Meanshift

1.0

Generated by Doxygen 1.8.13

Contents

Chapter 1

Mean shift implementation

Author

Damir Demirović damir.demirovic@untz.ba

Mean shift algorithm represent a general non-parametric mode finding procedure. It's a hill-climbing algorithm on the density defined by a finite mixture or a kernel density estimate. Mean shift can be used as a non-parametric clustering method, for object tracking, image segmentation.

Chapter 2

Todo List

File io_png.c

handle lossless 16bit data

add a test suite

internally handle RGB/gray conversion in io_png_read_raw()

handle deinterlacing as a libpng transform function

Member io_png_read_u8 (const char *fname, size_t *nxp, size_t *nyp, size_t *ncp)

don't downscale 16bit images.

Member io_png_write_f32 (const char *fname, const float *data, size_t nx, size_t ny, size_t nc)

handle 16bit images and flexible min/max

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

Class Documentation

5.1 MSPoint Struct Reference

Public Attributes

- int **x**
- int **y**

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

- src/ms/ms.h

Chapter 6

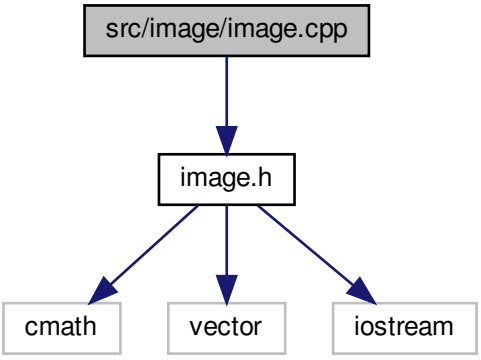
File Documentation

6.1 src/image/image.cpp File Reference

Image helper functions.

```
#include "image.h"
```

Include dependency graph for image.cpp:



Functions

- uchar * AllocateUcharImage (int width, int height, int nchannel)

  Allocate space for the uchar image,.
- int ** GenerateLabels (int width, int height)

  Generate labels for clustering.
- vector< int > GenerateRandomNumbers (int count)

  Function generate random numbers.
- void LabelImage (uchar *image, int width, int height, int **labels, int regCount)

  Function LabelImage in RGB colors.

- void RGB2LUV (float r, float g, float b, uchar ∗l, uchar ∗u, uchar ∗v)

  Function RGB2LUV converts RGB pixel value to LUV pixel value.
- void LUV2RGB (float L, float u, float v, uchar ∗R, uchar ∗G, uchar ∗B)

  Function LUV2RGB converts LUV pixel value to RGB pixel value.
- uchar ∗ ConvertRGB2LUV (uchar ∗rgb, int width, int height, int nchannel)

  Function ConvertRGB2LUV convert RGB image to LUV.
- uchar ∗ ConvertLUV2RGB (uchar ∗luv, int width, int height, int nchannel)

  Function ConvertLUV2RGB converts input image image from LUV to RGB color space.
- void SetPixel (uchar ∗im, int width, int height, int x, int y, const uchar val, int nchannel)

  Set Pixel at channel component of image at postition given with x and y.
- uchar GetPixel (uchar ∗im, int width, int height, int x, int y, int nchannel)

  Get Pixel at channel component of image at postition given with x and y.
- int range_distance (uchar ∗image, int width, int height, int x1, int y1, int x2, int y2)

  Function range_distance calculate range distance between two pixels.
- float color_distance (const float ∗a, const float ∗b)

  Function color_distance calculate color distance between the two pixels.

6.1.1 Detailed Description

Image helper functions.

Author

Damir Demirović damir.demirovic@untz.ba

6.1.2 Function Documentation

6.1.2.1 AllocateUcharImage()

```
uchar* AllocateUcharImage (
            int width,
            int height,
            int nchannel )
```

Allocate space for the uchar image,.

Parameters

width,height	and nchannel

Returns

img - allocated image

6.1.2.2 color_distance()

```
float color_distance (
            const float * a,
            const float * b )
```

Function color_distance calculate color distance between the two pixels.

Parameters

a,b	input images

Returns

the color distance

6.1.2.3 ConvertLUV2RGB()

```
uchar* ConvertLUV2RGB (
            uchar * luv,
            int width,
            int height,
            int nchannel )
```

Function ConvertLUV2RGB converts input image image from LUV to RGB color space.

