rsvs3D

0.0.0

Generated by Doxygen 1.8.15

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

Class Documentation

3.1 tetgen::apiparam Class Reference

Public Member Functions

- void **ReadJsonString** (const std::string &jsonStr)
- apiparam (const std::string &jsonStr)

  Constructs the object from a json string.

Public Attributes

- std::array< double, 3 > lowerB

  Lower domain bound.
- std::array< double, 3 > upperB

  Upper domain bound.
- std::array< double, 2 > surfedgelengths

  Controls the surface edgelengths in CFD in the order: {point of lowest curvature, point of highest curvature}.
- int **curvatureSmoothing**
- std::vector< double > edgelengths

  Controls the edgelengths at regular intervals.
- double distanceTol

  Distance tolerance.
- bool **generateMeshInside**
- std::string **command**

3.1.1 Constructor & Destructor Documentation

3.1.1.1 apiparam()

```
tetgen::apiparam::apiparam (
            const std::string & jsonStr )  [inline]
```

Constructs the object from a json string.

Parameters

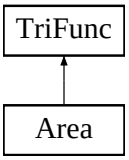
in	*jsonStr*	The json string

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

- incl/tetgenrsvs.hpp
- src/interfaces/tetgenrsvs.cpp

3.2 Area Class Reference

Inheritance diagram for Area:



Public Member Functions

- void **Calc** () override

Private Member Functions

- **TriFunc** ()
- **TriFunc** (int a)
- void **PreCalc** ()

Private Attributes

- vector< double > const * **p0**
- vector< double > const * **p1**
- vector< double > const * **p2**
- double **fun**
- ArrayVec< double > **jac**
- ArrayVec< double > **hes**

Additional Inherited Members

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

- incl/RSVSmath.hpp
- src/rsvs/RSVSmath.cpp

3.3 tetgenmesh::arraypool Class Reference

Public Member Functions

- void **restart** ()
- void **poolinit** (int sizeofobject, int log2objperblk)
- char * **getblock** (int objectindex)
- void * **lookup** (int objectindex)
- int **newindex** (void **newptr)
- **arraypool** (int sizeofobject, int log2objperblk)

Public Attributes

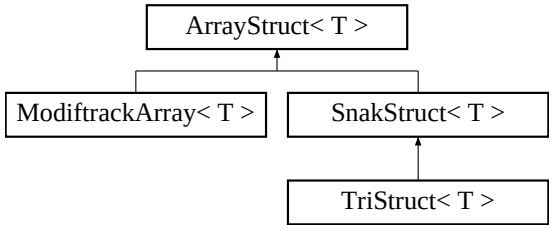
- int **objectbytes**
- int **objectsperblock**
- int **log2objectsperblock**
- int **objectsperblockmark**
- int **toparraylen**
- char ** **toparray**
- long **objects**
- unsigned long **totalmemory**

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

- modules/tetgen/tetgen.h
- modules/tetgen/tetgen.cpp
- modules/tetgen/tetgen.cxx

3.4 ArrayStruct< T > Class Template Reference

Inheritance diagram for ArrayStruct< T >:



Public Member Functions

- void **disp** () const
- void **disp** (const vector< int > &subs) const
- void **disp** (int iStart, int iEnd) const
- int **find** (int key, bool noWarn=false) const
- vector< int > **find_list** (const vector< int > &key, bool noWarn=false) const
- int **GetMaxIndex** () const
- void **Init** (int n)
- bool **isready** () const
- bool **checkready** ()
- void **Concatenate** (const ArrayStruct< T > &other)
- void **PopulateIndices** ()
- void **SetMaxIndex** ()
- void **HashArray** ()
- void **PrepareForUse** ()
- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu)
- void **write** (FILE *fid) const
- void **read** (FILE *fid)
- void **remove** (vector< int > delInd)
- void **TightenConnectivity** ()
- int **size** () const
- int **capacity** () const
- void **assign** (int n, T &newelem)
- void **push_back** (T &newelem)
- void **reserve** (int n)
- void **clear** ()
- void **issafeaccess** (const int a)
- const T * **operator()** (const int a) const
- const T * **isearch** (const int b) const
- T & **operator[ ]** (const int a)

Protected Member Functions

- void **ForceArrayReady** ()
- void **SetLastIndex** ()

Protected Attributes

- int **maxIndex**
- int **isHash** =0
- int **isSetMI** =0
- bool **readyforuse** =false
- bool **isInMesh** =false
- vector< T > **elems**
- unordered_multimap< int, int > **hashTable**

