

MeshLib

1.4.1.0

Generated by Doxygen 1.8.9.1

Sat Jun 25 2016 20:54:43

Contents

1	Meshlib	1
1.1	Introduction	1
1.2	Build	1
1.3	Contents	1
2	Data Structure Index	3
2.1	Data Structures	3
3	File Index	5
3.1	File List	5
4	Data Structure Documentation	7
4.1	mesh Struct Reference	7
4.1.1	Field Documentation	8
4.1.1.1	dummy	8
4.1.1.2	edges	8
4.1.1.3	faces	8
4.1.1.4	fareas	8
4.1.1.5	fcolors	8
4.1.1.6	ffaces	8
4.1.1.7	fnormals	8
4.1.1.8	is_edges	8
4.1.1.9	is_faces	8
4.1.1.10	is_fareas	8
4.1.1.11	is_fcolors	9
4.1.1.12	is_ffaces	9
4.1.1.13	is_fnormals	9
4.1.1.14	is_loaded	9
4.1.1.15	is_trimesh	9
4.1.1.16	is_vcolors	9
4.1.1.17	is_vertices	9
4.1.1.18	is_vfaces	9

4.1.1.19	is_vnormals	9
4.1.1.20	num_edges	9
4.1.1.21	num_faces	9
4.1.1.22	num_vertices	9
4.1.1.23	origin_type	10
4.1.1.24	vcolors	10
4.1.1.25	vertices	10
4.1.1.26	vfaces	10
4.1.1.27	vnormals	10
4.2	mesh_adjface Struct Reference	10
4.2.1	Field Documentation	10
4.2.1.1	faces	10
4.2.1.2	num_faces	10
4.3	mesh_color Struct Reference	10
4.3.1	Field Documentation	11
4.3.1.1	a	11
4.3.1.2	b	11
4.3.1.3	g	11
4.3.1.4	r	11
4.4	mesh_edge Struct Reference	11
4.4.1	Field Documentation	11
4.4.1.1	faces	11
4.4.1.2	vertices	11
4.5	mesh_face Struct Reference	12
4.5.1	Field Documentation	12
4.5.1.1	num_vertices	12
4.5.1.2	vertices	12
4.6	mesh_rotation Struct Reference	12
4.6.1	Field Documentation	12
4.6.1.1	data	12
4.7	mesh_struct Struct Reference	12
4.7.1	Field Documentation	13
4.7.1.1	items	13
4.7.1.2	num_items	13
4.8	mesh_struct2 Struct Reference	13
4.8.1	Field Documentation	13
4.8.1.1	items	13
4.8.1.2	num_items	13
4.9	mesh_struct3 Struct Reference	13
4.9.1	Field Documentation	13

4.9.1.1	items	13
4.9.1.2	num_items	14
4.10	mesh_transform Struct Reference	14
4.10.1	Field Documentation	14
4.10.1.1	data	14
4.11	mesh_vector3 Struct Reference	14
4.11.1	Field Documentation	14
4.11.1.1	x	14
4.11.1.2	y	14
4.11.1.3	z	14
5	File Documentation	17
5.1	meshcalc.c File Reference	17
5.1.1	Detailed Description	18
5.1.2	Function Documentation	18
5.1.2.1	mesh_calc_edges	18
5.1.2.2	mesh_calc_face_adjacency	19
5.1.2.3	mesh_calc_face_normal	19
5.1.2.4	mesh_calc_face_normals	20
5.1.2.5	mesh_calc_triangle_area	21
5.1.2.6	mesh_calc_vertex_adjacency	22
5.1.2.7	mesh_calc_vertex_normals	23
5.1.2.8	mesh_cross_normal	24
5.1.2.9	mesh_cross_vector3	24
5.1.2.10	mesh_find	25
5.1.2.11	mesh_find2	26
5.1.2.12	mesh_find3	26
5.1.2.13	mesh_upsample	26
5.2	meshclean.c File Reference	27
5.2.1	Detailed Description	27
5.2.2	Function Documentation	28
5.2.2.1	mesh_remove_boundary_faces	28
5.2.2.2	mesh_remove_boundary_vertices	28
5.2.2.3	mesh_remove_close_vertices	28
5.2.2.4	mesh_remove_ear_faces	28
5.2.2.5	mesh_remove_triangles_with_small_area	29
5.2.2.6	mesh_remove_unreferenced_vertices	30
5.2.2.7	mesh_remove_zero_area_faces	31
5.3	meshcreate.c File Reference	32
5.3.1	Detailed Description	32

5.3.2	Function Documentation	33
5.3.2.1	mesh_create_mesh_new	33
5.3.2.2	mesh_create_mesh_new_cone	34
5.3.2.3	mesh_create_mesh_new_cuboid	34
5.3.2.4	mesh_create_mesh_new_cylinder	34
5.3.2.5	mesh_create_mesh_new_ellipsoid	35
5.3.2.6	mesh_free_mesh	35
5.4	meshdraw.c File Reference	36
5.4.1	Detailed Description	37
5.4.2	Function Documentation	37
5.4.2.1	mesh_draw_mesh	37
5.4.2.2	mesh_draw_mesh_smooth	37
5.4.2.3	mesh_draw_point_cloud	38
5.5	mesherror.c File Reference	38
5.5.1	Detailed Description	39
5.5.2	Function Documentation	39
5.5.2.1	mesh_error	39
5.6	meshfilter.c File Reference	40
5.6.1	Detailed Description	41
5.6.2	Function Documentation	41
5.6.2.1	mesh_bilateral_filter	41
5.6.2.2	mesh_laplacian_filter	42
5.6.2.3	mesh_restricted_laplacian_filter	42
5.7	meshlib.h File Reference	43
5.7.1	Detailed Description	48
5.7.2	Macro Definition Documentation	48
5.7.2.1	_CRT_SECURE_NO_DEPRECATED	48
5.7.2.2	FLOATDATA	48
5.7.2.3	INTDATA	48
5.7.2.4	MESH_CLONE_ALL_PROPS	48
5.7.2.5	MESH_CLONE_EDGES	48
5.7.2.6	MESH_CLONE_F_ALL_PROPS	48
5.7.2.7	MESH_CLONE_FACES	48
5.7.2.8	MESH_CLONE_FAREAS	48
5.7.2.9	MESH_CLONE_FCOLORS	48
5.7.2.10	MESH_CLONE_FFACES	49
5.7.2.11	MESH_CLONE_FNORMALS	49
5.7.2.12	MESH_CLONE_V_ALL_PROPS	49
5.7.2.13	MESH_CLONE_VCOLORS	49
5.7.2.14	MESH_CLONE_VERTICES	49

5.7.2.15	MESH_CLONE_VFACES	49
5.7.2.16	MESH_CLONE_VNORMALS	49
5.7.2.17	MESH_ERR_FNOTOPEN	49
5.7.2.18	MESH_ERR_INCOMPATIBLE	49
5.7.2.19	MESH_ERR_MALLOC	49
5.7.2.20	MESH_ERR_SIZE_MISMATCH	49
5.7.2.21	MESH_ERR_UNKNOWN	49
5.7.2.22	MESH_FLOATDATA_TYPE	50
5.7.2.23	MESH_INTDATA_TYPE	50
5.7.2.24	MESH_ORIGIN_TYPE_BUILD	50
5.7.2.25	MESH_ORIGIN_TYPE_COFF	50
5.7.2.26	MESH_ORIGIN_TYPE_NCOFF	50
5.7.2.27	MESH_ORIGIN_TYPE_NOFF	50
5.7.2.28	MESH_ORIGIN_TYPE_OFF	50
5.7.2.29	MESH_ORIGIN_TYPE_PLY_ASCII	50
5.7.2.30	MESH_ORIGIN_TYPE_PLY_BINARY_BIG_ENDIAN	50
5.7.2.31	MESH_ORIGIN_TYPE_PLY_BINARY_LITTLE_ENDIAN	50
5.7.2.32	MESH_ORIGIN_TYPE_XYZ	50
5.7.2.33	MESH_PI	50
5.7.2.34	MESH_TWOPI	51
5.7.2.35	MESHLIBAPI	51
5.7.3	Typedef Documentation	51
5.7.3.1	FILEPOINTER	51
5.7.3.2	INTDATA2	51
5.7.3.3	INTDATA3	51
5.7.3.4	mesh	51
5.7.3.5	MESH	51
5.7.3.6	mesh_adjface	51
5.7.3.7	mesh_color	51
5.7.3.8	MESH_COLOR	51
5.7.3.9	mesh_edge	51
5.7.3.10	MESH_EDGE	51
5.7.3.11	mesh_face	51
5.7.3.12	MESH_FACE	52
5.7.3.13	mesh_fface	52
5.7.3.14	MESH_FFACE	52
5.7.3.15	mesh_normal	52
5.7.3.16	MESH_NORMAL	52
5.7.3.17	mesh_rotation	52
5.7.3.18	MESH_ROTATION	52

5.7.3.19	<code>mesh_struct</code>	52
5.7.3.20	<code>MESH_STRUCT</code>	52
5.7.3.21	<code>mesh_struct2</code>	52
5.7.3.22	<code>MESH_STRUCT2</code>	52
5.7.3.23	<code>mesh_struct3</code>	52
5.7.3.24	<code>MESH_STRUCT3</code>	53
5.7.3.25	<code>mesh_transform</code>	53
5.7.3.26	<code>MESH_TRANSFORM</code>	53
5.7.3.27	<code>mesh_vector3</code>	53
5.7.3.28	<code>MESH_VECTOR3</code>	53
5.7.3.29	<code>mesh_vertex</code>	53
5.7.3.30	<code>MESH_VERTEX</code>	53
5.7.3.31	<code>mesh_vface</code>	53
5.7.3.32	<code>MESH_VFACE</code>	53
5.7.4	Function Documentation	53
5.7.4.1	<code>mesh_bilateral_filter</code>	53
5.7.4.2	<code>mesh_calc_edges</code>	54
5.7.4.3	<code>mesh_calc_face_adjacency</code>	55
5.7.4.4	<code>mesh_calc_face_normal</code>	56
5.7.4.5	<code>mesh_calc_face_normals</code>	56
5.7.4.6	<code>mesh_calc_triangle_area</code>	57
5.7.4.7	<code>mesh_calc_vertex_adjacency</code>	58
5.7.4.8	<code>mesh_calc_vertex_normals</code>	59
5.7.4.9	<code>mesh_clone_mesh</code>	60
5.7.4.10	<code>mesh_combine_mesh</code>	61
5.7.4.11	<code>mesh_count_words_in_line</code>	61
5.7.4.12	<code>mesh_create_mesh_new</code>	61
5.7.4.13	<code>mesh_create_mesh_new_cone</code>	62
5.7.4.14	<code>mesh_create_mesh_new_cuboid</code>	63
5.7.4.15	<code>mesh_create_mesh_new_cylinder</code>	63
5.7.4.16	<code>mesh_create_mesh_new_ellipsoid</code>	64
5.7.4.17	<code>mesh_cross_normal</code>	64
5.7.4.18	<code>mesh_cross_vector3</code>	65
5.7.4.19	<code>mesh_draw_mesh</code>	65
5.7.4.20	<code>mesh_draw_mesh_smooth</code>	65
5.7.4.21	<code>mesh_draw_point_cloud</code>	66
5.7.4.22	<code>mesh_error</code>	66
5.7.4.23	<code>mesh_find</code>	67
5.7.4.24	<code>mesh_find2</code>	67
5.7.4.25	<code>mesh_find3</code>	68

5.7.4.26	mesh_free_mesh	68
5.7.4.27	mesh_go_next_word	68
5.7.4.28	mesh_isnumeric	69
5.7.4.29	mesh_laplacian_filter	69
5.7.4.30	mesh_load_file	69
5.7.4.31	mesh_load_off	70
5.7.4.32	mesh_load_ply	71
5.7.4.33	mesh_load_xyz	72
5.7.4.34	mesh_read_word	73
5.7.4.35	mesh_read_word_only	73
5.7.4.36	mesh_remove_boundary_faces	73
5.7.4.37	mesh_remove_boundary_vertices	73
5.7.4.38	mesh_remove_close_vertices	74
5.7.4.39	mesh_remove_ear_faces	74
5.7.4.40	mesh_remove_triangles_with_small_area	75
5.7.4.41	mesh_remove_unreferenced_vertices	75
5.7.4.42	mesh_remove_zero_area_faces	76
5.7.4.43	mesh_restricted_laplacian_filter	77
5.7.4.44	mesh_rotate	77
5.7.4.45	mesh_rotation_create	78
5.7.4.46	mesh_rotation_free	78
5.7.4.47	mesh_rotation_set_angleaxis	79
5.7.4.48	mesh_rotation_set_matrix	79
5.7.4.49	mesh_scale	80
5.7.4.50	mesh_skip_line	81
5.7.4.51	mesh_translate	81
5.7.4.52	mesh_translate_vector	81
5.7.4.53	mesh_upsample	82
5.7.4.54	mesh_vertex_rotate	82
5.7.4.55	mesh_write_file	83
5.7.4.56	mesh_write_off	83
5.7.4.57	mesh_write_ply	84
5.7.4.58	mesh_write_xyz	85
5.8	meshload.c File Reference	85
5.8.1	Detailed Description	86
5.8.2	Function Documentation	86
5.8.2.1	mesh_load_file	86
5.8.2.2	mesh_load_off	87
5.8.2.3	mesh_load_ply	88
5.8.2.4	mesh_load_xyz	88

5.9	meshops.c File Reference	89
5.9.1	Detailed Description	90
5.9.2	Function Documentation	90
5.9.2.1	mesh_clone_mesh	90
5.9.2.2	mesh_combine_mesh	91
5.10	meshtext.c File Reference	91
5.10.1	Detailed Description	92
5.10.2	Function Documentation	92
5.10.2.1	mesh_count_words_in_line	92
5.10.2.2	mesh_go_next_word	93
5.10.2.3	mesh_isnumeric	93
5.10.2.4	mesh_read_word	93
5.10.2.5	mesh_read_word_only	93
5.10.2.6	mesh_skip_line	94
5.11	meshttransform.c File Reference	94
5.11.1	Detailed Description	95
5.11.2	Function Documentation	95
5.11.2.1	mesh_rotate	95
5.11.2.2	mesh_rotation_create	96
5.11.2.3	mesh_rotation_free	96
5.11.2.4	mesh_rotation_set_angleaxis	96
5.11.2.5	mesh_rotation_set_matrix	97
5.11.2.6	mesh_scale	97
5.11.2.7	mesh_translate	98
5.11.2.8	mesh_translate_vector	98
5.11.2.9	mesh_vertex_rotate	99
5.12	meshwrite.c File Reference	100
5.12.1	Detailed Description	100
5.12.2	Function Documentation	101
5.12.2.1	mesh_write_file	101
5.12.2.2	mesh_write_off	101
5.12.2.3	mesh_write_ply	102
5.12.2.4	mesh_write_xyz	103

Chapter 1

Meshlib

1.1 Introduction

Meshlib is a simple mesh library written in C.

1.2 Build

To build the whole project, Code::blocks is required.

1.3 Contents

Load/Write PLY, OFF, ASC files.

Basic Vertex Manipulations.

Basic Vertex Transformations.

Basic Face Manipulations.

Bilateral Filtering.

Laplacian Filtering.

Mesh Cleaning Algorithms.

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

mesh	7
mesh_adjface	10
mesh_color	10
mesh_edge	11
mesh_face	12
mesh_rotation	12
mesh_struct	12
mesh_struct2	13
mesh_struct3	13
mesh_transform	14
mesh_vector3	14

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

meshcalc.c	This file contains functions pertaining to different mesh computations	17
meshclean.c	This file contains functions pertaining to different mesh cleaning algorithms	27
meshcreate.c	This file contains functions pertaining to mesh creation and freeing	32
meshdraw.c	This file contains functions pertaining to mesh drawing in OpenGL	36
mesherror.c	This file contains functions pertaining to handling errors	38
meshfilter.c	This file contains functions pertaining to different mesh filtering algorithms	40
meshlib.h	This header file contains declarations of all functions of meshlib	43
meshload.c	This file contains functions pertaining to loading different mesh file types	85
meshops.c	This file contains functions pertaining to mesh combinatorial operations	89
meshtext.c	This file contains functions pertaining to different text routines	91
meshtransform.c	This file contains functions pertaining to different mesh transformations	94
meshwrite.c	This file contains functions pertaining to writing different mesh file types	100

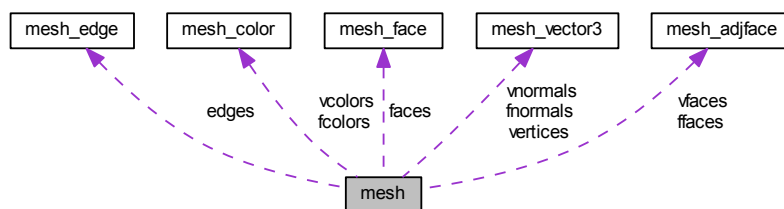
Chapter 4

Data Structure Documentation

4.1 mesh Struct Reference

```
#include <meshlib.h>
```

Collaboration diagram for mesh:



Data Fields

- `uint8_t origin_type`
- `uint8_t is_loaded`
- `uint8_t is_vertices`
- `uint8_t is_faces`
- `uint8_t is_edges`
- `uint8_t is_vnormals`
- `uint8_t is_fnormals`
- `uint8_t is_vcolors`
- `uint8_t is_fcolors`
- `uint8_t is_vfaces`
- `uint8_t is_ffaces`
- `uint8_t is_fareas`
- `INTDATA num_vertices`
- `INTDATA num_faces`
- `INTDATA num_edges`
- `MESH_VERTEX` vertices
- `MESH_FACE` faces
- `MESH_EDGE` edges
- `MESH_NORMAL` vnormals

- [MESH_NORMAL fnormals](#)
- [MESH_COLOR vcolors](#)
- [MESH_COLOR fcolors](#)
- [MESH_VFACE vfaces](#)
- [MESH_FFACE ffaces](#)
- [FLOATDATA * fareas](#)
- [uint8_t is_trimesh](#)
- [uint8_t dummy](#)

4.1.1 Field Documentation

4.1.1.1 uint8_t dummy

4.1.1.2 MESH_EDGE edges

Pointer to edges

4.1.1.3 MESH_FACE faces

Pointer to faces

4.1.1.4 FLOATDATA* fareas

Pointer to face areas

4.1.1.5 MESH_COLOR fcolors

Pointer to face colors

4.1.1.6 MESH_FFACE ffaces

Pointer to face adjacent faces

4.1.1.7 MESH_NORMAL fnormals

Pointer to face normals

4.1.1.8 uint8_t is_edges

Has edges?

4.1.1.9 uint8_t is_faces

Has faces?

4.1.1.10 uint8_t is_fareas

Has face areas?

4.1.1.11 `uint8_t is_fcolors`

Has face colors?

4.1.1.12 `uint8_t is_ffaces`

Has face adjacent faces?

4.1.1.13 `uint8_t is_fnormals`

Has face normals?

4.1.1.14 `uint8_t is_loaded`

Is loaded?

4.1.1.15 `uint8_t is_trimesh`

Is trimesh?

4.1.1.16 `uint8_t is_vcolors`

Has vertex colors?

4.1.1.17 `uint8_t is_vertices`

Has vertices?

4.1.1.18 `uint8_t is_vfaces`

Has vertex adjacent faces?

4.1.1.19 `uint8_t is_vnormals`

Has vertex normals?