Parameters

luv	LUV image to convert
width	width of the image
height	gheight of the image
nchannel	number of image channels

Returns

The input character

6.1.2.4 ConvertRGB2LUV()

```
uchar* ConvertRGB2LUV (
            uchar * rgb,
            int width,
            int height,
            int nchannel )
```

Function ConvertRGB2LUV convert RGB image to LUV.

Parameters

rgb	RGB image to convert
width	width of the image
height	gheight of the image
nchannel	number of image channels

Returns

luv the converted image

6.1.2.5 GenerateLabels()

```
int** GenerateLabels (
            int width,
            int height )
```

Generate labels for clustering.

Function generate labels.

Parameters

width,height	

Returns

labels

6.1.2.6 GenerateRandomNumbers()

```
vector<int> GenerateRandomNumbers (
            int count )
```

Function generate random numbers.

Parameters

count	- Count of numbers to be generated

Returns

number as vector<int>

6.1.2.7 GetPixel()

```
uchar GetPixel (
            uchar * im,
            int width,
            int height,
            int x,
            int y,
            int nchannel )
```

Get Pixel at channel component of image at postition given with x and y.

Parameters

im	image to convert
width	width of the image
height	gheight of the image
x	x position in the image 0 $<$ x $<$ width
y	y position in the image 0 $<$ y $<$ height
nchannel	number of image channels

Returns

pixel value

6.1.2.8 LabelImage()

```
void LabelImage (
            uchar * image,
            int width,
            int height,
            int ** labels,
            int regCount )
```

Function LabelImage in RGB colors.

Parameters

image	image to be labeled
width	width of the image
height	heighto of the image
labels	color labels
regCount	regions to be labeled

6.1.2.9 LUV2RGB()

```
void LUV2RGB (
            float L,
            float u,
            float v,
            uchar * R,
            uchar * G,
            uchar * B )  [inline]
```

Function LUV2RGB converts LUV pixel value to RGB pixel value.

Parameters

L	component of input image
u	component of input image
v	component of input image
r	component of output image
g	component of output image
b	component of output image

6.1.2.10 range_distance()

```
int range_distance (
            uchar * image,
            int width,
            int height,
            int x1,
            int y1,
            int x2,
            int y2 )
```

Function range_distance calculate range distance between two pixels.

Parameters

image	image to calculate
width	width of the image
height	height of the image
x1	first x position in the image
y1	first y position in the image
x2	second x position in the image
y2	second y position in the image

Returns

squared sum of distances

6.1.2.11 RGB2LUV()

```
void RGB2LUV (
            float r,
            float g,
            float b,
            uchar * l,
            uchar * u,
            uchar * v )  [inline]
```

Function RGB2LUV converts RGB pixel value to LUV pixel value.

Parameters

r	component of input image
g	component of input image
b	component of input image
l	component of output image
u	component of output image
v	component of output image 0<=l<=100, 134<=u<=220, 140<=v<=122

6.1.2.12 SetPixel()

```
void SetPixel (
            uchar * im,
            int width,
            int height,
            int x,
            int y,
            const uchar val,
            int nchannel )
```

Set Pixel at channel component of image at postition given with x and y.

Parameters

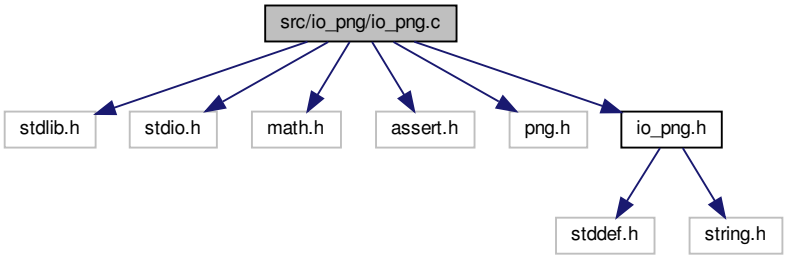
im	image to convert
width	width of the image
height	gheight of the image
x	x position in the image
y	y position in the image
val	value to set at pixel location
nchannel	number of image channels

6.2 src/io_png/io_png.c File Reference

PNG read/write simplified interface.

```
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <assert.h>
#include <png.h>
#include "io_png.h"
```

Include dependency graph for io_png.c:



Macros

- #define **PNG_SIG_LEN** 4
- #define **IO_PNG_U8** 0x0001 /∗ 8bit unsigned integer ∗/
- #define **IO_PNG_F32** 0x0002 /∗ 32bit float ∗/

Functions

- char ∗ io_png_info (void)

  helps tracking versions, via the string tag inserted into the library
- unsigned char ∗ io_png_read_u8 (const char ∗fname, size_t ∗nxp, size_t ∗nyp, size_t ∗ncp)

  read a PNG file into a 8bit integer array
- unsigned char ∗ io_png_read_u8_rgb (const char ∗fname, size_t ∗nxp, size_t ∗nyp)

  read a PNG file into a 8bit integer array, converted to RGB
- unsigned char ∗ io_png_read_u8_gray (const char ∗fname, size_t ∗nxp, size_t ∗nyp)

  read a PNG file into a 8bit integer array, converted to gray
- float ∗ io_png_read_f32 (const char ∗fname, size_t ∗nxp, size_t ∗nyp, size_t ∗ncp)

  read a PNG file into a 32bit float array
- float ∗ io_png_read_f32_rgb (const char ∗fname, size_t ∗nxp, size_t ∗nyp)

  read a PNG file into a 32bit float array, converted to RGB
- float ∗ io_png_read_f32_gray (const char ∗fname, size_t ∗nxp, size_t ∗nyp)

  read a PNG file into a 32bit float array, converted to gray
- int io_png_write_u8 (const char ∗fname, const unsigned char ∗data, size_t nx, size_t ny, size_t nc)

  write a 8bit unsigned integer array into a PNG file
- int io_png_write_f32 (const char ∗fname, const float ∗data, size_t nx, size_t ny, size_t nc)

  write a float array into a PNG file

6.2.1 Detailed Description

PNG read/write simplified interface.

This is a front-end to libpng, with routines to:

- read a PNG file as a deinterlaced 8bit integer or float array
- write a 8bit integer or float array to a PNG file

Multi-channel images are handled: grey, grey+alpha, rgb and rgb+alpha, as well as on-the-fly color model conversion.

Todo handle lossless 16bit data

add a test suite

internally handle RGB/gray conversion in io_png_read_raw()

handle deinterlacing as a libpng transform function

Author

Nicolas Limare nicolas.limare@cmla.ens-cachan.fr

6.2.2 Function Documentation

6.2.2.1 io_png_info()

```
char* io_png_info (
            void  )
```

helps tracking versions, via the string tag inserted into the library

This function is not expected to be used in real-world programs.

Returns

a pointer to a version info string

6.2.2.2 io_png_read_f32()

```
float* io_png_read_f32 (
            const char * fname,
            size_t * nxp,
            size_t * nyp,
            size_t * ncp )
```

read a PNG file into a 32bit float array

The array contains the deinterlaced channels. 1, 2, 4 and 8bit images are converted to float values between 0. and 1., 3., 15. or 255. 16bit images are also downscaled to 8bit before conversion.

Parameters

fname	PNG file name
nxp,nyp,ncp	pointers to variables to be filled with the number of columns, lines and channels of the image

Returns

pointer to an allocated unsigned char array of pixels, or NULL if an error happens

6.2.2.3 io_png_read_f32_gray()

```
float* io_png_read_f32_gray (
            const char * fname,
            size_t * nxp,
            size_t * nyp )
```

read a PNG file into a 32bit float array, converted to gray

See io_png_read_f32() for details.

6.2.2.4 io_png_read_f32_rgb()

```
float* io_png_read_f32_rgb (
            const char * fname,
            size_t * nxp,
            size_t * nyp )
```

read a PNG file into a 32bit float array, converted to RGB

See io_png_read_f32() for details.

6.2.2.5 io_png_read_u8()

```
unsigned char* io_png_read_u8 (
            const char * fname,
            size_t * nxp,
            size_t * nyp,
            size_t * ncp )
```

read a PNG file into a 8bit integer array

The array contains the deinterlaced channels. 1, 2 and 4bit images are converted to 8bit. 16bit images are previously downscaled to 8bit.

Todo don't downscale 16bit images.