Friends

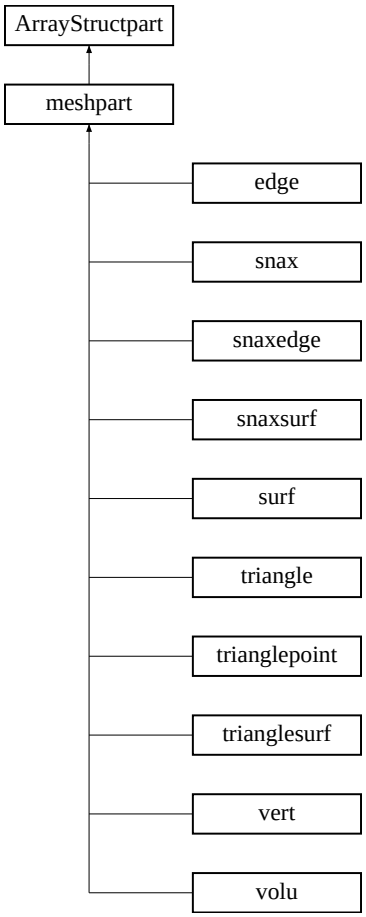
- class **mesh**
- class **snake**
- class **surf**
- int **TestTemplate_ArrayStruct** ()

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

- incl/arraystructures.hpp
- incl/arraystructures_incl.cpp

3.5 ArrayStructpart Class Reference

Inheritance diagram for ArrayStructpart:



Public Member Functions

- virtual void **disp** () const =0
- virtual int **Key** () const =0
- virtual void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu)=0
- virtual void **PrepareForUse** ()=0
- virtual bool **isready** (bool isInMesh) const =0
- virtual void **read** (FILE ∗fid)=0
- virtual void **write** (FILE ∗fid) const =0
- virtual void **TightenConnectivity** ()=0

Public Attributes

- int **index** =0
- bool **isBorder** =false

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

- incl/arraystructures.hpp

3.6 ArrayVec< T > Class Template Reference

Public Member Functions

- void **assign** (int nR, int nC, T newelem)
- void **size** (int &nR, int &nC) const
- void **clear** ()
- vector< T > & **operator[ ]** (const int a)
- const vector< T > & **operator[ ]** (const int a) const

Protected Attributes

- vector< vector< T > > **elems**
- vector< int > **dim**

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

- incl/vectorarray.hpp
- incl/vectorarray_incl.cpp

3.7 tetgenmesh::badface Class Reference

Public Attributes

- triface **tt**
- face **ss**
- REAL **key**
- REAL **cent** [6]
- point **forg**
- point **fdest**
- point **fapex**
- point **foppo**
- point **noppo**
- badface * **nextitem**

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

- modules/tetgen/tetgen.h

3.8 ConnecRemv Class Reference

Public Member Functions

- void **disp** ()

Public Attributes

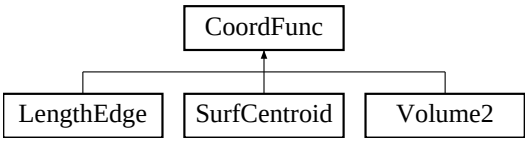
- int **keepind**
- int **typeobj**
- vector< int > **rmvind**
- vector< int > **scopeind**

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

- incl/mesh.hpp
- src/snake/snakeengine.cpp

3.9 CoordFunc Class Reference

Inheritance diagram for CoordFunc:



Public Member Functions

- bool **CheckValid** ()
- bool **MakeValid** ()
- void **PreCalc** ()
- void **assign** (vector< vector< double > const * > &coords)
- void **assign** (int pRepI, const vector< double > &pRep)
- void **ReturnDat** (double &a, ArrayVec< double > &b, ArrayVec< double > &c)
- void **ReturnDat** (ArrayVec< double > &a, ArrayVec< double > &b, ArrayVec< double > &c)
- void **ReturnDatPoint** (double **a, ArrayVec< double > **b, ArrayVec< double > **c)
- void **ReturnDatPoint** (ArrayVec< double > **a, ArrayVec< double > **b, ArrayVec< double > **c)
- virtual void **Calc** ()=0
- void **ResetDim** (int n)
- void **ResetNCoord** (int n)
- void **ResetNFun** (int n)
- **CoordFunc** (int n1)
- **CoordFunc** (int n1, int n2)
- **CoordFunc** (int n1, int n2, int n3)