4.1.1.20 `INTDATA num_edges`

Number of edges

4.1.1.21 `INTDATA num_faces`

Number of faces

4.1.1.22 `INTDATA num_vertices`

Number of vertices

4.1.1.23 `uint8_t origin_type`

Origin type

4.1.1.24 `MESH_COLOR vcolors`

Pointer to vertex colors

4.1.1.25 `MESH_VERTEX vertices`

Pointer to vertices

4.1.1.26 `MESH_VFACE vfaces`

Pointer to vertex adjacent faces

4.1.1.27 `MESH_NORMAL vnormals`

Pointer to vertex normals

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

- [meshlib.h](#)

4.2 `mesh_adjface` Struct Reference

```
#include <meshlib.h>
```

Data Fields

- [INTDATA num_faces](#)
- [INTDATA * faces](#)

4.2.1 Field Documentation

4.2.1.1 `INTDATA* faces`

Pointer to adjacent face indices

4.2.1.2 `INTDATA num_faces`

Number of adjacent faces

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

- [meshlib.h](#)

4.3 `mesh_color` Struct Reference

```
#include <meshlib.h>
```

Data Fields

- [FLOATDATA r](#)
- [FLOATDATA g](#)
- [FLOATDATA b](#)
- [FLOATDATA a](#)

4.3.1 Field Documentation

4.3.1.1 FLOATDATA a

Alpha channel

4.3.1.2 FLOATDATA b

Green channel

4.3.1.3 FLOATDATA g

Blue channel

4.3.1.4 FLOATDATA r

Red channel

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

- [meshlib.h](#)

4.4 mesh_edge Struct Reference

```
#include <meshlib.h>
```

Data Fields

- [INTDATA vertices](#) [2]
- [INTDATA faces](#) [2]

4.4.1 Field Documentation

4.4.1.1 INTDATA faces[2]

Edge faces

4.4.1.2 INTDATA vertices[2]

Edge vertices

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

- [meshlib.h](#)

4.5 mesh_face Struct Reference

```
#include <meshlib.h>
```

Data Fields

- [INTDATA num_vertices](#)
- [INTDATA * vertices](#)

4.5.1 Field Documentation

4.5.1.1 INTDATA num_vertices

Number of vertices

4.5.1.2 INTDATA* vertices

Pointer to vertex indices

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

- [meshlib.h](#)

4.6 mesh_rotation Struct Reference

```
#include <meshlib.h>
```

Data Fields

- [FLOATDATA data](#) [9]

4.6.1 Field Documentation

4.6.1.1 FLOATDATA data[9]

Matrix data

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

- [meshlib.h](#)

4.7 mesh_struct Struct Reference

```
#include <meshlib.h>
```

Data Fields

- [INTDATA num_items](#)
- [INTDATA * items](#)

4.7.1 Field Documentation

4.7.1.1 INTDATA* items

Pointer to INTDATA items

4.7.1.2 INTDATA num_items

Number of items

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

- [meshlib.h](#)

4.8 mesh_struct2 Struct Reference

```
#include <meshlib.h>
```

Data Fields

- [INTDATA num_items](#)
- [INTDATA2 * items](#)

4.8.1 Field Documentation

4.8.1.1 INTDATA2* items

Pointer to INTDATA2 items

4.8.1.2 INTDATA num_items

Number of items

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

- [meshlib.h](#)

4.9 mesh_struct3 Struct Reference

```
#include <meshlib.h>
```

Data Fields

- [INTDATA num_items](#)
- [INTDATA3 * items](#)

4.9.1 Field Documentation

4.9.1.1 INTDATA3* items

Pointer to INTDATA3 items

4.9.1.2 INTDATA num_items

Number of items

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

- [meshlib.h](#)

4.10 mesh_transform Struct Reference

```
#include <meshlib.h>
```

Data Fields

- [FLOATDATA * data](#)

4.10.1 Field Documentation

4.10.1.1 FLOATDATA* data

Matrix data

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

- [meshlib.h](#)

4.11 mesh_vector3 Struct Reference

```
#include <meshlib.h>
```

Data Fields

- [FLOATDATA x](#)
- [FLOATDATA y](#)
- [FLOATDATA z](#)

4.11.1 Field Documentation

4.11.1.1 FLOATDATA x

x co-ordinate

4.11.1.2 FLOATDATA y

y co-ordinate

4.11.1.3 FLOATDATA z

z co-ordinate

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

- [meshlib.h](#)

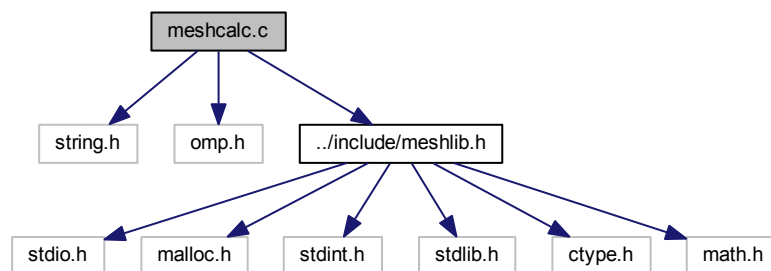
Chapter 5

File Documentation

5.1 meshcalc.c File Reference

This file contains functions pertaining to different mesh computations.

```
#include <string.h>
#include <omp.h>
#include "../include/meshlib.h"
Include dependency graph for meshcalc.c:
```



Functions

- void `mesh_cross_vector3` (`MESH_VECTOR3` x, `MESH_VECTOR3` y, `MESH_VECTOR3` z)
Computes the cross product of two 3-d vectors.
- void `mesh_cross_normal` (`MESH_NORMAL` x, `MESH_NORMAL` y, `MESH_NORMAL` z)
Computes the normalized cross product of two normals.
- void `mesh_calc_face_normal` (`MESH_VERTEX` v1, `MESH_VERTEX` v2, `MESH_VERTEX` v3, `MESH_NORMAL` n)
Computes the face normal given 3 vertices.
- int `mesh_calc_vertex_normals` (`MESH` m)
Computes vertex normals of a given mesh.
- int `mesh_calc_face_normals` (`MESH` m)
Computes face normals of a given mesh.
- int `mesh_calc_edges` (`MESH` m)
Computes edges of a given mesh.

- int [mesh_calc_vertex_adjacency](#) (MESH m)
Computes vertex adjacent faces of a given mesh.
- int [mesh_calc_face_adjacency](#) (MESH m)
Computes face adjacent faces of a given mesh.
- INTDATA [mesh_find](#) (MESH_STRUCT s, INTDATA q)
Finds an item in an INTDATA structure.
- INTDATA [mesh_find2](#) (MESH_STRUCT2 s, INTDATA q)
Finds an item in an INTDATA2 structure.
- INTDATA [mesh_find3](#) (MESH_STRUCT3 s, INTDATA q)
Finds an item in an INTDATA3 structure.
- int [mesh_upsample](#) (MESH m, int iters)
Upsamples a given mesh.
- FLOATDATA [mesh_calc_triangle_area](#) (MESH_VERTEX a, MESH_VERTEX b, MESH_VERTEX c)
Computes area of a triangle.

5.1.1 Detailed Description

This file contains functions pertaining to different mesh computations.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

5.1.2 Function Documentation

5.1.2.1 int mesh_calc_edges (MESH m)

Computes edges of a given mesh.

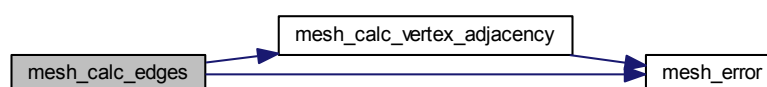
Parameters

in	m	Input mesh
----	---	------------

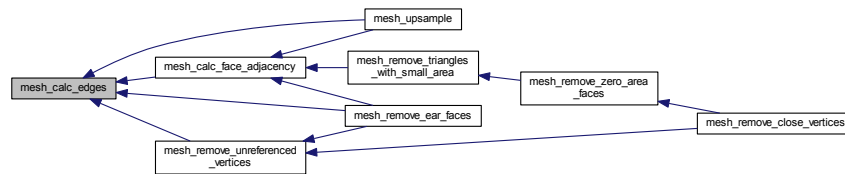
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.1.2.2 int mesh_calc_face_adjacency (MESH *m*)

Computes face adjacent faces of a given mesh.

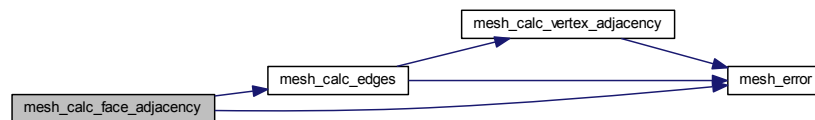
Parameters

in	<i>m</i>	Input mesh
----	----------	------------

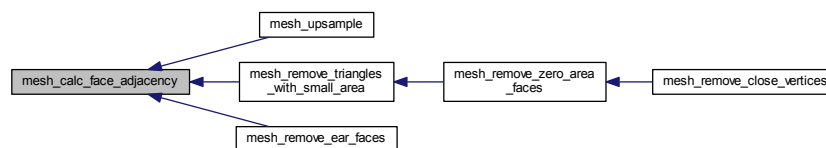
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.1.2.3 void mesh_calc_face_normal (MESH_VERTEX *v1*, MESH_VERTEX *v2*, MESH_VERTEX *v3*, MESH_NORMAL *n*)

Computes the face normal given 3 vertices.

Parameters

in	$v1$	First vertex
in	$v2$	Second vertex
in	$v3$	Third vertex
out	n	Output face normal \mathbf{n}_f

Returns

NULL

5.1.2.4 int mesh_calc_face_normals (MESH m)

Computes face normals of a given mesh.

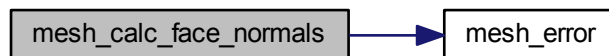
Parameters

in	m	Input mesh
----	-----	------------

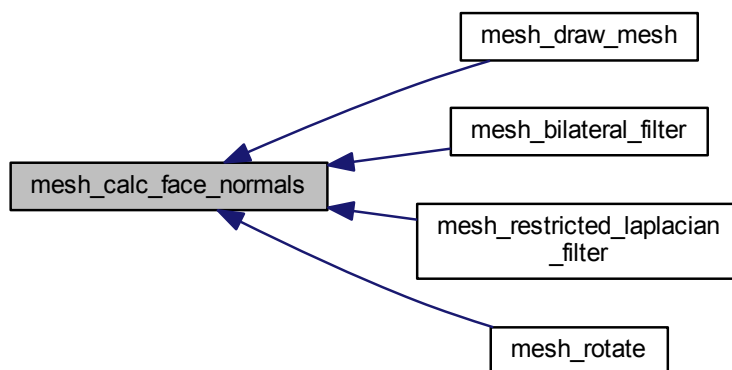
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.1.2.5 **FloatData** mesh_calc_triangle_area (MESH_VERTEX *a*, MESH_VERTEX *b*, MESH_VERTEX *c*)

Computes area of a triangle.

Parameters

in	<i>a</i>	First vertex
in	<i>b</i>	Second vertex
in	<i>c</i>	Third vertex

Returns

Area

Here is the call graph for this function:



Here is the caller graph for this function:



5.1.2.6 int mesh_calc_vertex_adjacency (MESH *m*)

Computes vertex adjacent faces of a given mesh.

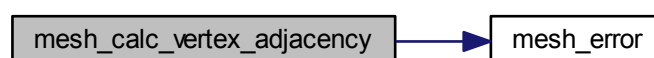
Parameters

in	<i>m</i>	Input mesh
----	----------	------------

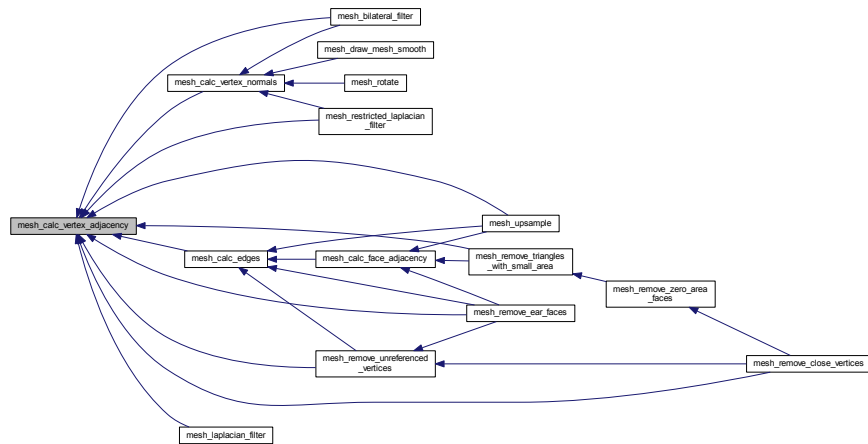
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.1.2.7 int mesh_calc_vertex_normals (MESH *m*)

Computes vertex normals of a given mesh.

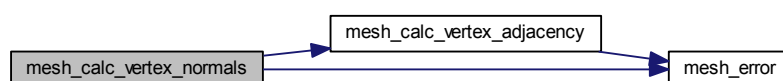
Parameters

in	<i>m</i>	Input mesh
----	----------	------------

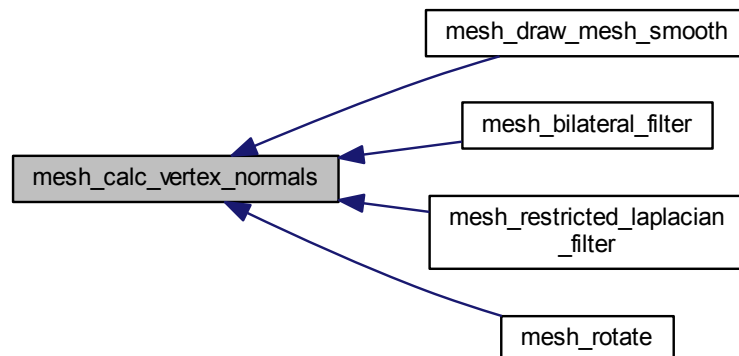
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.1.2.8 void mesh_cross_normal (MESH_NORMAL x, MESH_NORMAL y, MESH_NORMAL z)

Computes the normalized cross product of two normals.

Parameters

in	x	First normal
in	y	Second normal
out	z	Output cross product $\frac{\mathbf{x} \times \mathbf{y}}{\ \mathbf{x} \times \mathbf{y}\ _2}$

Returns

NULL

5.1.2.9 void mesh_cross_vector3 (MESH_VECTOR3 x, MESH_VECTOR3 y, MESH_VECTOR3 z)

Computes the cross product of two 3-d vectors.

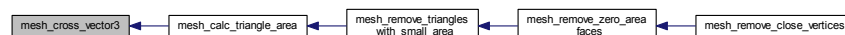
Parameters

in	x	First vector
in	y	Second vector
out	z	Output cross product $\mathbf{x} \times \mathbf{y}$

Returns

NULL

Here is the caller graph for this function:



5.1.2.10 INTDATA mesh_find (MESH_STRUCT *s*, INTDATA *q*)

Finds an item in an INTDATA structure.

Parameters

in	<i>s</i>	Input INTDATA structure
in	<i>q</i>	Query INTDATA

Returns

Index or -1

5.1.2.11 INTDATA mesh_find2 (MESH_STRUCT2 *s*, INTDATA *q*)

Finds an item in an INTDATA2 structure.

Parameters

in	<i>s</i>	Input INTDATA2 structure
in	<i>q</i>	Query INTDATA2

Returns

Index or -1

5.1.2.12 INTDATA mesh_find3 (MESH_STRUCT3 *s*, INTDATA *q*)

Finds an item in an INTDATA3 structure.

Parameters

in	<i>s</i>	Input INTDATA3 structure
in	<i>q</i>	Query INTDATA3

Returns

Index or -1

5.1.2.13 int mesh_upsample (MESH *m*, int *iters*)

Upsamples a given mesh.

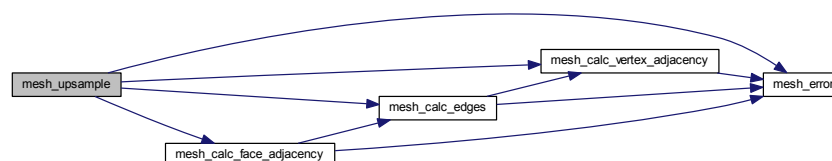
Parameters

in	<i>m</i>	Input mesh
in	<i>iters</i>	Number of iterations

Returns

Error code

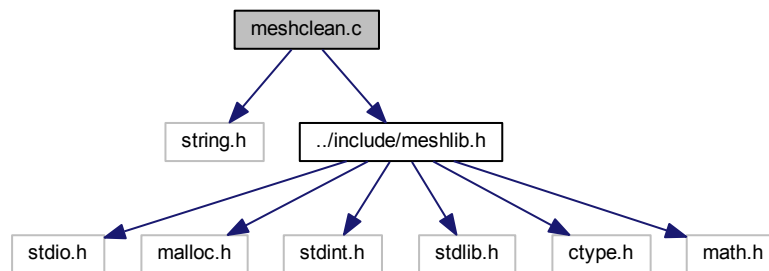
Here is the call graph for this function:



5.2 meshclean.c File Reference

This file contains functions pertaining to different mesh cleaning algorithms.

```
#include <string.h>
#include "../include/meshlib.h"
Include dependency graph for meshclean.c:
```



Functions

- int [mesh_remove_boundary_vertices](#) (MESH m, int iters)
Removes boundary vertices and connecting elements.
- int [mesh_remove_boundary_faces](#) (MESH m, int iters)
Removes boundary faces and connecting elements.
- int [mesh_remove_triangles_with_small_area](#) (MESH m, FLOATDATA area)
Removes triangles with area smaller than a given value.
- int [mesh_remove_zero_area_faces](#) (MESH m)
Removes triangles with zero area.
- int [mesh_remove_unreferenced_vertices](#) (MESH m)
Removes unreferenced vertices.
- int [mesh_remove_ear_faces](#) (MESH m, int niters)
Removes ear faces and connecting vertices.
- int [mesh_remove_close_vertices](#) (MESH m, FLOATDATA r)
Removes close vertices.

5.2.1 Detailed Description

This file contains functions pertaining to different mesh cleaning algorithms.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

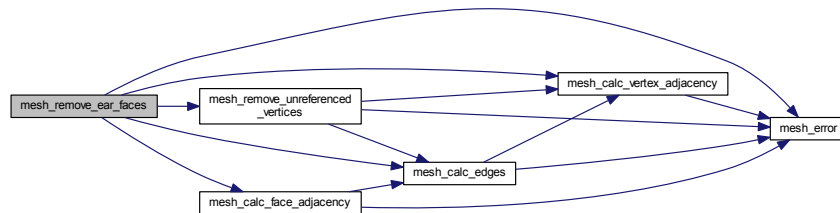
Parameters

in	m	Input mesh
in	$niters$	Number of iterations

Returns

Error code

Here is the call graph for this function:



5.2.2.5 int mesh_remove_triangles_with_small_area (MESH *m*, FLOATDATA *area*)

Removes triangles with area smaller than a given value.

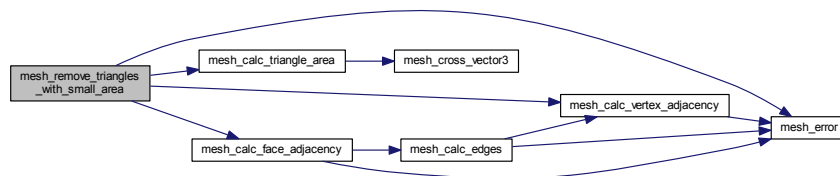
Parameters

in	m	Input mesh
in	$area$	Given area

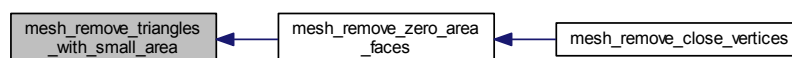
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.2.2.6 `int mesh_remove_unreferenced_vertices (MESH m)`

Removes unreferenced vertices.

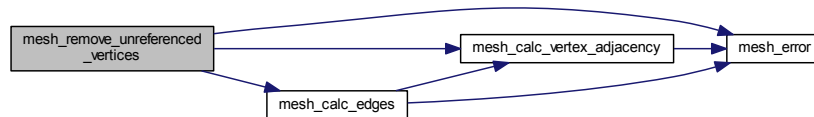
Parameters

in	<i>m</i>	Input mesh
----	----------	------------

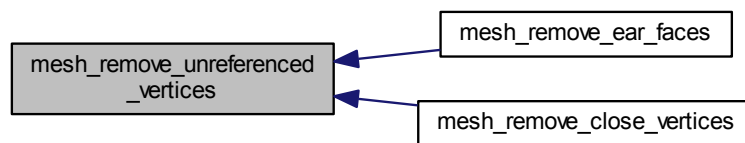
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.2.2.7 int mesh_remove_zero_area_faces (MESH *m*)

Removes triangles with zero area.