Parameters

fname	PNG file name
nxp,nyp,ncp	pointers to variables to be filled with the number of columns, lines and channels of the image

Returns

pointer to an allocated unsigned char array of pixels, or NULL if an error happens

6.2.2.6 io_png_read_u8_gray()

```
unsigned char* io_png_read_u8_gray (
            const char * fname,
            size_t * nxp,
            size_t * nyp )
```

read a PNG file into a 8bit integer array, converted to gray

See io_png_read_u8() for details.

6.2.2.7 io_png_read_u8_rgb()

```
unsigned char* io_png_read_u8_rgb (
            const char * fname,
            size_t * nxp,
            size_t * nyp )
```

read a PNG file into a 8bit integer array, converted to RGB

See io_png_read_u8() for details.

6.2.2.8 io_png_write_f32()

```
int io_png_write_f32 (
            const char * fname,
            const float * data,
            size_t nx,
            size_t ny,
            size_t nc )
```

write a float array into a PNG file

The float values are rounded to 8bit integers, and bounded to [0, 255].

Todo handle 16bit images and flexible min/max

Parameters

fname	PNG file name
data	array to write
nx,ny,nc	number of columns, lines and channels of the image

Returns

0 if everything OK, -1 if an error occured

6.2.2.9 io_png_write_u8()

```
int io_png_write_u8 (
            const char * fname,
            const unsigned char * data,
            size_t nx,
            size_t ny,
            size_t nc )
```

write a 8bit unsigned integer array into a PNG file

Parameters

fname	PNG file name
data	array to write
nx,ny,nc	number of columns, lines and channels of the image

Returns

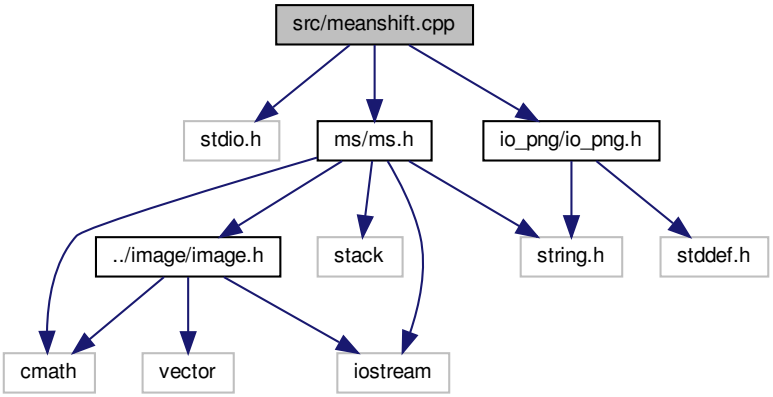
0 if everything OK, -1 if an error occured

6.3 src/meanshift.cpp File Reference

Main program for Meanshift segmentation.

```
#include <stdio.h>
#include "ms/ms.h"
#include "io_png/io_png.h"
```

Include dependency graph for meanshift.cpp:



Functions

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

6.3.1 Detailed Description

Main program for Meanshift segmentation.

Author

Damir Demirović damir.demirovic@untz.ba

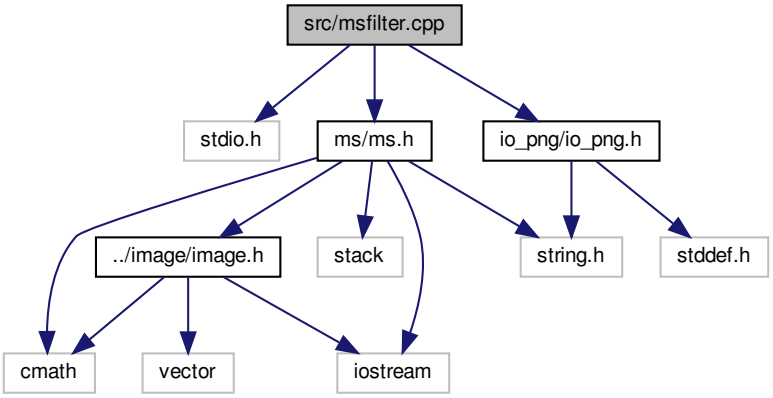
6.4 src/msfilter.cpp File Reference

Main program for Meanshift filtering.

```
#include <stdio.h>
#include "ms/ms.h"
```

```
#include "io_png/io_png.h"
```

Include dependency graph for msfilter.cpp:



Functions

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

6.4.1 Detailed Description

Main program for Meanshift filtering.

Author

Damir Demirović damir.demirovic@untz.ba