Protected Member Functions

- bool MakeValidField (vector< double > const *mp)

  CoordFunc supports the same stuff as tri func but can have any number of points.
- void **InitialiseArrays** ()

Protected Attributes

- vector< vector< double > const * > **coords**
- double **fun**
- ArrayVec< double > **funA**
- ArrayVec< double > **jac**
- ArrayVec< double > **hes**
- bool **isReady**
- bool **isCalc**
- int **nDim**
- int **nCoord**
- int **nFun**

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

- incl/RSVSmath.hpp
- src/rsvs/RSVSmath.cpp

3.10 coordvec Class Reference

Handles the use and norm of a vector for which the norm and the unit value might be needed.

`#include <mesh.hpp>`

Public Member Functions

- double **CalcNorm** ()
- double **GetNorm** ()
- double **GetNorm** () const
- void **PrepareForUse** ()
- coordvec **Unit** () const
- double **Unit** (const int a) const
- double **Normalize** ()
- void **assign** (double a, double b, double c)
- double & **operator[ ]** (int a)
- double **operator()** (int a) const
- void **disp** () const
- bool **isready** () const
- const vector< double > & **usedata** () const
- const vector< double > * **retPtr** () const
- void **max** (const vector< double > &vecin)
- void **min** (const vector< double > &vecin)
- void **add** (const vector< double > &vecin)
- void **substract** (const vector< double > &vecin)

- void **substractfrom** (const vector< double > &vecin)
- void **div** (const vector< double > &vecin)
- void **div** (double scalin)
- void **mult** (const vector< double > &vecin)
- void **mult** (double scalin)
- vector< double > **cross** (const vector< double > &vecin) const
- double **dot** (const vector< double > &vecin) const
- double **angle** (const coordvec &coordin) const
- void **operator=** (const vector< double > &a)

Protected Attributes

- vector< double > **elems**
- double **norm**
- int **isuptodate**

3.10.1 Detailed Description

Handles the use and norm of a vector for which the norm and the unit value might be needed.

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

- incl/mesh.hpp
- src/grid/mesh.cpp

3.11 rsvstest::customtest Class Reference

Public Member Functions

- **customtest** (const char *testNameIn="")
- int **Run** (function< int()> test, const char *funcName)
- int RunSilent (function< int()> test, const char *funcName)

  Runs a test function silently except if it returns an error.
- void **PrintSummary** ()

Private Attributes

- int **testCount**
- int **errFlag**
- int **errCount**
- int **unhandledError**
- int **prevTime**
- int **runTotal**
- int **lastRunTime**
- std::string **testName**

3.11.1 Member Function Documentation

3.11.1.1 RunSilent()

```
int customtest::RunSilent (
            function< int()> test,
            const char * funcName )
```

Runs a test function silently except if it returns an error.

Parameters

in	*test*	The test function
in	*funcName*	string descriptor for the test.

Returns

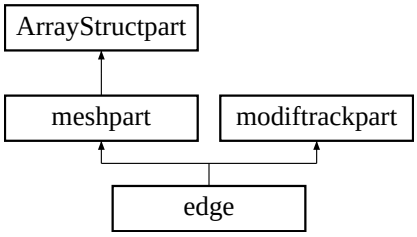
int number of errors captured.

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

- incl/test.hpp
- src/test/test.cpp

3.12 edge Class Reference

Inheritance diagram for edge:



Public Member Functions

- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu)
- void **disp** () const
- void **disptree** (const mesh &meshin, int n) const
- void **PrepareForUse** ()
- bool **isready** (bool isInMesh) const
- void **read** (FILE ∗fid)
- void **write** (FILE ∗fid) const
- void **TightenConnectivity** ()
- void GeometricProperties (const mesh ∗meshin, coordvec &centre, double &length) const

  MAth operations in mesh.
- double Length (const mesh &meshin) const

  Calculate the edge length.
- double LengthSquared (const mesh &meshin) const

  Calculate squared edge length.
- bool IsLength0 (const mesh &meshin, double eps=__DBL_EPSILON__) const

  Returns.
- bool **vertconneq** (const edge &other) const
- **edge** (const edge &oldEdge)
- void **operator=** (const edge ∗other)
- int **Key** () const

Public Attributes

- friend **edgearray**
- vector< int > **vertind**
- vector< int > **surfind**

Friends

- class **mesh**

Additional Inherited Members

3.12.1 Member Function Documentation

3.12.1.1 IsLength0()

```
bool edge::IsLength0 (
            const mesh & meshin,
            double eps = __DBL_EPSILON__ ) const
```

Returns.