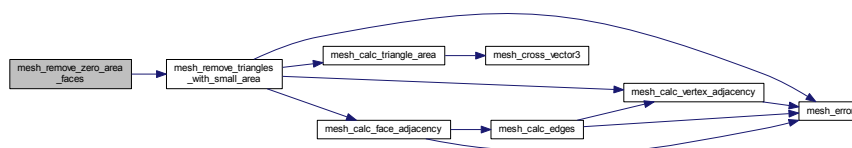
Parameters

in	<i>m</i>	Input mesh
----	----------	------------

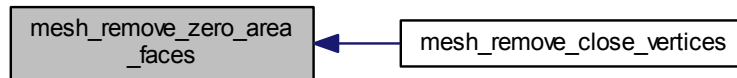
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:

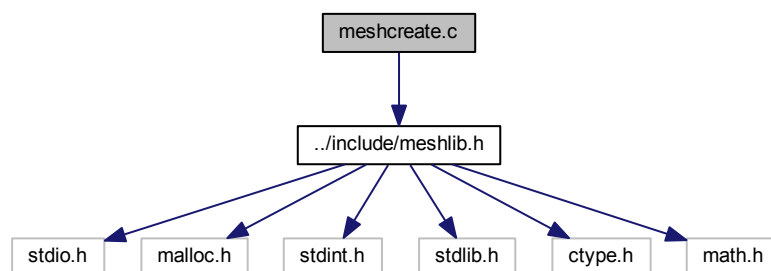


5.3 meshcreate.c File Reference

This file contains functions pertaining to mesh creation and freeing.

```
#include "../include/meshlib.h"
```

Include dependency graph for meshcreate.c:



Functions

- [MESH mesh_create_mesh_new \(\)](#)
Creates a new mesh.
- void [mesh_free_mesh \(MESH m\)](#)
Frees a mesh.
- [MESH mesh_create_mesh_new_cuboid \(MESH_VECTOR3 sz, MESH_VECTOR3 pos\)](#)
Creates a cuboid mesh.
- [MESH mesh_create_mesh_new_ellipsoid \(MESH_VECTOR3 sz, MESH_VECTOR3 pos\)](#)
Creates an ellipsoid mesh.
- [MESH mesh_create_mesh_new_cylinder \(MESH_VECTOR3 sz, MESH_VECTOR3 pos\)](#)
Creates a cylinder mesh.
- [MESH mesh_create_mesh_new_cone \(MESH_VECTOR3 sz, MESH_VECTOR3 pos\)](#)
Creates a cone mesh.

5.3.1 Detailed Description

This file contains functions pertaining to mesh creation and freeing.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

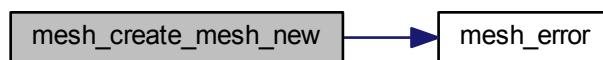
5.3.2 Function Documentation**5.3.2.1 MESH mesh_create_mesh_new ()**

Creates a new mesh.

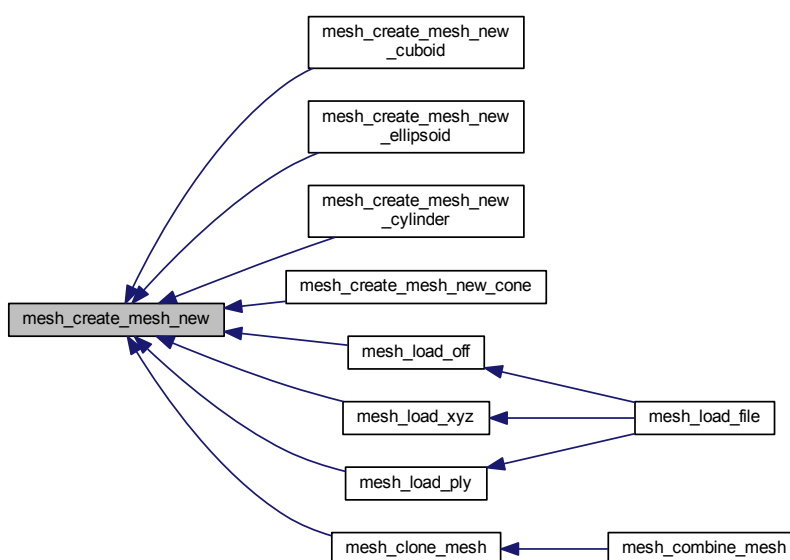
Returns

Output mesh

Here is the call graph for this function:



Here is the caller graph for this function:



5.3.2.2 MESH mesh_create_mesh_new_cone (MESH_VECTOR3 sz, MESH_VECTOR3 pos)

Creates a cone mesh.

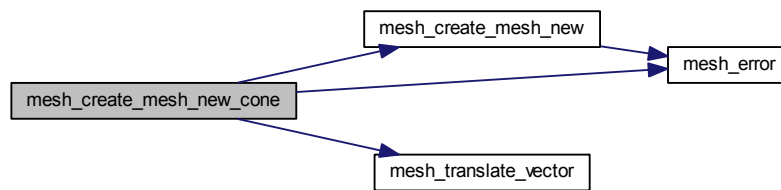
Parameters

in	sz	Size vector
in	pos	Position vector

Returns

Output mesh

Here is the call graph for this function:



5.3.2.3 MESH mesh_create_mesh_new_cuboid (MESH_VECTOR3 sz, MESH_VECTOR3 pos)

Creates a cuboid mesh.

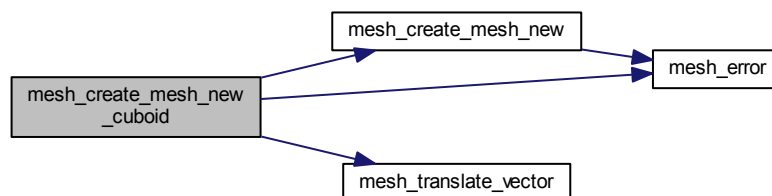
Parameters

in	sz	Size vector
in	pos	Position vector

Returns

Output mesh

Here is the call graph for this function:



5.3.2.4 MESH mesh_create_mesh_new_cylinder (MESH_VECTOR3 sz, MESH_VECTOR3 pos)

Creates a cylinder mesh.

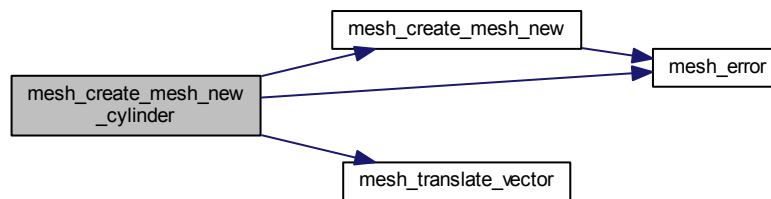
Parameters

in	<i>sz</i>	Size vector
in	<i>pos</i>	Position vector

Returns

Output mesh

Here is the call graph for this function:



5.3.2.5 MESH mesh_create_mesh_new_ellipsoid (MESH_VECTOR3 sz, MESH_VECTOR3 pos)

Creates an ellipsoid mesh.

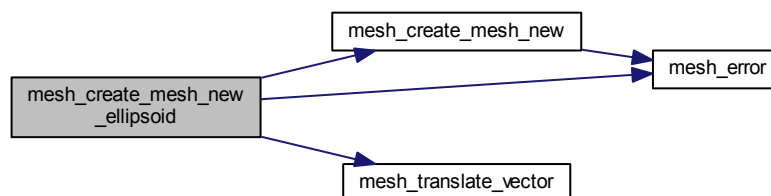
Parameters

in	<i>sz</i>	Size vector
in	<i>pos</i>	Position vector

Returns

Output mesh

Here is the call graph for this function:



5.3.2.6 void mesh_free_mesh (MESH m)

Frees a mesh.

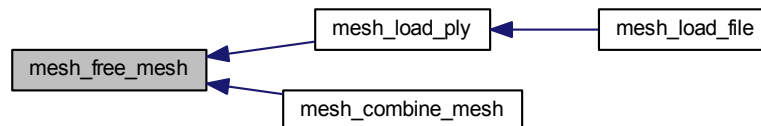
Parameters

<code>in</code>	<code>m</code>	Input mesh
-----------------	----------------	------------

Returns

NULL

Here is the caller graph for this function:

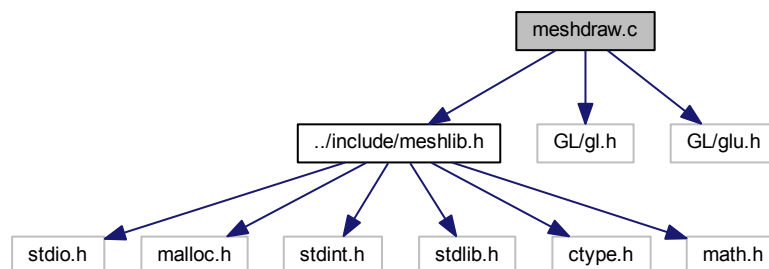


5.4 meshdraw.c File Reference

This file contains functions pertaining to mesh drawing in OpenGL.

```
#include "../include/meshlib.h"
#include <GL/gl.h>
#include <GL/glu.h>
```

Include dependency graph for meshdraw.c:



Functions

- void `mesh_draw_mesh` (MESH m)
Draws a given mesh in OpenGL context in flat shading.
- void `mesh_draw_mesh_smooth` (MESH m)
Draws a given mesh in OpenGL context in smoothing shading.
- void `mesh_draw_point_cloud` (MESH m)
Draws a given mesh in OpenGL context as pointcloud.

5.4.1 Detailed Description

This file contains functions pertaining to mesh drawing in OpenGL.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

5.4.2 Function Documentation

5.4.2.1 void mesh_draw_mesh (MESH *m*)

Draws a given mesh in OpenGL context in flat shading.

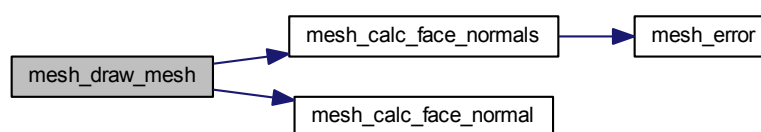
Parameters

in	<i>m</i>	Input mesh
----	----------	------------

Returns

NULL

Here is the call graph for this function:



5.4.2.2 void mesh_draw_mesh_smooth (MESH *m*)

Draws a given mesh in OpenGL context in smoothing shading.

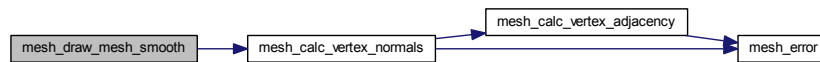
Parameters

in	<i>m</i>	Input mesh
----	----------	------------

Returns

NULL

Here is the call graph for this function:

**5.4.2.3 void mesh_draw_point_cloud (MESH m)**

Draws a given mesh in OpenGL context as pointcloud.

Parameters

in	<i>m</i>	Input mesh
----	----------	------------

Returns

NULL

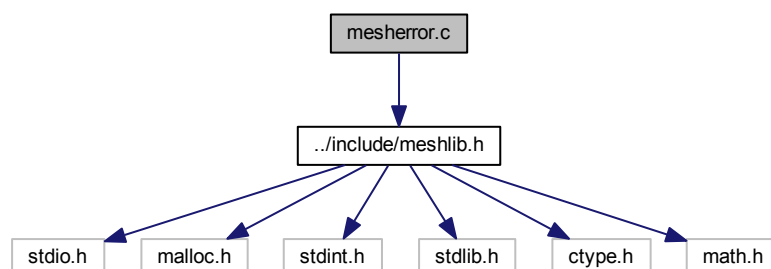
Here is the call graph for this function:

**5.5 mesherror.c File Reference**

This file contains functions pertaining to handling errors.

```
#include "../include/meshlib.h"
```

Include dependency graph for mesherror.c:



Functions

- void `mesh_error` (int type)

Displays error message and exits.

5.5.1 Detailed Description

This file contains functions pertaining to handling errors.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

5.5.2 Function Documentation

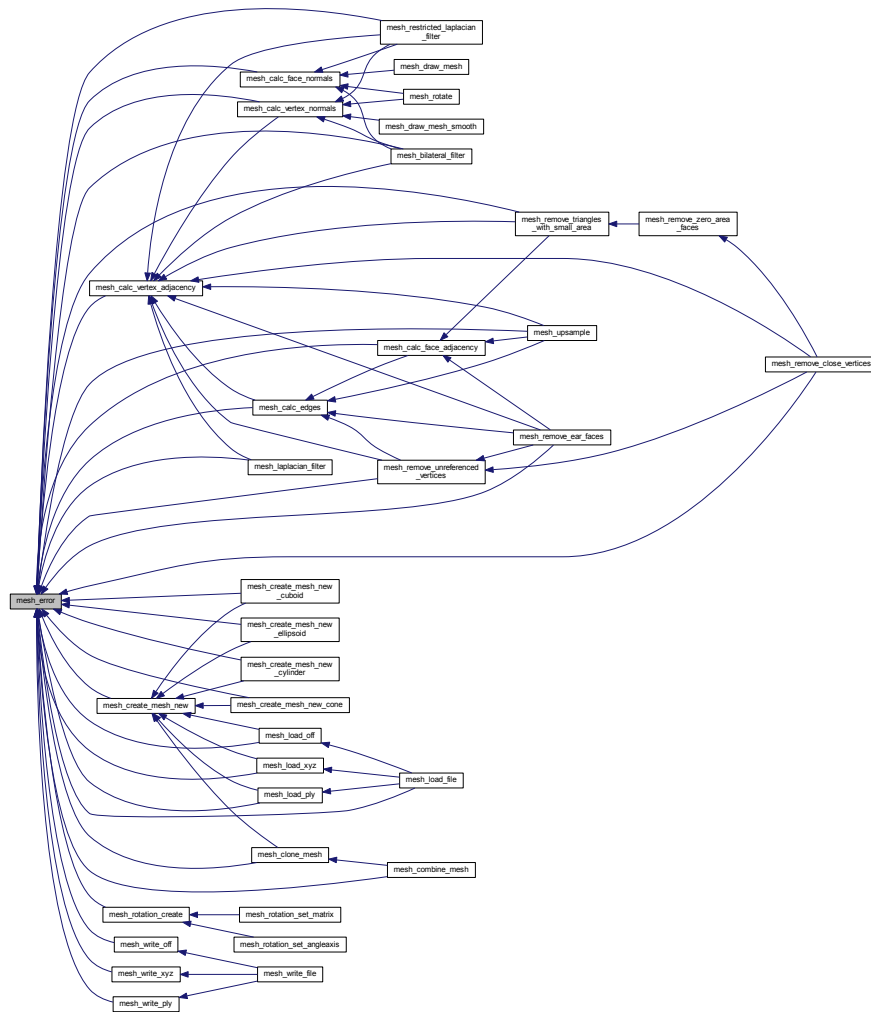
5.5.2.1 void `mesh_error` (int *type*)

Displays error message and exits.

Parameters

in	type	Error type (MESH_ERR_MALLOC/MESH_ERR_SIZE_MISMATCH/MESH_ERR_FNOTOPEN)
----	------	---

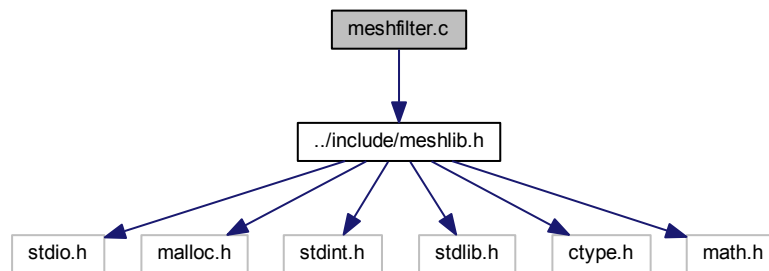
NULL



Generated on Sat, Jun 25 2016 20:54:43 for Macblib by Deryogen

```
#include "../include/meshlib.h"
```

Include dependency graph for meshfilter.c:



Functions

- int [mesh_bilateral_filter](#) (MESH m, FLOATDATA sigma_c, FLOATDATA sigma_s, int niters)
Mesh bilateral filter.
- int [mesh_laplacian_filter](#) (MESH m, FLOATDATA r)
Mesh Laplacian filter.
- int [mesh_restricted_laplacian_filter](#) (MESH m, FLOATDATA r, FLOATDATA ang)
Restricted Mesh Laplacian filter.

5.6.1 Detailed Description

This file contains functions pertaining to different mesh filtering algorithms.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

5.6.2 Function Documentation

5.6.2.1 int [mesh_bilateral_filter](#) (MESH m, FLOATDATA sigma_c, FLOATDATA sigma_s, int niters)

Mesh bilateral filter.

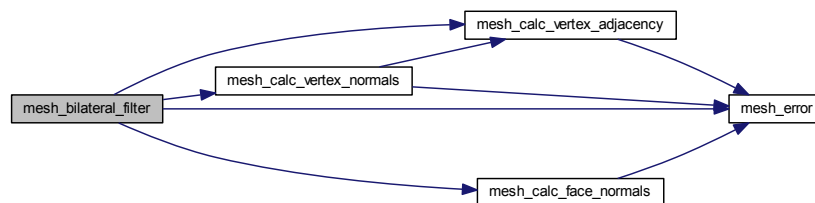
Parameters

in	<i>m</i>	Input mesh
in	<i>sigma_c</i>	Range standard deviation
in	<i>sigma_s</i>	Spatial standard deviation
in	<i>niters</i>	Number of iterations

Returns

Error code

Here is the call graph for this function:

**5.6.2.2 int mesh_laplacian_filter (MESH *m*, FLOATDATA *r*)**

Mesh Laplacian filter.

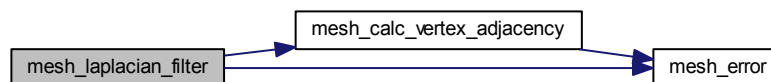
Parameters

in	<i>m</i>	Input mesh
in	<i>r</i>	Amount of diffusion

Returns

Error code

Here is the call graph for this function:

**5.6.2.3 int mesh_restricted_laplacian_filter (MESH *m*, FLOATDATA *r*, FLOATDATA *ang*)**

Restricted Mesh Laplacian filter.

