation, 29
 - escribirSegmentaciones, 31
 - indexCalcu, 31
 - indexCalcuMS, 31
 - indexCalcuRGB, 32
 - ndviCalculation, 32
 - ngrdiCalculation, 32
 - obtenerInput, 32
 - obtenerMSInput, 33
 - obtenerMSOutput, 33
 - obtenerRGBInput, 33
 - obtenerRGBOutput, 33
 - rgCalculation, 34
 - rviCalculation, 34
- info
 - CommonFunctions, 16
- manejarCarpeta
 - CommonFunctions, 16
- nameToDate
 - CommonFunctions, 16
- ndviCalculation
 - IndexCalculation, 32
- ngrdiCalculation
 - IndexCalculation, 32
- normalizateMat
 - Segmentation, 41
- nowDate
 - CommonFunctions, 17
- obtenerFecha
 - CommonFunctions, 17
- obtenerImagenes
 - CommonFunctions, 17
- obtenerInput
 - Calibration, 4
 - IndexCalculation, 32
 - uav::Stitcher, 47
 - Undistort, 50
- obtenerInputOrto
 - uav::Stitcher, 47
- obtenerMSInput
 - IndexCalculation, 33
- obtenerMSOutput
 - IndexCalculation, 33
- obtenerOutputOrto
 - uav::Stitcher, 47
- obtenerOutputRF
 - uav::Stitcher, 47
- obtenerParametro
 - CommonFunctions, 18
- obtenerRGBInput
 - IndexCalculation, 33
- obtenerRGBOutput
 - IndexCalculation, 33
- obtenerUltimoDirectorio
 - CommonFunctions, 18
- obtenerUltimoDirectorio2
 - CommonFunctions, 19
- Performance, 34
- readCalibrationMat
 - Calibration, 4
- readFloat
 - FSManager, 27
- readInt
 - FSManager, 28
- readString
 - FSManager, 28
- readVFloat

- FSManager, 29
- readVInt
 - FSManager, 29
- rectROI
 - CommonFunctions, 19
- Redaction, 35
 - generateChartText, 36
 - generatePDF, 36
 - writeChart, 36
 - writelnImg, 37
 - writelnIndex, 37
 - writelnLut, 38
 - writelnOriginal, 38
 - writelnText, 38
- removeAlpha
 - CommonFunctions, 19
- removerExtension
 - CommonFunctions, 20
- rgCalculation
 - IndexCalculation, 34
- runAll
 - uav::Stitcher, 48
- rviCalculation
 - IndexCalculation, 34
- scalarToVec
 - Color, 8
- Segmentation, 39
 - createLut, 40
 - drawIndexOfIndex, 40
 - generarGrafico, 40
 - normalizateMat, 41
 - segmentationVariation, 41
 - separarSuelo, 42
- segmentationVariation
 - Segmentation, 41
- separarSuelo
 - Segmentation, 42
- showWindowNormal
 - CommonFunctions, 20
- stitchImgs
 - uav::Stitcher, 48
- stitchWarp
 - uav::Stitcher, 48
- storeCalibrationMat
 - Calibration, 4
- stringAzul
 - CommonFunctions, 20
- stringRojo
 - CommonFunctions, 21
- stringVerde
 - CommonFunctions, 21
- tiempo
 - CommonFunctions, 21
- uav::Stitcher, 42
 - compareMats, 45
 - copyToTransparent, 45
 - escribirOutput, 46
 - escribirOutputOrto, 46
 - evaluateHomography, 46
 - getHomography, 46
 - obtenerInput, 47
 - obtenerInputOrto, 47
 - obtenerOutputOrto, 47
 - obtenerOutputRF, 47
 - runAll, 48
 - stitchImgs, 48
 - stitchWarp, 48
- Undistort, 49
 - escribirOutput, 49
 - obtenerInput, 50
 - undistortImgs, 50
- undistortImgs
 - Undistort, 50
- UtilInformation, 50
 - calcularHectareas, 51
- writeChart
 - Redaction, 36
- writelnImg
 - Redaction, 37
- writelnIndex
 - Redaction, 37
- writelnLut
 - Redaction, 38
- writelnOriginal
 - Redaction, 38
- writelnText
 - Redaction, 38