Parameters

in	*meshin*	the mesh in which the edge existes
in	*eps*	Tolerance, number under which the length must be to be considered 0. Defaults to **DBL_EPSILON**.

Returns

Wether Length squared is below eps squared.

3.12.1.2 Length()

```
double edge::Length (
            const mesh & meshin ) const
```

Calculate the edge length.

Parameters

in	*meshin*	the mesh in which the edge existes

Returns

the length of the edge

3.12.1.3 LengthSquared()

```
double edge::LengthSquared (
            const mesh & meshin ) const
```

Calculate squared edge length.

Parameters

in	*meshin*	the mesh in which the edge existes

Returns

the squared length of the edge

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

- incl/mesh.hpp
- src/grid/mesh.cpp

3.13 tetgenmesh::face Class Reference

Public Member Functions

- face & **operator=** (const face &s)

Public Attributes

- shellface * **sh**
- int **shver**

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

- modules/tetgen/tetgen.h

3.14 tetgenio::facet Struct Reference

Public Attributes

- polygon * **polygonlist**
- int **numberofpolygons**
- REAL * **holelist**
- int **numberofholes**

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

- modules/tetgen/tetgen.h

3.15 param::files Class Reference

Public Member Functions

- void **PrepareForUse** ()

Public Attributes

- bool **appcasename2outdir**
- ioin **ioin**
- ioout **ioout**
- exports **exportconfig**

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

- incl/parameters.hpp
- src/parameters.cpp

3.16 param::filltype< T > Struct Template Reference

Public Attributes

- bool **active** =false
- T **fill**

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

- incl/parameters.hpp

3.17 tetgenmesh::flipconstraints Class Reference

Public Attributes

- int **enqflag**
- int **chkencflag**
- int **unflip**
- int **collectnewtets**
- int **collectencsegflag**
- int **remove_ndelaunay_edge**
- REAL **bak_tetprism_vol**
- REAL **tetprism_vol_sum**
- int **remove_large_angle**
- REAL **cosdihed_in**
- REAL **cosdihed_out**
- int **checkflipeligibility**
- point **seg** [2]
- point **fac** [3]
- point **remvert**

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

- modules/tetgen/tetgen.h

3.18 param::grid Class Reference

Public Member Functions

- void **PrepareForUse** ()

Public Attributes

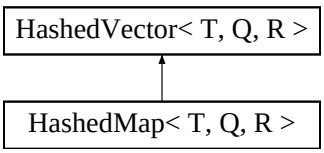
- voxel **voxel**
- voronoi **voronoi**
- std::array< realbounds, 3 > **domain**
- std::array< realbounds, 3 > **physdomain**
- std::string **activegrid**

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

- incl/parameters.hpp
- src/parameters.cpp

3.19 HashedMap< T, Q, R > Class Template Reference

Inheritance diagram for HashedMap< T, Q, R >:



Public Member Functions

- void **GenerateHash** ()

Public Attributes

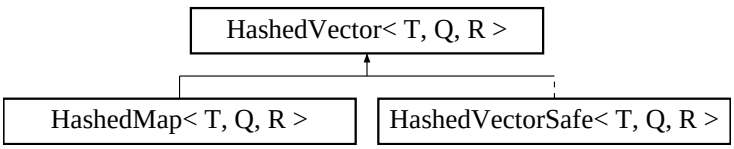
- vector< R > **targ**

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

- incl/arraystructures.hpp
- incl/arraystructures_incl.cpp

3.20 HashedVector< T, Q, R > Class Template Reference

Inheritance diagram for HashedVector< T, Q, R >:



Public Member Functions

- void **GenerateHash** ()
- int **find** (const T key) const
- vector< int > **findall** (const T key) const
- int **count** (const T key) const
- vector< int > **count** (const vector< T > &key) const
- vector< int > **find_list** (const vector< T > &key) const
- bool **operator()** (const Q &key) const
- bool **IsInVec** (const Q &key) const

Public Attributes

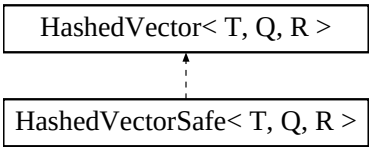
- vector< T > **vec**
- unordered_multimap< T, R > **hashTable**
- bool **isHash** =false