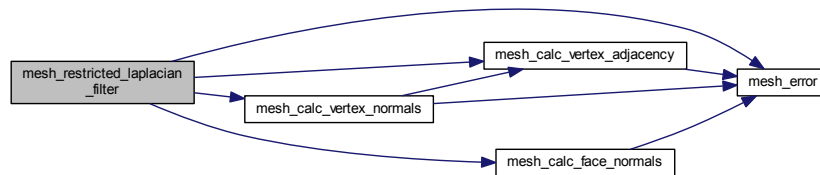
Parameters

in	<i>m</i>	Input mesh
in	<i>r</i>	Amount of diffusion
in	<i>ang</i>	Minimum angle in degrees to suppress filtering

Returns

Error code

Here is the call graph for this function:

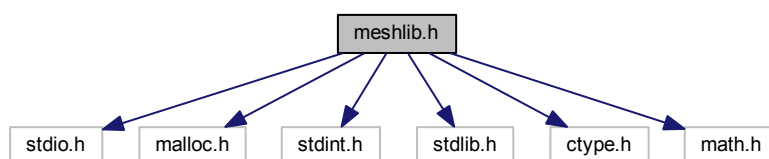


5.7 meshlib.h File Reference

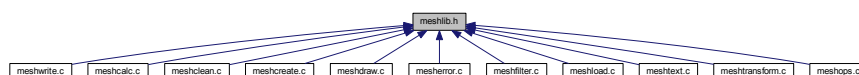
This header file contains declarations of all functions of meshlib.

```
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
#include <ctype.h>
#include <math.h>
```

Include dependency graph for meshlib.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [mesh_vector3](#)

- struct [mesh_color](#)
- struct [mesh_struct](#)
- struct [mesh_struct2](#)
- struct [mesh_struct3](#)
- struct [mesh_face](#)
- struct [mesh_edge](#)
- struct [mesh_adjface](#)
- struct [mesh_rotation](#)
- struct [mesh_transform](#)
- struct [mesh](#)

Macros

- #define [_CRT_SECURE_NO_DEPRECATED](#)
- #define [MESHLIBAPI](#) extern
- #define [MESH_INTDATA_TYPE](#) 0
- #define [MESH_FLOATDATA_TYPE](#) 1
- #define [INTDATA](#) int32_t /* do not change this, careful see meshload fscanf and other functions */
- #define [FLOATDATA](#) double /* do not change this, careful see meshload fscanf and other functions */
- #define [MESH_ORIGIN_TYPE_BUILD](#) 00
- #define [MESH_ORIGIN_TYPE_OFF](#) 11
- #define [MESH_ORIGIN_TYPE_NOFF](#) 12
- #define [MESH_ORIGIN_TYPE_COFF](#) 13
- #define [MESH_ORIGIN_TYPE_NCOFF](#) 14
- #define [MESH_ORIGIN_TYPE_XYZ](#) 20
- #define [MESH_ORIGIN_TYPE_PLY_ASCII](#) 30
- #define [MESH_ORIGIN_TYPE_PLY_BINARY_LITTLE_ENDIAN](#) 31
- #define [MESH_ORIGIN_TYPE_PLY_BINARY_BIG_ENDIAN](#) 32
- #define [MESH_ERR_MALLOC](#) 0
- #define [MESH_ERR_SIZE_MISMATCH](#) 1
- #define [MESH_ERR_FNOTOPEN](#) 2
- #define [MESH_ERR_INCOMPATIBLE](#) 3
- #define [MESH_ERR_UNKNOWN](#) 4
- #define [MESH_PI](#) (3.14159265359)
- #define [MESH_TWOPI](#) (6.28318530718)
- #define [MESH_CLONE_VERTICES](#) (0x01)
- #define [MESH_CLONE_VNORMALS](#) ([MESH_CLONE_VERTICES](#) | __MESH_CLONE_VNORMALS)
- #define [MESH_CLONE_VCOLORS](#) ([MESH_CLONE_VERTICES](#) | __MESH_CLONE_VCOLORS)
- #define [MESH_CLONE_VFACES](#) ([MESH_CLONE_VERTICES](#) | __MESH_CLONE_VFACES)
- #define [MESH_CLONE_V_ALL_PROPS](#) (0x0F)
- #define [MESH_CLONE_FACES](#) ([MESH_CLONE_VERTICES](#) | __MESH_CLONE_FACES)
- #define [MESH_CLONE_FNORMALS](#) ([MESH_CLONE_FACES](#) | __MESH_CLONE_FNORMALS)
- #define [MESH_CLONE_FCOLORS](#) ([MESH_CLONE_FACES](#) | __MESH_CLONE_FCOLORS)
- #define [MESH_CLONE_FAREAS](#) ([MESH_CLONE_FACES](#) | __MESH_CLONE_FAREAS)
- #define [MESH_CLONE_FFACES](#) ([MESH_CLONE_FACES](#) | __MESH_CLONE_FFACES)
- #define [MESH_CLONE_F_ALL_PROPS](#) ([MESH_CLONE_FACES](#) | __MESH_CLONE_F_ALL_PROPS)
- #define [MESH_CLONE_EDGES](#) ([MESH_CLONE_VERTICES](#) | __MESH_CLONE_FACES | __MESH_CLONE_EDGES)
- #define [MESH_CLONE_ALL_PROPS](#) (0xFFFF)

Typedefs

- typedef struct _iobuf * FILEPOINTER
- typedef INTDATA INTDATA2[2]
- typedef INTDATA INTDATA3[3]
- typedef struct mesh_vector3 mesh_vector3
- typedef mesh_vector3 * MESH_VECTOR3
- typedef mesh_vector3 mesh_vertex
- typedef mesh_vertex * MESH_VERTEX
- typedef mesh_vector3 mesh_normal
- typedef mesh_normal * MESH_NORMAL
- typedef struct mesh_color mesh_color
- typedef mesh_color * MESH_COLOR
- typedef struct mesh_struct mesh_struct
- typedef mesh_struct * MESH_STRUCT
- typedef struct mesh_struct2 mesh_struct2
- typedef mesh_struct2 * MESH_STRUCT2
- typedef struct mesh_struct3 mesh_struct3
- typedef mesh_struct3 * MESH_STRUCT3
- typedef struct mesh_face mesh_face
- typedef mesh_face * MESH_FACE
- typedef struct mesh_edge mesh_edge
- typedef struct mesh_edge * MESH_EDGE
- typedef struct mesh_adjface mesh_adjface
- typedef struct mesh_adjface mesh_vface
- typedef mesh_vface * MESH_VFACE
- typedef struct mesh_adjface mesh_fface
- typedef mesh_fface * MESH_FFACE
- typedef struct mesh_rotation mesh_rotation
- typedef mesh_rotation * MESH_ROTATION
- typedef struct mesh_transform mesh_transform
- typedef mesh_transform * MESH_TRANSFORM
- typedef struct mesh mesh
- typedef mesh * MESH

Functions

- MESHLIBAPI void mesh_error (int type)
Displays error message and exits.
- MESHLIBAPI MESH mesh_create_mesh_new ()
Creates a new mesh.
- MESHLIBAPI void mesh_free_mesh (MESH m)
Frees a mesh.
- MESHLIBAPI MESH mesh_create_mesh_new_cuboid (MESH_VECTOR3 sz, MESH_VECTOR3 pos)
Creates a cuboid mesh.
- MESHLIBAPI MESH mesh_create_mesh_new_ellipsoid (MESH_VECTOR3 sz, MESH_VECTOR3 pos)
Creates an ellipsoid mesh.
- MESHLIBAPI MESH mesh_create_mesh_new_cylinder (MESH_VECTOR3 sz, MESH_VECTOR3 pos)
Creates a cylinder mesh.
- MESHLIBAPI MESH mesh_create_mesh_new_cone (MESH_VECTOR3 sz, MESH_VECTOR3 pos)
Creates a cone mesh.
- MESHLIBAPI MESH mesh_clone_mesh (MESH m, uint16_t flags)
Clones a given mesh into another mesh.

- [MESHAPI MESH mesh_combine_mesh](#) ([MESH](#) m1, [MESH](#) m2)
Combines a given mesh with another given mesh.
- [MESHAPI MESH mesh_load_file](#) (const char *fname)
Reads a mesh from an OFF/PLY/ASC/XYZ file.
- [MESHAPI MESH mesh_load_off](#) (const char *fname)
Reads a mesh from an OFF file.
- [MESHAPI MESH mesh_load_xyz](#) (const char *fname)
Read a mesh from an ASC/XYZ file.
- [MESHAPI MESH mesh_load_ply](#) (const char *fname)
Reads a mesh from a PLY file.
- [MESHAPI int mesh_write_file](#) ([MESH](#) m, const char *fname)
Write a mesh to an OFF/PLY/ASC/XYZ file.
- [MESHAPI int mesh_write_off](#) ([MESH](#) m, const char *fname)
Write a mesh to an OFF file.
- [MESHAPI int mesh_write_xyz](#) ([MESH](#) m, const char *fname)
Write a mesh to an XYZ file.
- [MESHAPI int mesh_write_ply](#) ([MESH](#) m, const char *fname)
Write a mesh to an PLY file.
- [MESHAPI int mesh_calc_vertex_normals](#) ([MESH](#) m)
Computes vertex normals of a given mesh.
- [MESHAPI int mesh_calc_face_normals](#) ([MESH](#) m)
Computes face normals of a given mesh.
- [MESHAPI int mesh_calc_edges](#) ([MESH](#) m)
Computes edges of a given mesh.
- [MESHAPI int mesh_calc_vertex_adjacency](#) ([MESH](#) m)
Computes vertex adjacent faces of a given mesh.
- [MESHAPI int mesh_calc_face_adjacency](#) ([MESH](#) m)
Computes face adjacent faces of a given mesh.
- [MESHAPI int mesh_upsample](#) ([MESH](#) m, int iters)
Upsamples a given mesh.
- [MESHAPI void mesh_cross_vector3](#) ([MESH_VECTOR3](#) x, [MESH_VECTOR3](#) y, [MESH_VECTOR3](#) z)
Computes the cross product of two 3-d vectors.
- [MESHAPI void mesh_cross_normal](#) ([MESH_NORMAL](#) x, [MESH_NORMAL](#) y, [MESH_NORMAL](#) z)
Computes the normalized cross product of two normals.
- [MESHAPI FLOATDATA mesh_calc_triangle_area](#) ([MESH_VERTEX](#) a, [MESH_VERTEX](#) b, [MESH_VERTEX](#) c)
Computes area of a triangle.
- [MESHAPI void mesh_calc_face_normal](#) ([MESH_VERTEX](#) v1, [MESH_VERTEX](#) v2, [MESH_VERTEX](#) v3, [MESH_NORMAL](#) n)
Computes the face normal given 3 vertices.
- [MESHAPI INTDATA mesh_find](#) ([MESH_STRUCT](#) s, [INTDATA](#) q)
Finds an item in an INTDATA structure.
- [MESHAPI INTDATA mesh_find2](#) ([MESH_STRUCT2](#) s, [INTDATA](#) q)
Finds an item in an INTDATA2 structure.
- [MESHAPI INTDATA mesh_find3](#) ([MESH_STRUCT3](#) s, [INTDATA](#) q)
Finds an item in an INTDATA3 structure.
- [MESHAPI int mesh_remove_boundary_vertices](#) ([MESH](#) m, int iters)
Removes boundary vertices and connecting elements.
- [MESHAPI int mesh_remove_boundary_faces](#) ([MESH](#) m, int iters)
Removes boundary faces and connecting elements.
- [MESHAPI int mesh_remove_triangles_with_small_area](#) ([MESH](#) m, [FLOATDATA](#) area)

- Removes triangles with area smaller than a given value.*

 - MESHLIBAPI int [mesh_remove_unreferenced_vertices](#) (MESH m)

Removes unreferenced vertices.
- MESHLIBAPI int [mesh_remove_zero_area_faces](#) (MESH m)

Removes triangles with zero area.
- MESHLIBAPI int [mesh_remove_close_vertices](#) (MESH m, FLOATDATA r)

Removes close vertices.
- MESHLIBAPI int [mesh_remove_ear_faces](#) (MESH m, int niters)

Removes ear faces and connecting vertices.
- MESHLIBAPI int [mesh_isnumeric](#) (FILEPOINTER fp)

Checks if numeric or not.
- MESHLIBAPI int [mesh_go_next_word](#) (FILEPOINTER fp)

Points to the next word.
- MESHLIBAPI int [mesh_read_word](#) (FILEPOINTER fp, char *c_word, int sz)

Reads current word and moves to the next word.
- MESHLIBAPI int [mesh_read_word_only](#) (FILEPOINTER fp, char *c_word, int sz)

Reads current word without moving to the next word.
- MESHLIBAPI int [mesh_count_words_in_line](#) (FILEPOINTER fp, int *count)

Counts number of words in the current line.
- MESHLIBAPI int [mesh_skip_line](#) (FILEPOINTER fp)

Skips to next line.
- MESHLIBAPI int [mesh_bilateral_filter](#) (MESH m, FLOATDATA sigma_c, FLOATDATA sigma_s, int niters)

Mesh bilateral filter.
- MESHLIBAPI int [mesh_laplacian_filter](#) (MESH m, FLOATDATA r)

Mesh Laplacian filter.
- MESHLIBAPI int [mesh_restricted_laplacian_filter](#) (MESH m, FLOATDATA r, FLOATDATA ang)

Restricted Mesh Laplacian filter.
- MESHLIBAPI MESH_ROTATION [mesh_rotation_create](#) ()

Creates a new rotation.
- MESHLIBAPI void [mesh_rotation_free](#) (MESH_ROTATION r)

Frees a given rotation.
- MESHLIBAPI MESH_ROTATION [mesh_rotation_set_matrix](#) (FLOATDATA *mat, MESH_ROTATION r)

Sets rotation from a matrix.
- MESHLIBAPI MESH_ROTATION [mesh_rotation_set_angleaxis](#) (FLOATDATA ang, MESH_NORMAL axis, MESH_ROTATION r)

Sets rotation from angle axis.
- MESHLIBAPI int [mesh_translate](#) (MESH m, FLOATDATA x, FLOATDATA y, FLOATDATA z)

Translates a mesh by x, y and z amounts.
- MESHLIBAPI int [mesh_translate_vector](#) (MESH m, MESH_VERTEX v)

Translates a mesh by a given 3-d vector.
- MESHLIBAPI int [mesh_scale](#) (MESH m, FLOATDATA sx, FLOATDATA sy, FLOATDATA sz)

Scales a mesh by x, y and z amounts.
- MESHLIBAPI MESH_VERTEX [mesh_vertex_rotate](#) (MESH_VERTEX v, MESH_ROTATION r)

Rotates a vertex by a given rotation.
- MESHLIBAPI int [mesh_rotate](#) (MESH m, MESH_ROTATION r)

Rotates a mesh by a given rotation.
- MESHLIBAPI void [mesh_draw_mesh](#) (MESH m)

Draws a given mesh in OpenGL context in flat shading.
- MESHLIBAPI void [mesh_draw_mesh_smooth](#) (MESH m)

Draws a given mesh in OpenGL context in smoothing shading.
- MESHLIBAPI void [mesh_draw_point_cloud](#) (MESH m)

Draws a given mesh in OpenGL context as pointcloud.

5.7.1 Detailed Description

This header file contains declarations of all functions of meshlib.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

5.7.2 Macro Definition Documentation

5.7.2.1 `#define _CRT_SECURE_NO_DEPRECATED`

5.7.2.2 `#define FLOATDATA double /* do not change this, careful see meshload fscanf and other functions */`

Float datatype

5.7.2.3 `#define INTDATA int32_t /* do not change this, careful see meshload fscanf and other functions */`

Integer datatype

5.7.2.4 `#define MESH_CLONE_ALL_PROPS (0xFFFF)`

Clone mesh all properties

5.7.2.5 `#define MESH_CLONE_EDGES (MESH_CLONE_VERTICES | __MESH_CLONE_FACES | __MESH_CLONE_EDGES)`

Clone mesh edges

5.7.2.6 `#define MESH_CLONE_F_ALL_PROPS (MESH_CLONE_FACES | __MESH_CLONE_F_ALL_PROPS)`

Clone mesh all face properties

5.7.2.7 `#define MESH_CLONE_FACES (MESH_CLONE_VERTICES | __MESH_CLONE_FACES)`

Clone mesh faces

5.7.2.8 `#define MESH_CLONE_FAREAS (MESH_CLONE_FACES | __MESH_CLONE_FAREAS)`

Clone mesh faces and face areas

5.7.2.9 `#define MESH_CLONE_FCOLORS (MESH_CLONE_FACES | __MESH_CLONE_FCOLORS)`

Clone mesh faces and face colors

5.7.2.10 `#define MESH_CLONE_FFACES (MESH_CLONE_FACES | __MESH_CLONE_FFACES)`

Clone mesh faces and face face adjacency

5.7.2.11 `#define MESH_CLONE_FNORMALS (MESH_CLONE_FACES | __MESH_CLONE_FNORMALS)`

Clone mesh faces and face normals

5.7.2.12 `#define MESH_CLONE_V_ALL_PROPS (0x0F)`

Clone mesh all vertex properties

5.7.2.13 `#define MESH_CLONE_VCOLORS (MESH_CLONE_VERTICES | __MESH_CLONE_VCOLORS)`

Clone mesh vertices and vertex colors

5.7.2.14 `#define MESH_CLONE_VERTICES (0x01)`

Clone mesh vertices

5.7.2.15 `#define MESH_CLONE_VFACES (MESH_CLONE_VERTICES | __MESH_CLONE_VFACES)`

Clone mesh vertices and vertex face adjacency

5.7.2.16 `#define MESH_CLONE_VNORMALS (MESH_CLONE_VERTICES | __MESH_CLONE_VNORMALS)`

Clone mesh vertices and vertex normals

5.7.2.17 `#define MESH_ERR_FNOTOPEN 2`

Mesh error type - file open

5.7.2.18 `#define MESH_ERR_INCOMPATIBLE 3`

Mesh error type - incompatible data

5.7.2.19 `#define MESH_ERR_MALLOC 0`

Mesh error type - allocation

5.7.2.20 `#define MESH_ERR_SIZE_MISMATCH 1`

Mesh error type - size mismatch

5.7.2.21 `#define MESH_ERR_UNKNOWN 4`

Mesh error type - unknown

5.7.2.22 #define MESH_FLOATDATA_TYPE 1

Float datatype selector

5.7.2.23 #define MESH_INTDATA_TYPE 0

Integer datatype selector

5.7.2.24 #define MESH_ORIGIN_TYPE_BUILD 00

Mesh origin type - create new

5.7.2.25 #define MESH_ORIGIN_TYPE_COFF 13

Mesh origin type - COFF file

5.7.2.26 #define MESH_ORIGIN_TYPE_NCOFF 14

Mesh origin type - NCOFF file

5.7.2.27 #define MESH_ORIGIN_TYPE_NOFF 12

Mesh origin type - NOFF file

5.7.2.28 #define MESH_ORIGIN_TYPE_OFF 11

Mesh origin type - OFF file

5.7.2.29 #define MESH_ORIGIN_TYPE_PLY_ASCII 30

Mesh origin type - PLY ascii file

5.7.2.30 #define MESH_ORIGIN_TYPE_PLY_BINARY_BIG_ENDIAN 32

Mesh origin type - PLY binary BE file

5.7.2.31 #define MESH_ORIGIN_TYPE_PLY_BINARY_LITTLE_ENDIAN 31

Mesh origin type - PLY binary LE file

5.7.2.32 #define MESH_ORIGIN_TYPE_XYZ 20

Mesh origin type - XYZ file

5.7.2.33 #define MESH_PI (3.14159265359)

π

5.7.2.34 `#define MESH_TWOPI (6.28318530718)`

2π

5.7.2.35 `#define MESHLIBAPI extern`

5.7.3 Typedef Documentation

5.7.3.1 `typedef struct _iobuf* FILEPOINTER`

File pointer

5.7.3.2 `typedef INTDATA INTDATA2[2]`

2- element INTDATA

5.7.3.3 `typedef INTDATA INTDATA3[3]`

3- element INTDATA

5.7.3.4 `typedef struct mesh mesh`

Mesh

5.7.3.5 `typedef mesh* MESH`

Pointer to mesh

5.7.3.6 `typedef struct mesh_adjface mesh_adjface`

Adjacent face structure

5.7.3.7 `typedef struct mesh_color mesh_color`

5.7.3.8 `typedef mesh_color* MESH_COLOR`

Color

5.7.3.9 `typedef struct mesh_edge mesh_edge`

Edge

5.7.3.10 `typedef struct mesh_edge* MESH_EDGE`

Pointer to edge

5.7.3.11 `typedef struct mesh_face mesh_face`

Face

5.7.3.12 `typedef mesh_face* MESH_FACE`

Pointer to face

5.7.3.13 `typedef struct mesh_adjface mesh_fface`

Face adjacent faces

5.7.3.14 `typedef mesh_fface* MESH_FFACE`

Pointer to face adjacent faces

5.7.3.15 `typedef mesh_vector3 mesh_normal`

Normal

5.7.3.16 `typedef mesh_normal* MESH_NORMAL`

Normal pointer

5.7.3.17 `typedef struct mesh_rotation mesh_rotation`

Rotation

5.7.3.18 `typedef mesh_rotation* MESH_ROTATION`

Pointer to rotation

5.7.3.19 `typedef struct mesh_struct mesh_struct`

INTDATA Structure

5.7.3.20 `typedef mesh_struct* MESH_STRUCT`

INTDATA Structure pointer

5.7.3.21 `typedef struct mesh_struct2 mesh_struct2`

INTDATA2 Structure

5.7.3.22 `typedef mesh_struct2* MESH_STRUCT2`

INTDATA2 Structure pointer

5.7.3.23 `typedef struct mesh_struct3 mesh_struct3`

INTDATA3 Structure

5.7.3.24 typedef mesh_struct3* MESH_STRUCT3

INTDATA3 Structure pointer

5.7.3.25 typedef struct mesh_transform mesh_transform

Transformation

5.7.3.26 typedef mesh_transform* MESH_TRANSFORM

Pointer to transformation

5.7.3.27 typedef struct mesh_vector3 mesh_vector3

Generic 3-d vector

5.7.3.28 typedef mesh_vector3* MESH_VECTOR3

Generic 3-d vector pointer

5.7.3.29 typedef mesh_vector3 mesh_vertex

Vertex

5.7.3.30 typedef mesh_vertex* MESH_VERTEX

Vertex pointer

5.7.3.31 typedef struct mesh_adjface mesh_vface

Vertex adjacent faces

5.7.3.32 typedef mesh_vface* MESH_VFACE

Pointer to vertex adjacent faces

5.7.4 Function Documentation**5.7.4.1 MESHAPI int mesh_bilateral_filter (MESH *m*, FLOATDATA *sigma_c*, FLOATDATA *sigma_s*, int *niters*)**

Mesh bilateral filter.