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

- incl/arraystructures.hpp
- incl/arraystructures_incl.cpp

3.21 HashedVectorSafe< T, Q, R > Class Template Reference

Inheritance diagram for HashedVectorSafe< T, Q, R >:



Public Member Functions

- void **operator=** (const vector< T > &a)
- void **operator=** (const HashedVector< T, Q > &a)
- T & **operator[ ]** (const int a)
- const T & **operator[ ]** (const int a) const
- const T & **isearch** (const int b) const

Additional Inherited Members

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

- incl/arraystructures.hpp

3.22 tetgenmesh::insertvertexflags Class Reference

Public Attributes

- int **iloc**
- int **bowywat**
- int **lawson**
- int **splitbdflag**
- int **validflag**
- int **respectbdflag**
- int **rejflag**

- int **chkencflag**
- int **cdtflag**
- int **assignmeshsize**
- int **sloc**
- int **sbowywat**
- int **refineflag**
- triface **refinetet**
- face **refinesh**
- int **smlenflag**
- REAL **smlen**
- point **parentpt**

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

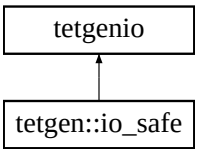
- modules/tetgen/tetgen.h

3.23 tetgen::io_safe Class Reference

Class for memory safe interface with tetgen.h.

```
#include <tetgenrsvs.hpp>
```

Inheritance diagram for tetgen::io_safe:



Public Member Functions

- void **allocate** ()
- void **allocatefacet** (int fIndex)
- void **allocatefacet** (int fIndex, int numPoly)
- void **allocatefacetpolygon** (int fIndex, int pIndex)
- void **allocatefacetpolygon** (int fIndex, int pIndex, int numVerts)
- void **SpecifyTetPointMetric** (int startPnt, int numPnt, const std::vector< double > &mtrs)
- void **SpecifyIndividualTetPointMetric** (int startPnt, int numPnt, const std::vector< double > &mtrs)
- void **SpecifyTetFacetMetric** (int startPnt, int numPnt, int marker)

Public Attributes

- REAL * **pointlist**
- REAL * **pointattributelist**
- REAL * **pointmtrlist**
- int * **pointmarkerlist**
- int **numberofpointmtrs**
- int * **tetrahedronlist**
- REAL * **tetrahedronattributelist**
- REAL * **tetrahedronvolumelist**
- int * **neighborlist**
- facet * **facetlist**
- int * **facetmarkerlist**
- REAL * **facetconstraintlist**
- int **numberoffacetconstraints**
- REAL * **holelist**
- REAL * **regionlist**
- REAL * **segmentconstraintlist**
- int * **edgelist**
- int * **edgemarkerlist**
- int * **o2edgelist**
- int * **edge2tetlist**
- int * **face2edgelist**
- int * **face2tetlist**
- REAL * **vpointlist**
- voroedge * **vedgelist**
- vorofacet * **vfacetlist**
- int ** **vcelllist**

Additional Inherited Members

3.23.1 Detailed Description

Class for memory safe interface with tetgen.h.

This class provides a method called `allocate` which allocates the memory for the io arrays using the new command. Command `deallocate` can be used to free the memory before destruction, or otherwise it is called uppon when object goes out of scope.

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

- incl/tetgenrsvs.hpp
- src/interfaces/tetgenrsvs.cpp

3.24 param::ioin Class Reference

Class containing the input configuration these are files to load etc.

`#include <parameters.hpp>`

Public Member Functions

- void **PrepareForUse** ()

Public Attributes

- std::string **snakemeshname**
- std::string **volumeshname**
- std::string **targetfill**
- std::string **casename**

3.24.1 Detailed Description

Class containing the input configuration these are files to load etc.

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

- incl/parameters.hpp
- src/parameters.cpp

3.25 param::ioout Class Reference

Class containing the output configuration these are files to store and where to store them.

`#include <parameters.hpp>`

Public Member Functions

- void **PrepareForUse** ()

Public Attributes

- std::string **pathoutdir**
- std::string **pathpattern**
- std::string **basenamepattern**
- std::string **basenameoutdir**
- std::string **outdir**
- std::string **pattern**
- bool **redirectcout**
- bool **redirectcerr**
- int **logginglvl**
- int **outputlvl**

3.25.1 Detailed Description

Class containing the output configuration these are files to store and where to store them.