Parameters

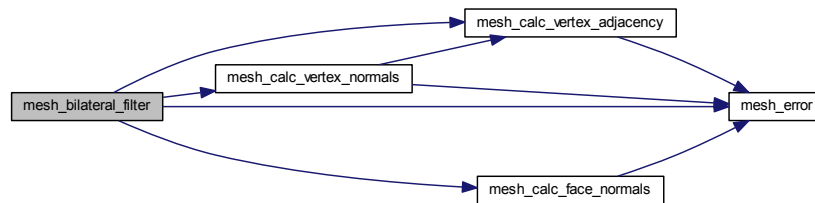
in	<i>m</i>	Input mesh
in	<i>sigma_c</i>	Range standard deviation
in	<i>sigma_s</i>	Spatial standard deviation

<i>in</i>	<i>niters</i>	Number of iterations
-----------	---------------	----------------------

Returns

Error code

Here is the call graph for this function:



5.7.4.2 MESHLIBAPI int mesh_calc_edges (MESH *m*)

Computes edges of a given mesh.

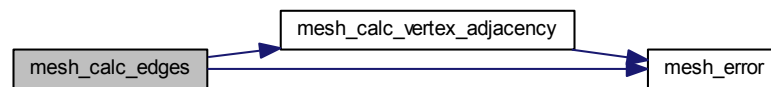
Parameters

<i>in</i>	<i>m</i>	Input mesh
-----------	----------	------------

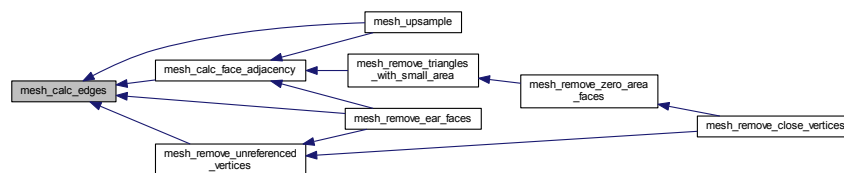
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.3 MESHAPI int mesh_calc_face_adjacency (MESH *m*)

Computes face adjacent faces of a given mesh.

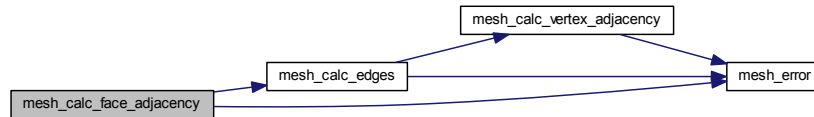
Parameters

in	<i>m</i>	Input mesh
----	----------	------------

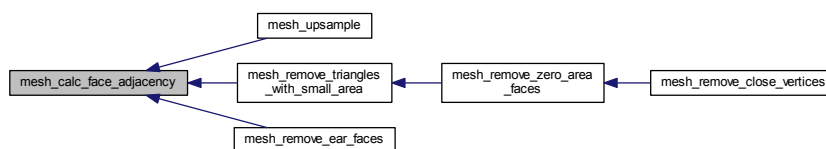
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.4 MESHAPI void mesh_calc_face_normal (MESH_VERTEX *v1*, MESH_VERTEX *v2*, MESH_VERTEX *v3*, MESH_NORMAL *n*)

Computes the face normal given 3 vertices.

Parameters

in	<i>v1</i>	First vertex
in	<i>v2</i>	Second vertex
in	<i>v3</i>	Third vertex
out	<i>n</i>	Output face normal \mathbf{n}_f

Returns

NULL

5.7.4.5 MESHAPI int mesh_calc_face_normals (MESH *m*)

Computes face normals of a given mesh.

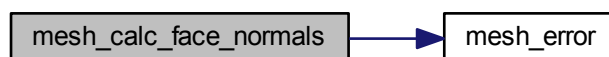
Parameters

<i>in</i>	<i>m</i>	Input mesh
-----------	----------	------------

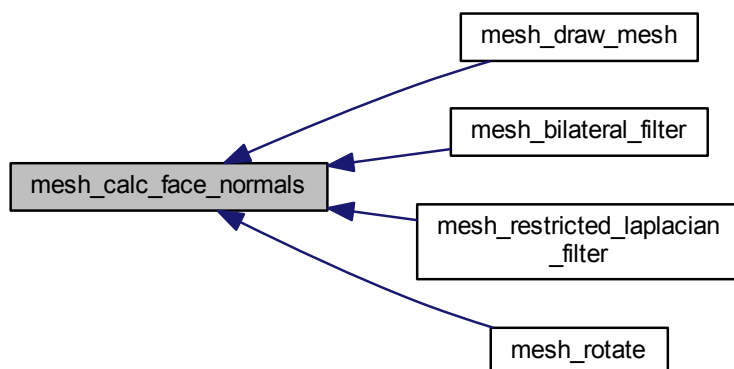
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.6 MESHAPI_FLOATDATA mesh_calc_triangle_area (MESH_VERTEX *a*, MESH_VERTEX *b*, MESH_VERTEX *c*)

Computes area of a triangle.

Parameters

<i>in</i>	<i>a</i>	First vertex
<i>in</i>	<i>b</i>	Second vertex
<i>in</i>	<i>c</i>	Third vertex

Returns

Area

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.7 MESHAPI int mesh_calc_vertex_adjacency (MESH *m*)

Computes vertex adjacent faces of a given mesh.

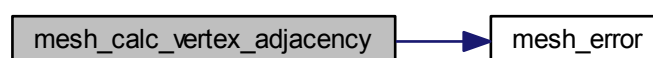
Parameters

<code>in</code>	<code>m</code>	Input mesh
-----------------	----------------	------------

Returns

Error code

Here is the call graph for this function:

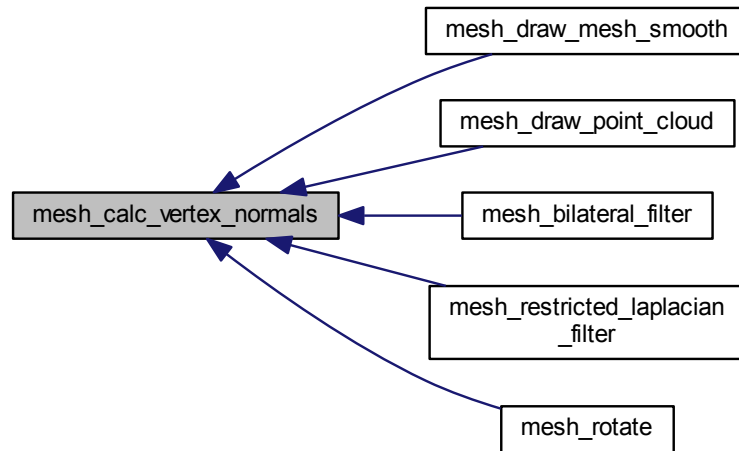


Parameters

ln

Error code

Here is the caller graph for this function:



5.7.4.9 MESHLIBAPI MESH mesh_clone_mesh (MESH *m*, uint16_t *flags*)

Clones a given mesh into another mesh.

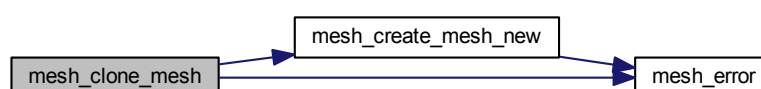
Parameters

in	<i>m</i>	Input mesh to clone
in	<i>flags</i>	Flags to copy which properties (MESH_CLONE_VERTICES/MESH_CLONE_VNORMALS/MESH_CLONE_VCOLORS/MESH_CLONE_VFACES/MESH_CLONE_VALL_PROPS/MESH_CLONE_FACES/MESH_CLONE_FNORMALS/MESH_CLONE_FCOLORS/MESH_CLONE_FAREAS/MESH_CLONE_FALL_PROPS/MESH_CLONE_ALL_PROPS)

Returns

Output cloned mesh

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.10 MESHAPI MESH mesh_combine_mesh (MESH *m1*, MESH *m2*)

Combines a given mesh with another given mesh.

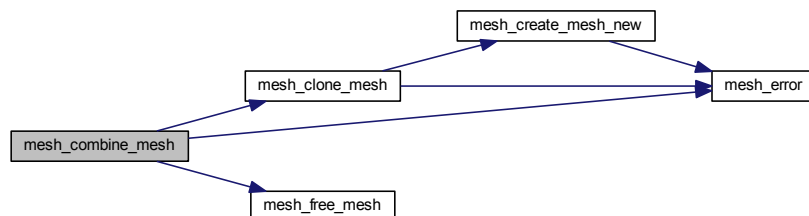
Parameters

in	<i>m1</i>	Input mesh to combine with
in	<i>m2</i>	Input mesh to combine

Returns

Output combined mesh

Here is the call graph for this function:



5.7.4.11 MESHAPI int mesh_count_words_in_line (FILEPOINTER *fp*, int * *count*)

Counts number of words in the current line.

Parameters

in	<i>fp</i>	Pointer to input file
out	<i>count</i>	Count

Returns

Status 0 - Normal/ 1- EOF

5.7.4.12 MESHAPI MESH mesh_create_mesh_new ()

Creates a new mesh.

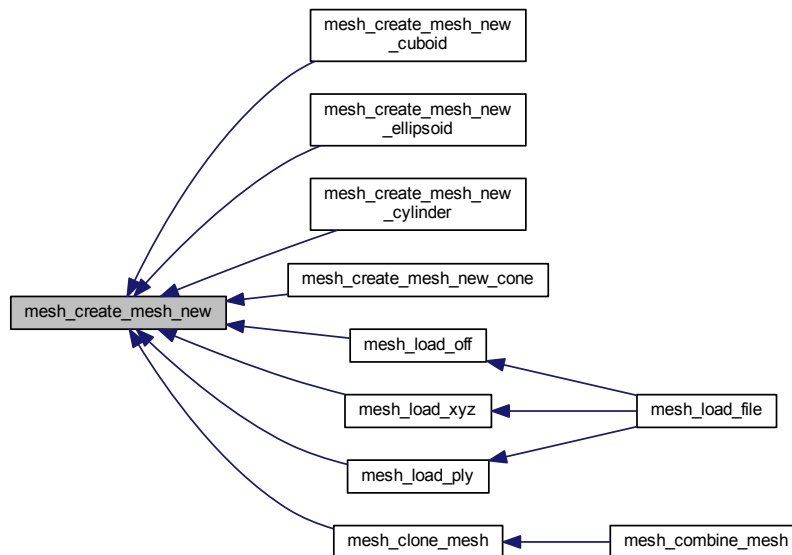
Returns

Output mesh

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.13 MESHAPI MESH mesh_create_mesh_new_cone (MESH_VECTOR3 sz, MESH_VECTOR3 pos)

Creates a cone mesh.

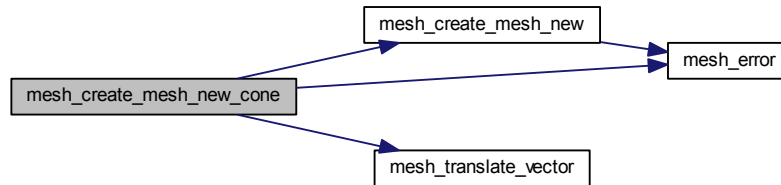
Parameters

in	sz	Size vector
in	pos	Position vector

Returns

Output mesh

Here is the call graph for this function:



5.7.4.14 MESHLIBAPI MESH mesh_create_mesh_new_cuboid (MESH_VECTOR3 sz, MESH_VECTOR3 pos)

Creates a cuboid mesh.

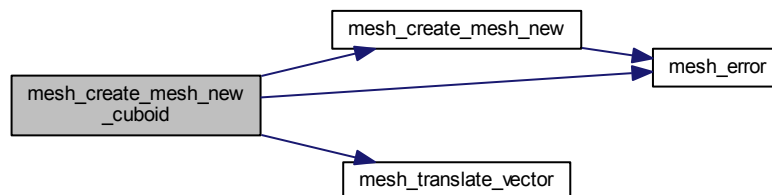
Parameters

in	sz	Size vector
in	pos	Position vector

Returns

Output mesh

Here is the call graph for this function:



5.7.4.15 MESHLIBAPI MESH mesh_create_mesh_new_cylinder (MESH_VECTOR3 sz, MESH_VECTOR3 pos)

Creates a cylinder mesh.

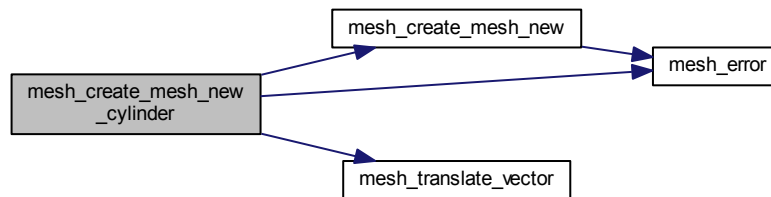
Parameters

in	<i>sz</i>	Size vector
in	<i>pos</i>	Position vector

Returns

Output mesh

Here is the call graph for this function:



5.7.4.16 MESHLIBAPI MESH mesh_create_mesh_new_ellipsoid (MESH_VECTOR3 sz, MESH_VECTOR3 pos)

Creates an ellipsoid mesh.

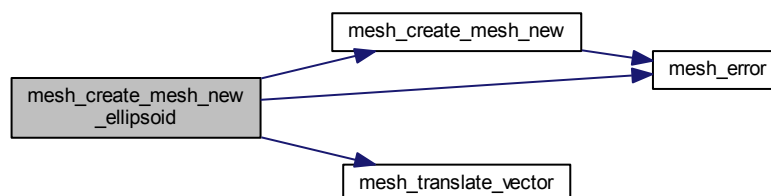
Parameters

in	<i>sz</i>	Size vector
in	<i>pos</i>	Position vector

Returns

Output mesh

Here is the call graph for this function:



5.7.4.17 MESHLIBAPI void mesh_cross_normal (MESH_NORMAL x, MESH_NORMAL y, MESH_NORMAL z)

Computes the normalized cross product of two normals.

Parameters

in	x	First normal
in	y	Second normal
out	z	Output cross product $\frac{\mathbf{x} \times \mathbf{y}}{\ \mathbf{x} \times \mathbf{y}\ _2}$

Returns

NULL

5.7.4.18 MESHLIBAPI void mesh_cross_vector3 (MESH_VECTOR3 x , MESH_VECTOR3 y , MESH_VECTOR3 z)

Computes the cross product of two 3-d vectors.

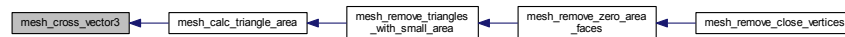
Parameters

in	x	First vector
in	y	Second vector
out	z	Output cross product $\mathbf{x} \times \mathbf{y}$

Returns

NULL

Here is the caller graph for this function:

5.7.4.19 MESHLIBAPI void mesh_draw_mesh (MESH m)

Draws a given mesh in OpenGL context in flat shading.

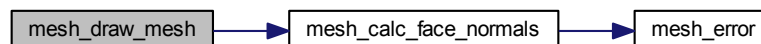
Parameters

in	m	Input mesh
----	-----	------------

Returns

NULL

Here is the call graph for this function:

5.7.4.20 MESHLIBAPI void mesh_draw_mesh_smooth (MESH m)

Draws a given mesh in OpenGL context in smoothing shading.

Parameters

<i>in</i>	<i>m</i>	Input mesh
-----------	----------	------------

Returns

NULL

Here is the call graph for this function:



5.7.4.21 MESHLIBAPI void mesh_draw_point_cloud (MESH *m*)

Draws a given mesh in OpenGL context as pointcloud.

Parameters

<i>in</i>	<i>m</i>	Input mesh
-----------	----------	------------

Returns

NULL

Here is the call graph for this function:



5.7.4.22 MESHLIBAPI void mesh_error (int *type*)

Displays error message and exits.

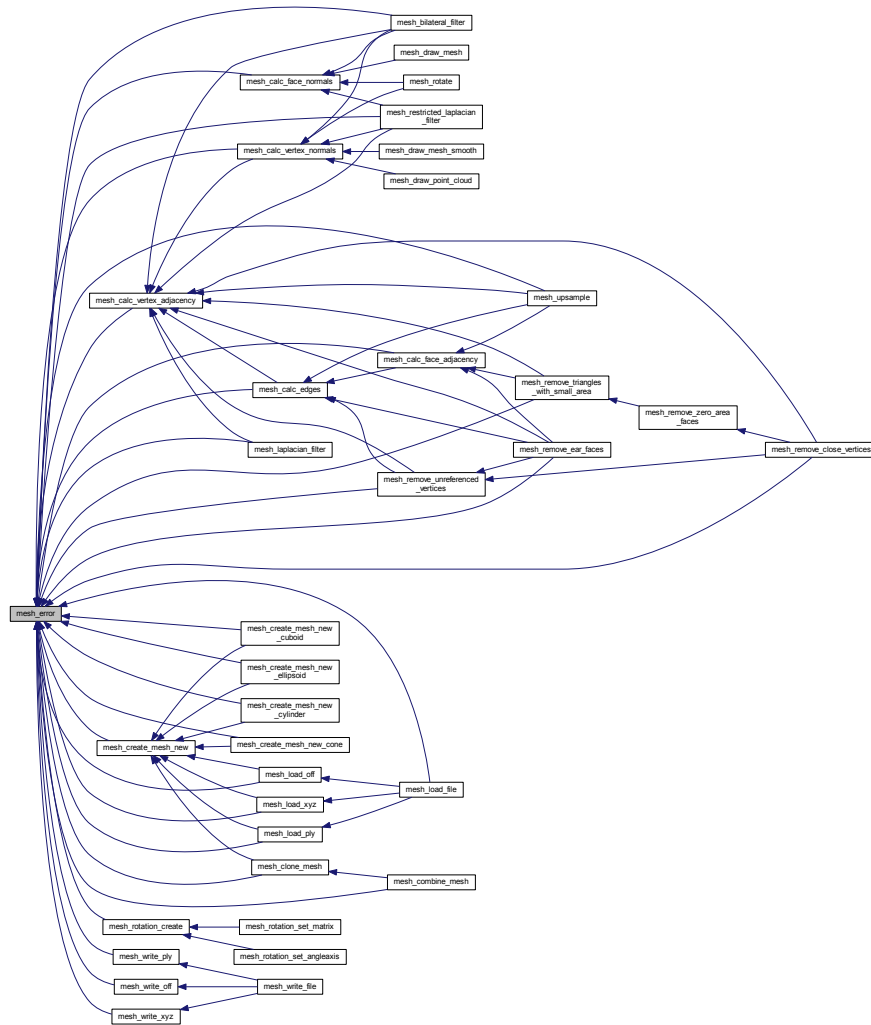
Parameters

<i>in</i>	<i>type</i>	Error type (MESH_ERR_MALLOC/MESH_ERR_SIZE_MISMATCH/MESH_ERR_FNOTOPEN)
-----------	-------------	---

Returns

NULL

Here is the caller graph for this function:



5.7.4.23 MESHAPI INTDATA mesh_find (MESH_STRUCT s, INTDATA q)

Finds an item in an INTDATA structure.

Parameters

in	s	Input INTDATA structure
in	q	Query INTDATA

Returns

Index or -1

5.7.4.24 MESHAPI INTDATA mesh_find2 (MESH_STRUCT2 s, INTDATA q)

Finds an item in an INTDATA2 structure.