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

- incl/parameters.hpp
- src/parameters.cpp

3.26 integrate::iteratereturns Class Reference

Public Member Functions

- **iteratereturns** (int n, int s, double t)

Public Attributes

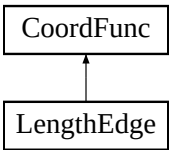
- int **nVoluZone** =0
- int **stepNum** =0
- double **timeT** =0.0

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

- incl/RSVSintegration.hpp

3.27 LengthEdge Class Reference

Inheritance diagram for LengthEdge:



Public Member Functions

- void **Calc** () override

Private Member Functions

- void **PreCalc** ()

Private Attributes

- vector< vector< double > const * > **coords**
- double **fun**
- ArrayVec< double > **jac**
- ArrayVec< double > **hes**

Additional Inherited Members

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

- incl/RSVSmath.hpp
- src/rsvs/RSVSmath.cpp

3.28 tetgenmesh::memorypool Class Reference

Public Member Functions

- **memorypool** (int, int, int, int)
- void **poolinit** (int, int, int, int)
- void **restart** ()
- void * **alloc** ()
- void **dealloc** (void *)
- void **traversalinit** ()
- void * **traverse** ()

Public Attributes

- void ** **firstblock**
- void ** **nowblock**
- void * **nextitem**
- void * **deaditemstack**
- void ** **pathblock**
- void * **pathitem**
- int **alignbytes**
- int **itembytes**
- int **itemwords**
- int **itemsperblock**
- long **items**
- long **maxitems**
- int **unallocateditems**
- int **pathitemsleft**

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

- modules/tetgen/tetgen.h
- modules/tetgen/tetgen.cpp
- modules/tetgen/tetgen.cxx

3.29 mesh Class Reference

Public Member Functions

- void **RemoveFromFamily** ()
- void **AddChild** (mesh ∗meshin)
- void **AddParent** (mesh ∗meshin)
- void **AddParent** (mesh ∗meshin, vector< int > &parentind)
- void **AddChild** (mesh ∗meshin, vector< int > &parentind)
- void **SetMeshDepElm** ()
- void **MaintainLineage** ()
- int **CountParents** () const
- int **SurfInParent** (int surfind) const
- void **SurfInParent** (vector< int > &listInParent) const
- void **ElmOnParentBound** (vector< int > &listInParent, vector< int > &voluInd, bool isBorderBound=true, bool outerVolume=true) const
- void **SurfOnParentBound** (vector< int > &listInParent, vector< int > &voluInd, bool isBorderBound, bool outerVolume) const
- void **EdgeOnParentBound** (vector< int > &listInParent, vector< int > &voluInd, bool isBorderBound, bool outerVolume) const
- int **CountVoluParent** () const
- void **ReturnParentMap** (vector< int > &currind, vector< int > &parentpos, vector< pair< int, int >> &parentcases, vector< double > &voluVals) const
- void **MapVolu2Parent** (const vector< double > &fillIn, const vector< pair< int, int >> &parentcases, double volu::∗mp=&volu::fill)
- void **MapVolu2Self** (const vector< double > &fillIn, const vector< int > &elms, double volu::∗mp=&volu::fill)
- void **VoluValuesofParents** (int elmInd, vector< double > &vals, int volType=0) const
- void **VoluValuesofParents** (int elmInd, vector< double > &vals, double volu::∗mp) const
- void **SurfValuesofParents** (int elmInd, vector< double > &vals, int volType=0) const
- void **SurfValuesofParents** (int elmInd, vector< double > &vals, double surf::∗mp) const
- int **ParentElementIndex** (int childElmInd, int parentInd=0) const
- int **WhatDim** () const
- void **HashArray** ()
- void **SetMaxIndex** ()
- void **GetMaxIndex** (int ∗nVert, int ∗nEdge, int ∗nSurf, int ∗nVolu) const
- void **Init** (int nVe, int nE, int nS, int nVo)
- void **size** (int &nVe, int &nE, int &nS, int &nVo) const
- void **reserve** (int nVe, int nE, int nS, int nVo)
- void **PrepareForUse** (bool needOrder=true)
- void **disp** () const
- void **displight** () const
- void **Concatenate** (const mesh &other)
- bool **isready** () const
- void **PopulateIndices** ()
- void **TightenConnectivity** ()
- int **TestConnectivity** (const char ∗strRoot="") const
- int **TestConnectivityBiDir** (const char ∗strRoot="", bool emptyIsErr=true) const
- void **write** (FILE ∗fid) const
- void **read** (FILE ∗fid)
- int **write** (const char ∗str) const
- int **read** (const char ∗str)
- void **MakeCompatible_inplace** (mesh &other) const
- mesh **MakeCompatible** (mesh other) const
- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu)