Parameters

in	<i>s</i>	Input INTDATA2 structure
in	<i>q</i>	Query INTDATA2

Returns

Index or -1

5.7.4.25 MESHLIBAPI INTDATA mesh_find3 (MESH_STRUCT3 *s*, INTDATA *q*)

Finds an item in an INTDATA3 structure.

Parameters

in	<i>s</i>	Input INTDATA3 structure
in	<i>q</i>	Query INTDATA3

Returns

Index or -1

5.7.4.26 MESHLIBAPI void mesh_free_mesh (MESH *m*)

Frees a mesh.

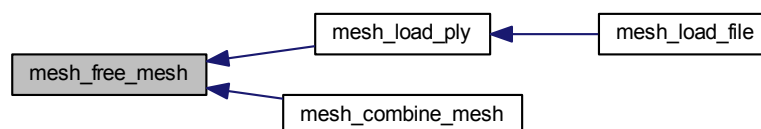
Parameters

in	<i>m</i>	Input mesh
----	----------	------------

Returns

NULL

Here is the caller graph for this function:

**5.7.4.27 MESHLIBAPI int mesh_go_next_word (FILEPOINTER *fp*)**

Points to the next word.

Parameters

<i>in</i>	<i>fp</i>	Pointer to input file
-----------	-----------	-----------------------

Returns

Status 0 - Normal/ 1- EOF

5.7.4.28 MESHLIBAPI int mesh_isnumeric (FILEPOINTER *fp*)

Checks if numeric or not.

Parameters

<i>in</i>	<i>fp</i>	Pointer to input file
-----------	-----------	-----------------------

Returns

1 for numeric/ else - for non-numeric

5.7.4.29 MESHLIBAPI int mesh_laplacian_filter (MESH *m*, FLOATDATA *r*)

Mesh Laplacian filter.

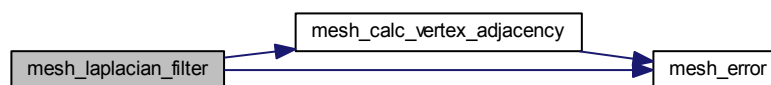
Parameters

<i>in</i>	<i>m</i>	Input mesh
<i>in</i>	<i>r</i>	Amount of diffusion

Returns

Error code

Here is the call graph for this function:

5.7.4.30 MESHLIBAPI MESH mesh_load_file (const char * *fname*)

Reads a mesh from an OFF/PLY/ASC/XYZ file.

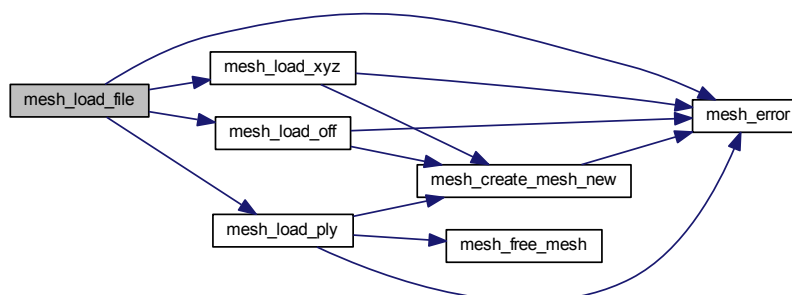
Parameters

<i>in</i>	<i>fname</i>	Input filename
-----------	--------------	----------------

Returns

Output mesh

Here is the call graph for this function:



5.7.4.31 MESHAPI MESH mesh_load_off (const char * *fname*)

Reads a mesh from an OFF file.

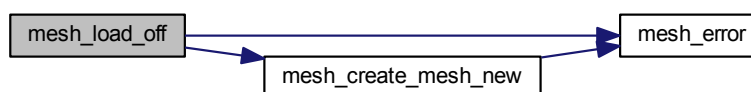
Parameters

<i>in</i>	<i>fname</i>	Input filename
-----------	--------------	----------------

Returns

Output mesh

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.32 MESHAPI MESH mesh_load_ply (const char * *fname*)

Reads a mesh from a PLY file.

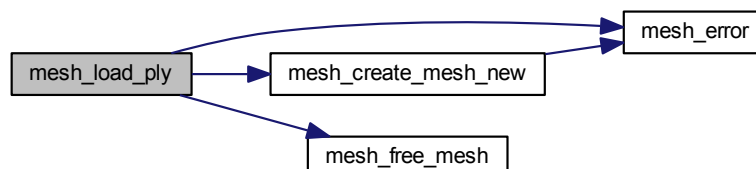
Parameters

<i>in</i>	<i>fname</i>	Input filename
-----------	--------------	----------------

Returns

Output mesh

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.33 MESHAPI MESH mesh_load_xyz (const char * *fname*)

Read a mesh from an ASC/XYZ file.

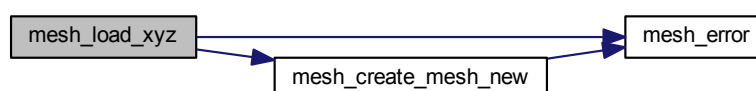
Parameters

<i>in</i>	<i>fname</i>	Input filename
-----------	--------------	----------------

Returns

Output mesh

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.34 MESHLIBAPI int mesh_read_word (FILEPOINTER *fp*, char * *c_word*, int *sz*)

Reads current word and moves to the next word.

Parameters

in	<i>fp</i>	Pointer to input file
out	<i>c_word</i>	Variable to store the word
in	<i>sz</i>	Maximum size to read

Returns

Status 0 - Normal/ 1- EOF

5.7.4.35 MESHLIBAPI int mesh_read_word_only (FILEPOINTER *fp*, char * *c_word*, int *sz*)

Reads current word without moving to the next word.

Parameters

in	<i>fp</i>	Pointer to input file
out	<i>c_word</i>	Variable to store the word
in	<i>sz</i>	Maximum size to read

Returns

Status 0 - Normal/ 1- EOF

5.7.4.36 MESHLIBAPI int mesh_remove_boundary_faces (MESH *m*, int *iters*)

Removes boundary faces and connecting elements.

Parameters

in	<i>m</i>	Input mesh
in	<i>iters</i>	Number of iterations

Returns

Error code

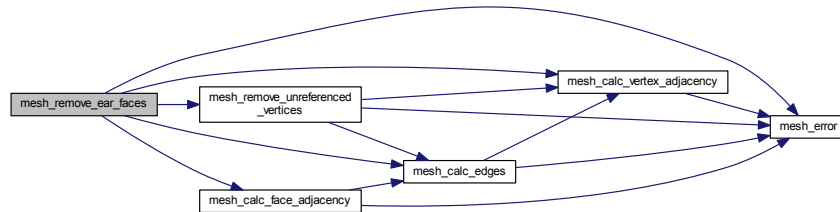
5.7.4.37 MESHLIBAPI int mesh_remove_boundary_vertices (MESH *m*, int *iters*)

Removes boundary vertices and connecting elements.

Returns

Error code

Here is the call graph for this function:



5.7.4.40 MESHLIBAPI int mesh_remove_triangles_with_small_area (MESH *m*, FLOATDATA *area*)

Removes triangles with area smaller than a given value.

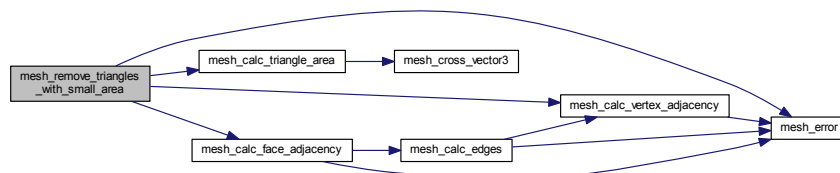
Parameters

in	<i>m</i>	Input mesh
in	<i>area</i>	Given area

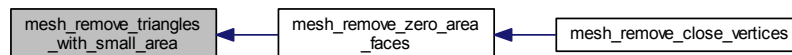
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.41 MESHLIBAPI int mesh_remove_unreferenced_vertices (MESH *m*)

Removes unreferenced vertices.

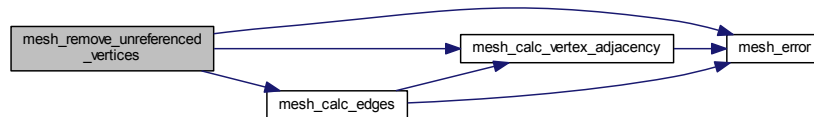
Parameters

in	<i>m</i>	Input mesh
----	----------	------------

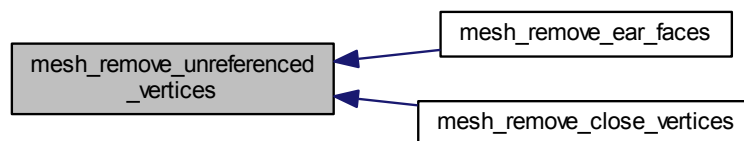
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.42 MESHLIBAPI int mesh_remove_zero_area_faces (MESH *m*)

Removes triangles with zero area.

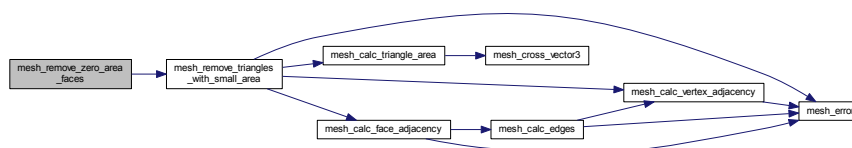
Parameters

in	<i>m</i>	Input mesh
----	----------	------------

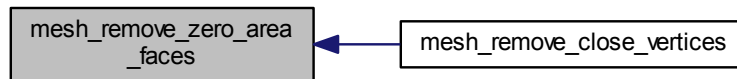
Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.43 MESHLIBAPI int mesh_restricted_laplacian_filter (MESH *m*, FLOATDATA *r*, FLOATDATA *ang*)

Restricted Mesh Laplacian filter.

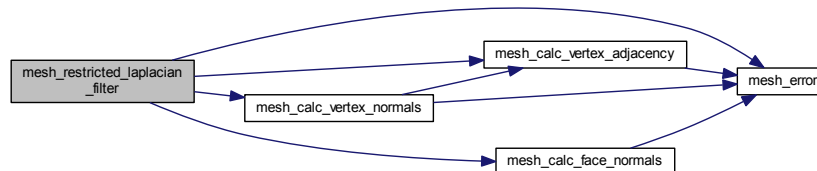
Parameters

in	<i>m</i>	Input mesh
in	<i>r</i>	Amount of diffusion
in	<i>ang</i>	Minimum angle in degrees to suppress filtering

Returns

Error code

Here is the call graph for this function:



5.7.4.44 MESHLIBAPI int mesh_rotate (MESH *m*, MESH_ROTATION *r*)

Rotates a mesh by a given rotation.

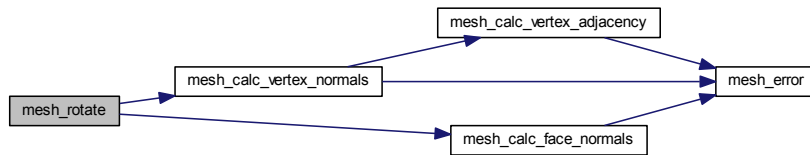
Parameters

in	<i>m</i>	Input vertex
in	<i>r</i>	Input rotation

Returns

Error code

Here is the call graph for this function:



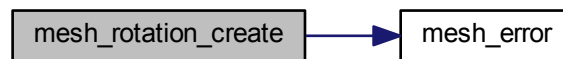
5.7.4.45 MESHLIBAPI MESH_ROTATION mesh_rotation_create ()

Creates a new rotation.

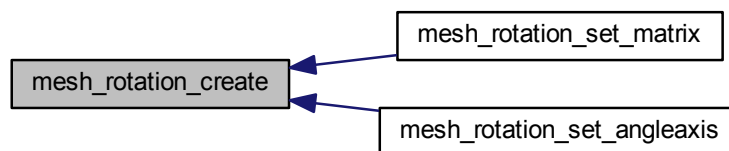
Returns

Output rotation

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.46 MESHLIBAPI void mesh_rotation_free (MESH_ROTATION r)

Frees a given rotation.

Parameters

<i>r</i>	Input rotation
----------	----------------

Returns

NULL

5.7.4.47 MESHAPI MESH_ROTATION mesh_rotation_set_angleaxis (FLOATDATA *ang*, MESH_NORMAL *axis*, MESH_ROTATION *r*)

Sets rotation from angle axis.

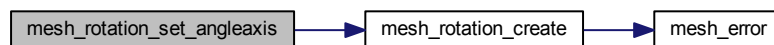
Parameters

in	<i>ang</i>	Input angle of rotation
out	<i>axis</i>	Input axis of rotation
out	<i>r</i>	Input rotation

Returns

Output rotation

Here is the call graph for this function:



5.7.4.48 MESHAPI MESH_ROTATION mesh_rotation_set_matrix (FLOATDATA * *mat*, MESH_ROTATION *r*)

Sets rotation from a matrix.

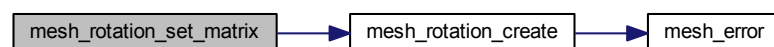
Parameters

in	<i>mat</i>	Input matrix
out	<i>r</i>	Input rotation

Returns

Output rotation

Here is the call graph for this function:



5.7.4.49 MESHAPI `int mesh_scale (MESH m, FLOATDATA sx, FLOATDATA sy, FLOATDATA sz)`

Scales a mesh by x, y and z amounts.

Parameters

<i>in</i>	<i>m</i>	Input mesh
<i>in</i>	<i>sx</i>	X component
<i>in</i>	<i>sy</i>	Y component
<i>in</i>	<i>sz</i>	Z component

Returns

Error code

5.7.4.50 MESHAPI int mesh_skip_line (FILEPOINTER *fp*)

Skips to next line.

Parameters

<i>in</i>	<i>fp</i>	Pointer to input file
-----------	-----------	-----------------------

Returns

Status 0 - Normal/ 1- EOF

5.7.4.51 MESHAPI int mesh_translate (MESH *m*, FLOATDATA *x*, FLOATDATA *y*, FLOATDATA *z*)

Translates a mesh by x, y and z amounts.

Parameters

<i>in</i>	<i>m</i>	Input mesh
<i>in</i>	<i>x</i>	X component
<i>in</i>	<i>y</i>	Y component
<i>in</i>	<i>z</i>	Z component

Returns

Error code

5.7.4.52 MESHAPI int mesh_translate_vector (MESH *m*, MESH_VECTOR3 *v*)

Translates a mesh by a given 3-d vector.

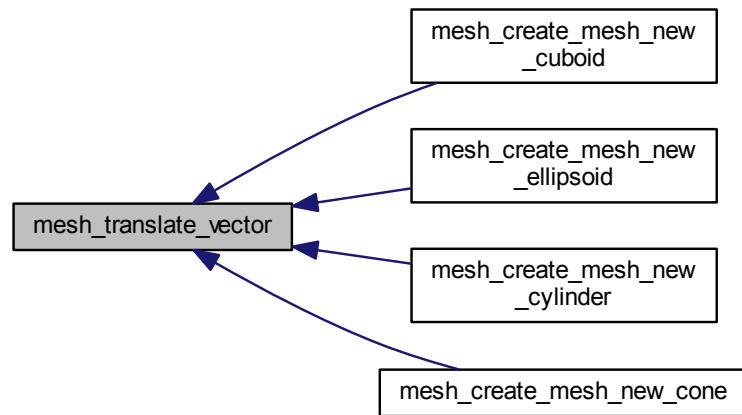
Parameters

<i>in</i>	<i>m</i>	Input mesh
<i>in</i>	<i>v</i>	Input vector

Returns

Error code

Here is the caller graph for this function:

**5.7.4.53 MESHLIBAPI int mesh_upsample (MESH *m*, int *iters*)**

Upsamples a given mesh.

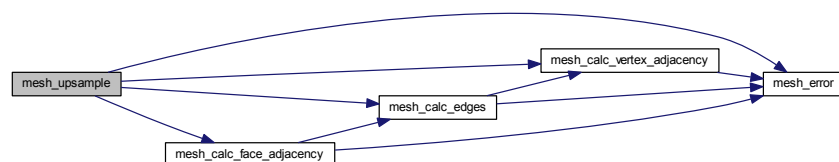
Parameters

in	<i>m</i>	Input mesh
in	<i>iters</i>	Number of iterations

Returns

Error code

Here is the call graph for this function:

**5.7.4.54 MESHLIBAPI MESH_VERTEX mesh_vertex_rotate (MESH_VERTEX *v*, MESH_ROTATION *r*)**

Rotates a vertex by a given rotation.

Parameters

in	<i>v</i>	Input vertex
in	<i>r</i>	Input rotation

Returns

Output vertex

5.7.4.55 MESHAPI int mesh_write_file (MESH *m*, const char * *fname*)

Write a mesh to an OFF/PLY/ASC/XYZ file.

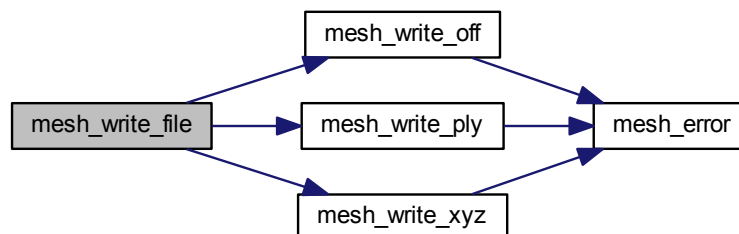
Parameters

in	<i>m</i>	Input mesh
in	<i>fname</i>	Output filename

Returns

Error code

Here is the call graph for this function:



5.7.4.56 MESHAPI int mesh_write_off (MESH *m*, const char * *fname*)

Write a mesh to an OFF file.

Parameters

in	<i>m</i>	Input mesh
in	<i>fname</i>	Output filename

Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.57 MESHAPI int mesh_write_ply (MESH *m*, const char * *fname*)

Write a mesh to an PLY file.

Parameters

in	<i>m</i>	Input mesh
in	<i>fname</i>	Output filename

Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.7.4.58 MESHLIBAPI int mesh_write_xyz (MESH *m*, const char * *fname*)

Write a mesh to an XYZ file.

Parameters

in	<i>m</i>	Input mesh
in	<i>fname</i>	Output filename

Returns

Error code

Here is the call graph for this function:



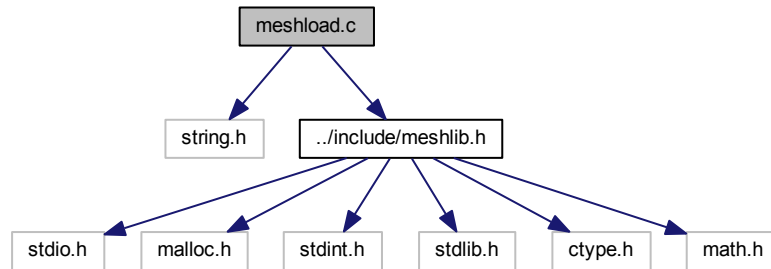
Here is the caller graph for this function:



5.8 meshload.c File Reference

This file contains functions pertaining to loading different mesh file types.

```
#include <string.h>
#include "../include/meshlib.h"
Include dependency graph for meshload.c:
```



Functions

- [MESH mesh_load_file](#) (const char *fname)
Reads a mesh from an OFF/PLY/ASC/XYZ file.
- [MESH mesh_load_off](#) (const char *fname)
Reads a mesh from an OFF file.
- [MESH mesh_load_xyz](#) (const char *fname)
Read a mesh from an ASC/XYZ file.
- [MESH mesh_load_ply](#) (const char *fname)
Reads a mesh from a PLY file.

5.8.1 Detailed Description

This file contains functions pertaining to loading different mesh file types.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

5.8.2 Function Documentation

5.8.2.1 MESH mesh_load_file (const char * fname)

Reads a mesh from an OFF/PLY/ASC/XYZ file.

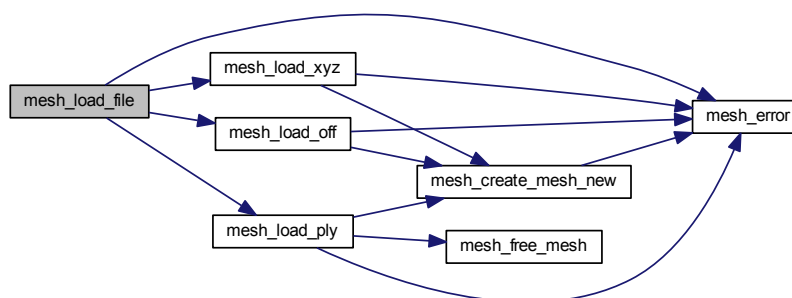
Parameters

<code>in</code>	<code>fname</code>	Input filename
-----------------	--------------------	----------------

Returns

Output mesh

Here is the call graph for this function:



5.8.2.2 MESH mesh_load_off (const char * fname)

Reads a mesh from an OFF file.

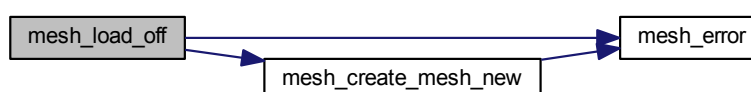
Parameters

<code>in</code>	<code>fname</code>	Input filename
-----------------	--------------------	----------------

Returns

Output mesh

Here is the call graph for this function:



Here is the caller graph for this function:



5.8.2.3 MESH mesh_load_ply (const char * *fname*)

Reads a mesh from a PLY file.

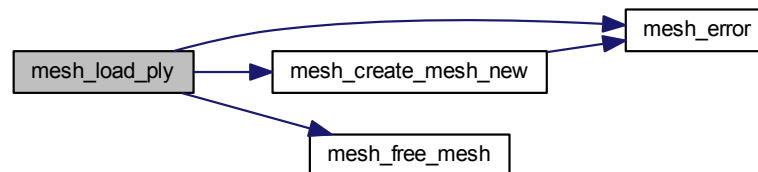
Parameters

in	<i>fname</i>	Input filename
----	--------------	----------------

Returns

Output mesh

Here is the call graph for this function:



Here is the caller graph for this function:



5.8.2.4 MESH mesh_load_xyz (const char * *fname*)

Read a mesh from an ASC/XYZ file.

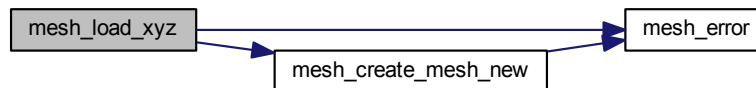
Parameters

<i>in</i>	<i>fname</i>	Input filename
-----------	--------------	----------------

Returns

Output mesh

Here is the call graph for this function:



Here is the caller graph for this function:

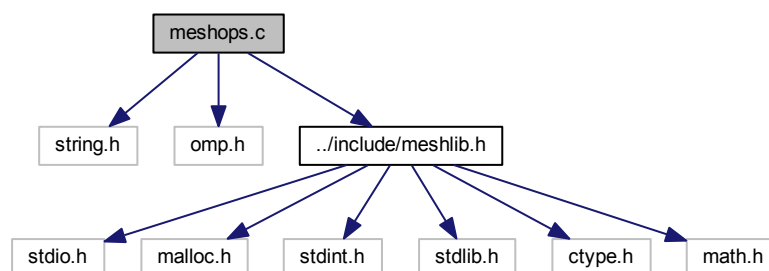


5.9 meshops.c File Reference

This file contains functions pertaining to mesh combinatorial operations.

```
#include <string.h>
#include <omp.h>
#include "../include/meshlib.h"
```

Include dependency graph for meshops.c:



Functions

- [MESH mesh_clone_mesh](#) ([MESH](#) m, [uint16_t](#) flags)
Clones a given mesh into another mesh.
- [MESH mesh_combine_mesh](#) ([MESH](#) m1, [MESH](#) m2)
Combines a given mesh with another given mesh.

5.9.1 Detailed Description

This file contains functions pertaining to mesh combinatorial operations.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

5.9.2 Function Documentation

5.9.2.1 [MESH mesh_clone_mesh](#) ([MESH](#) m, [uint16_t](#) flags)

Clones a given mesh into another mesh.

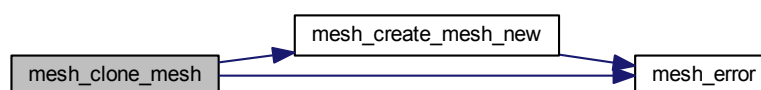
Parameters

in	<i>m</i>	Input mesh to clone
in	<i>flags</i>	Flags to copy which properties (MESH_CLONE_VERTICES/MESH_CLONE_VERTICES/MESH_CLONE_VNORMALS/MESH_CLONE_VCOLORS/MESH_CLONE_VFACES/MESH_CLONE_VSH_CLONE_V_ALL_PROPS/MESH_CLONE_FACES/MESH_CLONE_FNORMALS/MESH_CLONE_FCOLORS/MESH_CLONE_FAREAS/MESH_CLONE_F_ALL_PROPS/MESH_CLONE_ALL_PROPS)

Returns

Output cloned mesh

Here is the call graph for this function:



Here is the caller graph for this function:



5.9.2.2 MESH mesh_combine_mesh (MESH *m1*, MESH *m2*)

Combines a given mesh with another given mesh.

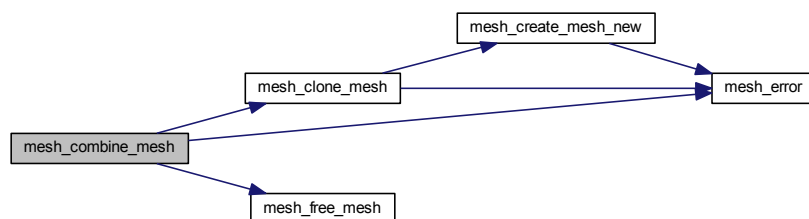
Parameters

in	<i>m1</i>	Input mesh to combine with
in	<i>m2</i>	Input mesh to combine

Returns

Output combined mesh

Here is the call graph for this function:

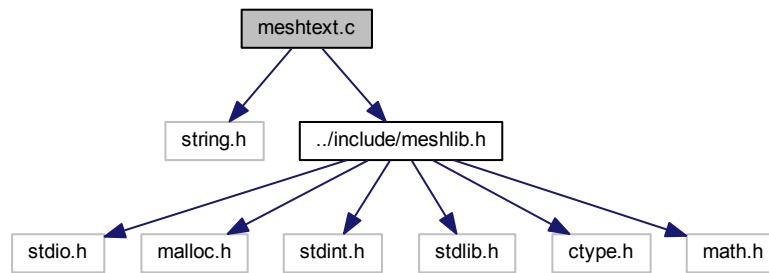


5.10 meshtext.c File Reference

This file contains functions pertaining to different text routines.

```
#include <string.h>
#include "../include/meshlib.h"
```

Include dependency graph for meshtext.c:



Functions

- int [mesh_isnumeric](#) (FILEPOINTER fp)
Checks if numeric or not.
- int [mesh_go_next_word](#) (FILEPOINTER fp)
Points to the next word.
- int [mesh_count_words_in_line](#) (FILEPOINTER fp, int *count)
Counts number of words in the current line.
- int [mesh_read_word](#) (FILEPOINTER fp, char *c_word, int sz)
Reads current word and moves to the next word.
- int [mesh_read_word_only](#) (FILEPOINTER fp, char *c_word, int sz)
Reads current word without moving to the next word.
- int [mesh_skip_line](#) (FILEPOINTER fp)
Skips to next line.

5.10.1 Detailed Description

This file contains functions pertaining to different text routines.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

5.10.2 Function Documentation

5.10.2.1 int [mesh_count_words_in_line](#) (FILEPOINTER fp, int * count)

Counts number of words in the current line.

Parameters

in	<i>fp</i>	Pointer to input file
out	<i>count</i>	Count

Returns

Status 0 - Normal/ 1- EOF

5.10.2.2 int mesh_go_next_word (FILEPOINTER *fp*)

Points to the next word.

Parameters

in	<i>fp</i>	Pointer to input file
----	-----------	-----------------------

Returns

Status 0 - Normal/ 1- EOF

5.10.2.3 int mesh_isnumeric (FILEPOINTER *fp*)

Checks if numeric or not.

Parameters

in	<i>fp</i>	Pointer to input file
----	-----------	-----------------------

Returns

1 for numeric/ else - for non-numeric

5.10.2.4 int mesh_read_word (FILEPOINTER *fp*, char * *c_word*, int *sz*)

Reads current word and moves to the next word.

Parameters

in	<i>fp</i>	Pointer to input file
out	<i>c_word</i>	Variable to store the word
in	<i>sz</i>	Maximum size to read

Returns

Status 0 - Normal/ 1- EOF

5.10.2.5 int mesh_read_word_only (FILEPOINTER *fp*, char * *c_word*, int *sz*)

Reads current word without moving to the next word.

Parameters

in	<i>fp</i>	Pointer to input file
out	<i>c_word</i>	Variable to store the word
in	<i>sz</i>	Maximum size to read

Returns

Status 0 - Normal/ 1- EOF

5.10.2.6 int mesh_skip_line (FILEPOINTER fp)

Skips to next line.

Parameters

in	<i>fp</i>	Pointer to input file
----	-----------	-----------------------

Returns

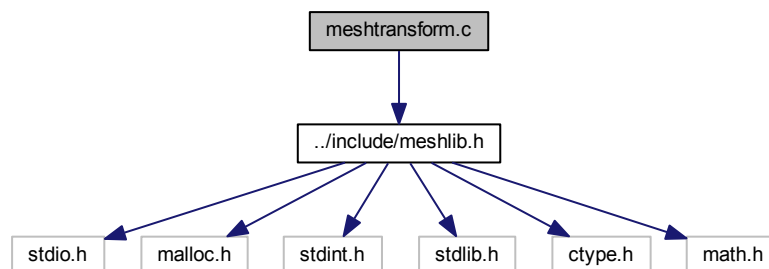
Status 0 - Normal/ 1- EOF

5.11 meshtransform.c File Reference

This file contains functions pertaining to different mesh transformations.

```
#include "../include/meshlib.h"
```

Include dependency graph for meshtransform.c:

**Functions**

- [MESH_ROTATION mesh_rotation_create \(\)](#)
Creates a new rotation.
- void [mesh_rotation_free \(MESH_ROTATION r\)](#)
Frees a given rotation.
- [MESH_ROTATION mesh_rotation_set_matrix \(FLOATDATA *mat, MESH_ROTATION r\)](#)
Sets rotation from a matrix.
- [MESH_ROTATION mesh_rotation_set_angleaxis \(FLOATDATA ang, MESH_NORMAL axis, MESH_ROTATION r\)](#)
Sets rotation from angle axis.

- int `mesh_translate` (`MESH` m, `FloatData` x, `FloatData` y, `FloatData` z)
Translates a mesh by x, y and z amounts.
- int `mesh_translate_vector` (`MESH` m, `MESH_VECTOR3` v)
Translates a mesh by a given 3-d vector.
- int `mesh_scale` (`MESH` m, `FloatData` sx, `FloatData` sy, `FloatData` sz)
Scales a mesh by x, y and z amounts.
- `MESH_VERTEX` `mesh_vertex_rotate` (`MESH_VERTEX` v, `MESH_ROTATION` r)
Rotates a vertex by a given rotation.
- int `mesh_rotate` (`MESH` m, `MESH_ROTATION` r)
Rotates a mesh by a given rotation.

5.11.1 Detailed Description

This file contains functions pertaining to different mesh transformations.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

5.11.2 Function Documentation

5.11.2.1 int `mesh_rotate` (`MESH` m, `MESH_ROTATION` r)

Rotates a mesh by a given rotation.

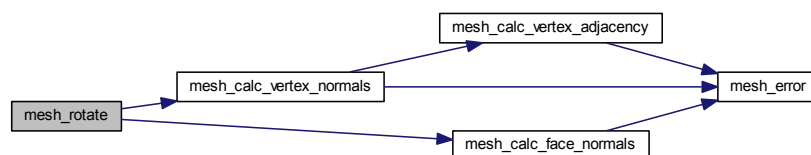
Parameters

in	m	Input vertex
in	r	Input rotation

Returns

Error code

Here is the call graph for this function:



5.11.2.2 MESH_ROTATION mesh_rotation_create ()

Creates a new rotation.

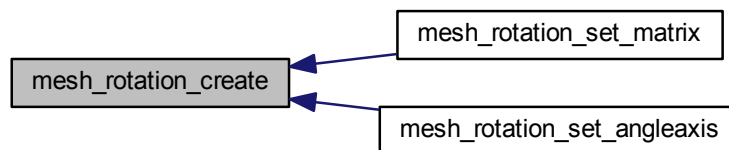
Returns

Output rotation

Here is the call graph for this function:



Here is the caller graph for this function:



5.11.2.3 void mesh_rotation_free (MESH_ROTATION r)

Frees a given rotation.

Parameters

<i>r</i>	Input rotation
----------	----------------

Returns

NULL

5.11.2.4 MESH_ROTATION mesh_rotation_set_angleaxis (FLOATDATA ang, MESH_NORMAL axis, MESH_ROTATION r)

Sets rotation from angle axis.

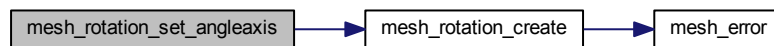
Parameters

in	<i>ang</i>	Input angle of rotation
out	<i>axis</i>	Input axis of rotation
out	<i>r</i>	Input rotation

Returns

Output rotation

Here is the call graph for this function:



5.11.2.5 MESH_ROTATION mesh_rotation_set_matrix (FLOATDATA * *mat*, MESH_ROTATION *r*)

Sets rotation from a matrix.

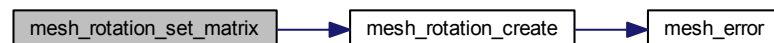
Parameters

in	<i>mat</i>	Input matrix
out	<i>r</i>	Input rotation

Returns

Output rotation

Here is the call graph for this function:



5.11.2.6 int mesh_scale (MESH *m*, FLOATDATA *sx*, FLOATDATA *sy*, FLOATDATA *sz*)

Scales a mesh by x, y and z amounts.

Parameters

in	<i>m</i>	Input mesh
in	<i>sx</i>	X component

in	sy	Y component
in	sz	Z component

Returns

Error code

5.11.2.7 int mesh_translate (MESH *m*, FLOATDATA *x*, FLOATDATA *y*, FLOATDATA *z*)

Translates a mesh by x, y and z amounts.

Parameters

in	<i>m</i>	Input mesh
in	<i>x</i>	X component
in	<i>y</i>	Y component
in	<i>z</i>	Z component

Returns

Error code

5.11.2.8 int mesh_translate_vector (MESH *m*, MESH_VECTOR3 *v*)

Translates a mesh by a given 3-d vector.

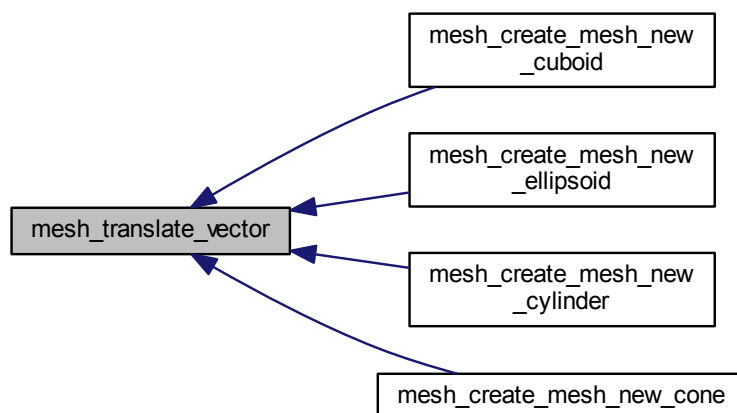
Parameters

in	<i>m</i>	Input mesh
in	<i>v</i>	Input vector

Returns

Error code

Here is the caller graph for this function:



5.11.2.9 **MESH_VERTEX** `mesh_vertex_rotate (MESH_VERTEX v, MESH_ROTATION r)`

Rotates a vertex by a given rotation.

Parameters

in	<i>v</i>	Input vertex
in	<i>r</i>	Input rotation

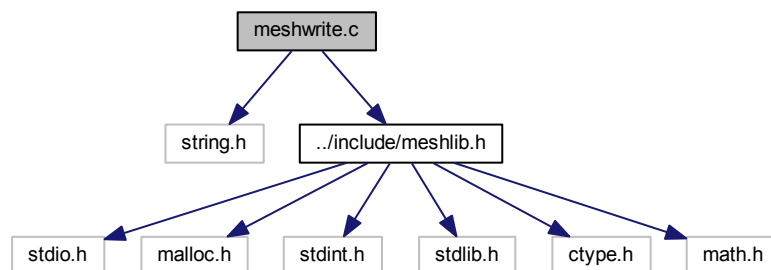
Returns

Output vertex

5.12 meshwrite.c File Reference

This file contains functions pertaining to writing different mesh file types.

```
#include <string.h>
#include "../include/meshlib.h"
Include dependency graph for meshwrite.c:
```

**Functions**

- int [mesh_write_file](#) (MESH m, const char *fname)
Write a mesh to an OFF/PLY/ASC/XYZ file.
- int [mesh_write_off](#) (MESH m, const char *fname)
Write a mesh to an OFF file.
- int [mesh_write_xyz](#) (MESH m, const char *fname)
Write a mesh to an XYZ file.
- int [mesh_write_ply](#) (MESH m, const char *fname)
Write a mesh to an PLY file.

5.12.1 Detailed Description

This file contains functions pertaining to writing different mesh file types.

Author

Sk. Mohammadul Haque

Version

1.4.1.0

Copyright

Copyright (c) 2013, 2014, 2015, 2016 Sk. Mohammadul Haque.

5.12.2 Function Documentation

5.12.2.1 int mesh_write_file (MESH *m*, const char * *fname*)

Write a mesh to an OFF/PLY/ASC/XYZ file.

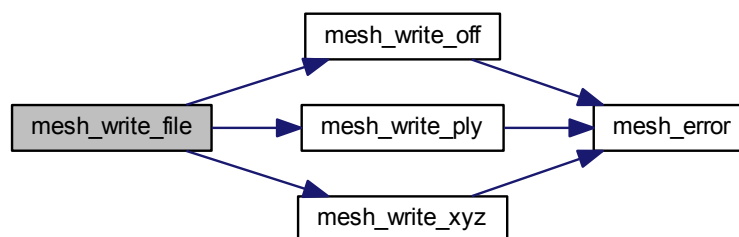
Parameters

in	<i>m</i>	Input mesh
in	<i>fname</i>	Output filename

Returns

Error code

Here is the call graph for this function:

5.12.2.2 int mesh_write_off (MESH *m*, const char * *fname*)

Write a mesh to an OFF file.

Parameters

in	<i>m</i>	Input mesh
in	<i>fname</i>	Output filename

Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.12.2.3 int mesh_write_ply (MESH *m*, const char * *fname*)

Write a mesh to an PLY file.

Parameters

in	<i>m</i>	Input mesh
in	<i>fname</i>	Output filename

Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



5.12.2.4 int mesh_write_xyz (MESH *m*, const char * *fname*)

Write a mesh to an XYZ file.