- void **SwitchIndex** (int typeInd, int oldInd, int newInd, const vector< int > &scopeInd={0})
- void **RemoveIndex** (int typeInd, int oldInd)
- int ConnectedVertex (vector< int > &vertBlock) const

  Return in a vector for each vertex a block number which it is part of.
- int **ConnectedVolumes** (vector< int > &volBlock, const vector< bool > &boundaryFaces={}) const
- void **ForceCloseContainers** ()
- void **RemoveSingularConnectors** (const std::vector< int > &rmvVertInds={}, bool voidError=true)
- std::vector< int > **MergeGroupedVertices** (HashedVector< int, int > &closeVert, bool delVerts=true)
- vector< int > **OrderEdges** ()
- void **SetBorders** ()
- void **OrientFaces** ()
- void **GetOffBorderVert** (vector< int > &vertList, vector< int > &voluInd, int outerVolume=-1)
- void **GetOffBorderVert** (vector< int > &vertList, vector< int > &voluInd, int outerVolume=-1) const
- void **GetOffBorderVert3D** (vector< int > &vertList, vector< int > &voluInd, int outerVolume=-1) const
- void **GetOffBorderVert2D** (vector< int > &vertInd, vector< int > &surfind, int outerVolume=-1) const
- coordvec **CalcCentreVolu** (int ind) const
- coordvec **CalcPseudoNormalSurf** (int ind) const
- vector< int > VertexInVolume (const vector< double > testVertices, int sizeVert=3) const

  Finds for each vertex, the volume object containing it.
- grid::transformation **Scale** ()
- grid::transformation **Scale** (const grid::limits &domain)
- void LinearTransform (const grid::transformation &transform)

  Applies a linear transformation to the points on a grid.
- void LinearTransformFamily (const grid::transformation &transform)

  Applies a linear transform to child and parent meshes.
- void **LoadTargetFill** (const std::string &fileName)
- grid::limits **BoundingBox** () const
- void **ReturnBoundingBox** (std::array< double, 3 > &lowerB, std::array< double, 3 > &upperB) const
- void **Crop** (vector< int > indList, int indType=1)
- vector< int > AddBoundary (const vector< double > &lb, const vector< double > &ub)

  Adds boundaries alond max and min xyz planes.
- void **CropAtBoundary** (const vector< double > &lb, const vector< double > &ub)

Public Attributes

- vertarray **verts**
- edgearray **edges**
- surfarray **surfs**
- voluarray **volus**
- meshdependence **meshtree**

Private Member Functions

- void **SetLastIndex** ()
- void **OrientSurfaceVolume** ()
- void **OrientEdgeSurface** ()
- int **OrientRelativeSurfaceVolume** (vector< int > &surfOrient)
- void **ArraysAreHashed** ()
- void _LinearTransformGeneration (const grid::transformation &transform, vector< mesh * > meshdependence↩ ::*mp)

  Applies reccursively linear transforms to a tree of meshes.

Private Attributes

- bool **borderIsSet** =false
- bool **meshDepIsSet** =false
- bool **facesAreOriented** =false
- int **meshDim** =0

Friends

- class **snake**

3.29.1 Member Function Documentation

3.29.1.1 _LinearTransformGeneration()

```
void mesh::_LinearTransformGeneration (
            const grid::transformation & transform,
            vector< mesh * > meshdependence::* mp )  [private]
```

Applies reccursively linear transforms to a tree of meshes.

Parameters

in	*transform*	The transform
in	*meshdependence*	A member pointer to either the parent meshes or the child meshes of the meshtree.

3.29.1.2 AddBoundary()

```
std::vector< int > mesh::AddBoundary (
            const vector< double > & lb,
            const vector< double > & ub )
```

Adds boundaries alond max and min xyz planes.

Arguments

Parameters

in	*lb*	lower boundary vector of 3 doubles.
in	*ub*	upper boundary vector of 3 doubles.

Returns

List of vertex indices in the mesh which lie outside.