Parameters

in	<i>m</i>	Input mesh
in	<i>fname</i>	Output filename

Returns

Error code

Here is the call graph for this function:



Here is the caller graph for this function:



Index

`_CRT_SECURE_NO_DEPRECATED`
 [meshlib.h, 48](#)

a
 [mesh_color, 11](#)

b
 [mesh_color, 11](#)

data
 [mesh_rotation, 12](#)
 [mesh_transform, 14](#)

dummy
 [mesh, 8](#)

edges
 [mesh, 8](#)

FILEPOINTER
 [meshlib.h, 51](#)

FLOATDATA
 [meshlib.h, 48](#)

faces
 [mesh, 8](#)
 [mesh_adjface, 10](#)
 [mesh_edge, 11](#)

fareas
 [mesh, 8](#)

fcolors
 [mesh, 8](#)

ffaces
 [mesh, 8](#)

fnormals
 [mesh, 8](#)

g
 [mesh_color, 11](#)

INTDATA
 [meshlib.h, 48](#)

INTDATA2
 [meshlib.h, 51](#)

INTDATA3
 [meshlib.h, 51](#)

is_edges
 [mesh, 8](#)

is_faces
 [mesh, 8](#)

is_fareas
 [mesh, 8](#)

is_fcolors

[mesh, 8](#)

is_ffaces
 [mesh, 9](#)

is_fnormals
 [mesh, 9](#)

is_loaded
 [mesh, 9](#)

is_trimesh
 [mesh, 9](#)

is_vcolors
 [mesh, 9](#)

is_vertices
 [mesh, 9](#)

is_vfaces
 [mesh, 9](#)

is_vnormals
 [mesh, 9](#)

items
 [mesh_struct, 13](#)
 [mesh_struct2, 13](#)
 [mesh_struct3, 13](#)

MESH
 [meshlib.h, 51](#)
MESH_CLONE_ALL_PROPS
 [meshlib.h, 48](#)
MESH_CLONE_EDGES
 [meshlib.h, 48](#)
MESH_CLONE_F_ALL_PROPS
 [meshlib.h, 48](#)
MESH_CLONE_FACES
 [meshlib.h, 48](#)
MESH_CLONE_FAREAS
 [meshlib.h, 48](#)
MESH_CLONE_FCOLORS
 [meshlib.h, 48](#)
MESH_CLONE_FFACES
 [meshlib.h, 48](#)
MESH_CLONE_FNORMALS
 [meshlib.h, 49](#)
MESH_CLONE_V_ALL_PROPS
 [meshlib.h, 49](#)
MESH_CLONE_VCOLORS
 [meshlib.h, 49](#)
MESH_CLONE_VERTICES
 [meshlib.h, 49](#)
MESH_CLONE_VFACES
 [meshlib.h, 49](#)
MESH_CLONE_VNORMALS
 [meshlib.h, 49](#)

- MESH_COLOR
 - meshlib.h, [51](#)
- MESH_EDGE
 - meshlib.h, [51](#)
- MESH_ERR_FNOTOPEN
 - meshlib.h, [49](#)
- MESH_ERR_INCOMPATIBLE
 - meshlib.h, [49](#)
- MESH_ERR_MALLOC
 - meshlib.h, [49](#)
- MESH_ERR_SIZE_MISMATCH
 - meshlib.h, [49](#)
- MESH_ERR_UNKNOWN
 - meshlib.h, [49](#)
- MESH_FACE
 - meshlib.h, [51](#)
- MESH_FFACE
 - meshlib.h, [52](#)
- MESH_FLOATDATA_TYPE
 - meshlib.h, [49](#)
- MESH_INTDATA_TYPE
 - meshlib.h, [50](#)
- MESH_NORMAL
 - meshlib.h, [52](#)
- MESH_ORIGIN_TYPE_BUILD
 - meshlib.h, [50](#)
- MESH_ORIGIN_TYPE_COFF
 - meshlib.h, [50](#)
- MESH_ORIGIN_TYPE_NCOFF
 - meshlib.h, [50](#)
- MESH_ORIGIN_TYPE_NOFF
 - meshlib.h, [50](#)
- MESH_ORIGIN_TYPE_OFF
 - meshlib.h, [50](#)
- MESH_ORIGIN_TYPE_PLY_ASCII
 - meshlib.h, [50](#)
- MESH_ORIGIN_TYPE_PLY_BINARY_BIG_ENDIAN
 - meshlib.h, [50](#)
- MESH_ORIGIN_TYPE_PLY_BINARY_LITTLE_ENDIAN
 - meshlib.h, [50](#)
- MESH_ORIGIN_TYPE_XYZ
 - meshlib.h, [50](#)
- MESH_PI
 - meshlib.h, [50](#)
- MESH_ROTATION
 - meshlib.h, [52](#)
- MESH_STRUCT
 - meshlib.h, [52](#)
- MESH_STRUCT2
 - meshlib.h, [52](#)
- MESH_STRUCT3
 - meshlib.h, [52](#)
- MESH_TRANSFORM
 - meshlib.h, [53](#)
- MESH_TWOP
 - meshlib.h, [50](#)
- MESH_VECTOR3
 - meshlib.h, [53](#)
- MESH_VERTEX
 - meshlib.h, [53](#)
- MESH_VFACE
 - meshlib.h, [53](#)
- MESHLIBAPI
 - meshlib.h, [51](#)
- mesh, [7](#)
 - dummy, [8](#)
 - edges, [8](#)
 - faces, [8](#)
 - fareas, [8](#)
 - fcolors, [8](#)
 - ffaces, [8](#)
 - fnormals, [8](#)
 - is_edges, [8](#)
 - is_faces, [8](#)
 - is_fareas, [8](#)
 - is_fcolors, [8](#)
 - is_ffaces, [9](#)
 - is_fnormals, [9](#)
 - is_loaded, [9](#)
 - is_trimesh, [9](#)
 - is_vcolors, [9](#)
 - is_vertices, [9](#)
 - is_vfaces, [9](#)
 - is_vnormals, [9](#)
 - meshlib.h, [51](#)
 - num_edges, [9](#)
 - num_faces, [9](#)
 - num_vertices, [9](#)
 - origin_type, [9](#)
 - vcolors, [10](#)
 - vertices, [10](#)
 - vfaces, [10](#)
 - vnormals, [10](#)
- mesh_adjface, [10](#)
 - faces, [10](#)
 - meshlib.h, [51](#)
 - num_faces, [10](#)
- mesh_bilateral_filter
 - meshfilter.c, [41](#)
 - meshlib.h, [53](#)
- mesh_calc_edges
 - meshcalc.c, [18](#)
 - meshlib.h, [54](#)
- mesh_calc_face_adjacency
 - meshcalc.c, [19](#)
 - meshlib.h, [54](#)
- mesh_calc_face_normal
 - meshcalc.c, [19](#)
 - meshlib.h, [56](#)
- mesh_calc_face_normals
 - meshcalc.c, [20](#)
 - meshlib.h, [56](#)
- mesh_calc_triangle_area
 - meshcalc.c, [20](#)
 - meshlib.h, [57](#)

- mesh_calc_vertex_adjacency
 - meshcalc.c, 22
 - meshlib.h, 58
- mesh_calc_vertex_normals
 - meshcalc.c, 23
 - meshlib.h, 59
- mesh_clone_mesh
 - meshlib.h, 60
 - meshops.c, 90
- mesh_color, 10
 - a, 11
 - b, 11
 - g, 11
 - meshlib.h, 51
 - r, 11
- mesh_combine_mesh
 - meshlib.h, 61
 - meshops.c, 91
- mesh_count_words_in_line
 - meshlib.h, 61
 - meshtext.c, 92
- mesh_create_mesh_new
 - meshcreate.c, 33
 - meshlib.h, 61
- mesh_create_mesh_new_cone
 - meshcreate.c, 33
 - meshlib.h, 62
- mesh_create_mesh_new_cuboid
 - meshcreate.c, 34
 - meshlib.h, 63
- mesh_create_mesh_new_cylinder
 - meshcreate.c, 34
 - meshlib.h, 63
- mesh_create_mesh_new_ellipsoid
 - meshcreate.c, 35
 - meshlib.h, 64
- mesh_cross_normal
 - meshcalc.c, 24
 - meshlib.h, 64
- mesh_cross_vector3
 - meshcalc.c, 24
 - meshlib.h, 65
- mesh_draw_mesh
 - meshdraw.c, 37
 - meshlib.h, 65
- mesh_draw_mesh_smooth
 - meshdraw.c, 37
 - meshlib.h, 65
- mesh_draw_point_cloud
 - meshdraw.c, 38
 - meshlib.h, 66
- mesh_edge, 11
 - faces, 11
 - meshlib.h, 51
 - vertices, 11
- mesh_error
 - mesherror.c, 39
 - meshlib.h, 66
- mesh_face, 12
 - meshlib.h, 51
 - num_vertices, 12
 - vertices, 12
- mesh_fface
 - meshlib.h, 52
- mesh_find
 - meshcalc.c, 24
 - meshlib.h, 67
- mesh_find2
 - meshcalc.c, 26
 - meshlib.h, 67
- mesh_find3
 - meshcalc.c, 26
 - meshlib.h, 68
- mesh_free_mesh
 - meshcreate.c, 35
 - meshlib.h, 68
- mesh_go_next_word
 - meshlib.h, 68
 - meshtext.c, 93
- mesh_isnumeric
 - meshlib.h, 69
 - meshtext.c, 93
- mesh_laplacian_filter
 - meshfilter.c, 42
 - meshlib.h, 69
- mesh_load_file
 - meshlib.h, 69
 - meshload.c, 86
- mesh_load_off
 - meshlib.h, 70
 - meshload.c, 87
- mesh_load_ply
 - meshlib.h, 70
 - meshload.c, 88
- mesh_load_xyz
 - meshlib.h, 72
 - meshload.c, 88
- mesh_normal
 - meshlib.h, 52
- mesh_read_word
 - meshlib.h, 73
 - meshtext.c, 93
- mesh_read_word_only
 - meshlib.h, 73
 - meshtext.c, 93
- mesh_remove_boundary_faces
 - meshclean.c, 28
 - meshlib.h, 73
- mesh_remove_boundary_vertices
 - meshclean.c, 28
 - meshlib.h, 73
- mesh_remove_close_vertices
 - meshclean.c, 28
 - meshlib.h, 74
- mesh_remove_ear_faces
 - meshclean.c, 28

- meshlib.h, 74
- mesh_remove_triangles_with_small_area
 - meshclean.c, 29
 - meshlib.h, 75
- mesh_remove_unreferenced_vertices
 - meshclean.c, 29
 - meshlib.h, 75
- mesh_remove_zero_area_faces
 - meshclean.c, 31
 - meshlib.h, 76
- mesh_restricted_laplacian_filter
 - meshfilter.c, 42
 - meshlib.h, 77
- mesh_rotate
 - meshlib.h, 77
 - meshtransform.c, 95
- mesh_rotation, 12
 - data, 12
 - meshlib.h, 52
- mesh_rotation_create
 - meshlib.h, 78
 - meshtransform.c, 95
- mesh_rotation_free
 - meshlib.h, 78
 - meshtransform.c, 96
- mesh_rotation_set_angleaxis
 - meshlib.h, 79
 - meshtransform.c, 96
- mesh_rotation_set_matrix
 - meshlib.h, 79
 - meshtransform.c, 97
- mesh_scale
 - meshlib.h, 79
 - meshtransform.c, 97
- mesh_skip_line
 - meshlib.h, 81
 - meshtext.c, 94
- mesh_struct, 12
 - items, 13
 - meshlib.h, 52
 - num_items, 13
- mesh_struct2, 13
 - items, 13
 - meshlib.h, 52
 - num_items, 13
- mesh_struct3, 13
 - items, 13
 - meshlib.h, 52
 - num_items, 13
- mesh_transform, 14
 - data, 14
 - meshlib.h, 53
- mesh_translate
 - meshlib.h, 81
 - meshtransform.c, 98
- mesh_translate_vector
 - meshlib.h, 81
 - meshtransform.c, 98
- mesh_upsample
 - meshcalc.c, 26
 - meshlib.h, 82
- mesh_vector3, 14
 - meshlib.h, 53
 - x, 14
 - y, 14
 - z, 14
- mesh_vertex
 - meshlib.h, 53
- mesh_vertex_rotate
 - meshlib.h, 82
 - meshtransform.c, 98
- mesh_vface
 - meshlib.h, 53
- mesh_write_file
 - meshlib.h, 83
 - meshwrite.c, 101
- mesh_write_off
 - meshlib.h, 83
 - meshwrite.c, 101
- mesh_write_ply
 - meshlib.h, 84
 - meshwrite.c, 102
- mesh_write_xyz
 - meshlib.h, 85
 - meshwrite.c, 103
- meshcalc.c, 17
 - mesh_calc_edges, 18
 - mesh_calc_face_adjacency, 19
 - mesh_calc_face_normal, 19
 - mesh_calc_face_normals, 20
 - mesh_calc_triangle_area, 20
 - mesh_calc_vertex_adjacency, 22
 - mesh_calc_vertex_normals, 23
 - mesh_cross_normal, 24
 - mesh_cross_vector3, 24
 - mesh_find, 24
 - mesh_find2, 26
 - mesh_find3, 26
 - mesh_upsample, 26
- meshclean.c, 27
 - mesh_remove_boundary_faces, 28
 - mesh_remove_boundary_vertices, 28
 - mesh_remove_close_vertices, 28
 - mesh_remove_ear_faces, 28
 - mesh_remove_triangles_with_small_area, 29
 - mesh_remove_unreferenced_vertices, 29
 - mesh_remove_zero_area_faces, 31
- meshcreate.c, 32
 - mesh_create_mesh_new, 33
 - mesh_create_mesh_new_cone, 33
 - mesh_create_mesh_new_cuboid, 34
 - mesh_create_mesh_new_cylinder, 34
 - mesh_create_mesh_new_ellipsoid, 35
 - mesh_free_mesh, 35
- meshdraw.c, 36
 - mesh_draw_mesh, 37

- mesh_draw_mesh_smooth, 37
- mesh_draw_point_cloud, 38
- mesherror.c, 38
 - mesh_error, 39
- meshfilter.c, 40
 - mesh_bilateral_filter, 41
 - mesh_laplacian_filter, 42
 - mesh_restricted_laplacian_filter, 42
- meshlib.h, 43
 - _CRT_SECURE_NO_DEPRECATED, 48
 - FILEPOINTER, 51
 - FLOATDATA, 48
 - INTDATA, 48
 - INTDATA2, 51
 - INTDATA3, 51
 - MESH, 51
 - MESH_CLONE_ALL_PROPS, 48
 - MESH_CLONE_EDGES, 48
 - MESH_CLONE_F_ALL_PROPS, 48
 - MESH_CLONE_FACES, 48
 - MESH_CLONE_FAREAS, 48
 - MESH_CLONE_FCOLORS, 48
 - MESH_CLONE_FFACES, 48
 - MESH_CLONE_FNORMALS, 49
 - MESH_CLONE_V_ALL_PROPS, 49
 - MESH_CLONE_VCOLORS, 49
 - MESH_CLONE_VERTICES, 49
 - MESH_CLONE_VFACES, 49
 - MESH_CLONE_VNORMALS, 49
 - MESH_COLOR, 51
 - MESH_EDGE, 51
 - MESH_ERR_FNOTOPEN, 49
 - MESH_ERR_INCOMPATIBLE, 49
 - MESH_ERR_MALLOC, 49
 - MESH_ERR_SIZE_MISMATCH, 49
 - MESH_ERR_UNKNOWN, 49
 - MESH_FACE, 51
 - MESH_FFACE, 52
 - MESH_FLOATDATA_TYPE, 49
 - MESH_INTDATA_TYPE, 50
 - MESH_NORMAL, 52
 - MESH_ORIGIN_TYPE_BUILD, 50
 - MESH_ORIGIN_TYPE_COFF, 50
 - MESH_ORIGIN_TYPE_NCOFF, 50
 - MESH_ORIGIN_TYPE_NOFF, 50
 - MESH_ORIGIN_TYPE_OFF, 50
 - MESH_ORIGIN_TYPE_PLY_ASCII, 50
 - MESH_ORIGIN_TYPE_PLY_BINARY_BIG_ENDIAN, 50
 - MESH_ORIGIN_TYPE_PLY_BINARY_LITTLE_ENDIAN, 50
 - MESH_ORIGIN_TYPE_XYZ, 50
 - MESH_PI, 50
 - MESH_ROTATION, 52
 - MESH_STRUCT, 52
 - MESH_STRUCT2, 52
 - MESH_STRUCT3, 52
 - MESH_TRANSFORM, 53
 - MESH_TWOPI, 50
 - MESH_VECTOR3, 53
 - MESH_VERTEX, 53
 - MESH_VFACE, 53
 - MESHLIBAPI, 51
 - mesh, 51
 - mesh_adjface, 51
 - mesh_bilateral_filter, 53
 - mesh_calc_edges, 54
 - mesh_calc_face_adjacency, 54
 - mesh_calc_face_normal, 56
 - mesh_calc_face_normals, 56
 - mesh_calc_triangle_area, 57
 - mesh_calc_vertex_adjacency, 58
 - mesh_calc_vertex_normals, 59
 - mesh_clone_mesh, 60
 - mesh_color, 51
 - mesh_combine_mesh, 61
 - mesh_count_words_in_line, 61
 - mesh_create_mesh_new, 61
 - mesh_create_mesh_new_cone, 62
 - mesh_create_mesh_new_cuboid, 63
 - mesh_create_mesh_new_cylinder, 63
 - mesh_create_mesh_new_ellipsoid, 64
 - mesh_cross_normal, 64
 - mesh_cross_vector3, 65
 - mesh_draw_mesh, 65
 - mesh_draw_mesh_smooth, 65
 - mesh_draw_point_cloud, 66
 - mesh_edge, 51
 - mesh_error, 66
 - mesh_face, 51
 - mesh_fface, 52
 - mesh_find, 67
 - mesh_find2, 67
 - mesh_find3, 68
 - mesh_free_mesh, 68
 - mesh_go_next_word, 68
 - mesh_isnumeric, 69
 - mesh_laplacian_filter, 69
 - mesh_load_file, 69
 - mesh_load_off, 70
 - mesh_load_ply, 70
 - mesh_load_xyz, 72
 - mesh_normal, 52
 - mesh_read_word, 73
 - mesh_read_word_only, 73
 - mesh_remove_boundary_faces, 73
 - mesh_remove_boundary_vertices, 73
 - mesh_remove_close_vertices, 74
 - mesh_remove_ear_faces, 74
 - mesh_remove_triangles_with_small_area, 75
 - mesh_remove_unreferenced_vertices, 75
 - mesh_remove_zero_area_faces, 76
 - mesh_restricted_laplacian_filter, 77
 - mesh_rotate, 77
 - mesh_rotation, 52
 - mesh_rotation_create, 78

- mesh_rotation_free, 78
- mesh_rotation_set_angleaxis, 79
- mesh_rotation_set_matrix, 79
- mesh_scale, 79
- mesh_skip_line, 81
- mesh_struct, 52
- mesh_struct2, 52
- mesh_struct3, 52
- mesh_transform, 53
- mesh_translate, 81
- mesh_translate_vector, 81
- mesh_upsample, 82
- mesh_vector3, 53
- mesh_vertex, 53
- mesh_vertex_rotate, 82
- mesh_vface, 53
- mesh_write_file, 83
- mesh_write_off, 83
- mesh_write_ply, 84
- mesh_write_xyz, 85
- meshload.c, 85
 - mesh_load_file, 86
 - mesh_load_off, 87
 - mesh_load_ply, 88
 - mesh_load_xyz, 88
- meshops.c, 89
 - mesh_clone_mesh, 90
 - mesh_combine_mesh, 91
- meshtext.c, 91
 - mesh_count_words_in_line, 92
 - mesh_go_next_word, 93
 - mesh_isnumeric, 93
 - mesh_read_word, 93
 - mesh_read_word_only, 93
 - mesh_skip_line, 94
- meshtransform.c, 94
 - mesh_rotate, 95
 - mesh_rotation_create, 95
 - mesh_rotation_free, 96
 - mesh_rotation_set_angleaxis, 96
 - mesh_rotation_set_matrix, 97
 - mesh_scale, 97
 - mesh_translate, 98
 - mesh_translate_vector, 98
 - mesh_vertex_rotate, 98
- meshwrite.c, 100
 - mesh_write_file, 101
 - mesh_write_off, 101
 - mesh_write_ply, 102
 - mesh_write_xyz, 103
- num_edges
 - mesh, 9
- num_faces
 - mesh, 9
 - mesh_adjface, 10
- num_items
 - mesh_struct, 13
 - mesh_struct2, 13
 - mesh_struct3, 13
- num_vertices
 - mesh, 9
 - mesh_face, 12
- origin_type
 - mesh, 9
- r
 - mesh_color, 11
- vcolors
 - mesh, 10
- vertices
 - mesh, 10
 - mesh_edge, 11
 - mesh_face, 12
- vfaces
 - mesh, 10
- vnormals
 - mesh, 10
- x
 - mesh_vector3, 14
- y
 - mesh_vector3, 14
- z
 - mesh_vector3, 14