Raises:

- logic_error,
-
-

Process: THis method could be readily refactored to allow treatment of more complex boundaries

Steps: 1 - Identify vertices lying outside 2 - Indentify connectors lying on boundary a - edges b - surfs c - volus 3 - Introduce boundary vertices (BV) 4 - Connect those BV to form new boundary edges (BE) 5 - Assemble BEs inside a volu into a boundary surf (BS) (This process is similar to the voronoisation)

3.29.1.3 ConnectedVertex()

```
int mesh::ConnectedVertex (
            vector< int > & vertBlock ) const
```

Return in a vector for each vertex a block number which it is part of.

Fills a vector with a number for each vertex corresponding to a group of connected edges it is part of , can be used close surfaces in 2D or volumes in 3D. Uses a flood fill with queue method.

Parameters

[in/out]	vertBlock Either a vector of the same size contaigning 0 for vertices which need to be labelled and some other integers in other positions. OR an empty vector.

Returns

The total number of blocks of vertices identified.

3.29.1.4 LinearTransform()

```
void mesh::LinearTransform (
            const grid::transformation & transform )
```

Applies a linear transformation to the points on a grid.

Parameters

in	*transform*	The transform to apply.

3.29.1.5 LinearTransformFamily()

```
void mesh::LinearTransformFamily (
            const grid::transformation & transform )
```

Applies a linear transform to child and parent meshes.

Parameters

in	*transform*	The transform

3.29.1.6 VertexInVolume()

```
vector< int > mesh::VertexInVolume (
            const vector< double > testVertices,
            int sizeVert = 3 ) const
```

Finds for each vertex, the volume object containing it.

This only works robustly for outside points for convex meshes.

Parameters

in	*testVertices*	The test vertices
in	*sizeVert*	The size of each vertex data

Returns

returns a list of indices containing the same number of values as there are input vertices (testVertices/sizeVert)

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

- incl/mesh.hpp
- src/grid/mesh.cpp

3.30 meshdependence Class Reference

Protected Member Functions

- int **AddParent** (mesh *meshin)
- int **AddChild** (mesh *meshin)
- void **AddParent** (mesh *meshin, vector< int > &parentind)
- void **RemoveChild** (mesh *meshin)
- void **RemoveParent** (mesh *meshin)

Protected Attributes

- int **nParents** = 0
- vector< int > **elemind**
- vector< mesh * > **parentmesh**
- vector< mesh * > **childmesh**
- vector< HashedVectorSafe< int, int > > **parentconn**

Friends

- class **mesh**

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

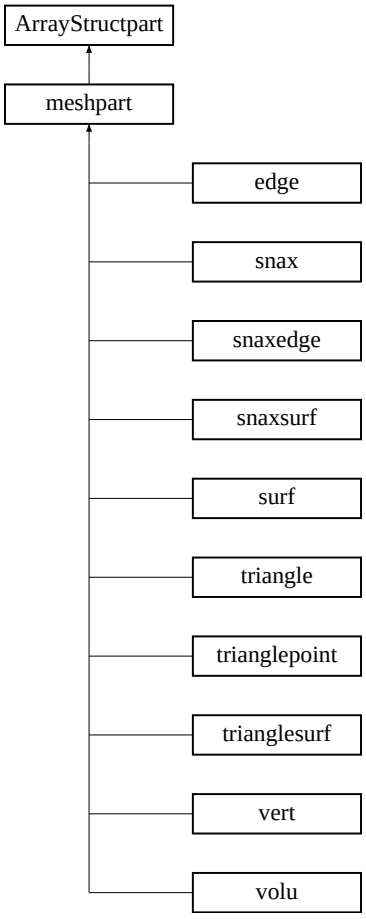
- incl/mesh.hpp
- src/grid/mesh.cpp

3.31 meshpart Class Reference

/Abstract class to ensure interface is correct

`#include <mesh.hpp>`

Inheritance diagram for meshpart:



Public Member Functions

- virtual void **disptree** (const mesh &meshin, int n) const =0

Additional Inherited Members

3.31.1 Detailed Description

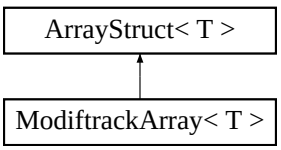
/Abstract class to ensure interface is correct

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

- incl/mesh.hpp

3.32 ModiftrackArray< T > Class Template Reference

Inheritance diagram for ModiftrackArray< T >:



Public Member Functions

- void **SetNoModif** ()
- void **ReturnModifInd** (vector< int > &vecind)
- void **ReturnModifLog** (vector< bool > &modiflog)
- T & **operator[ ]** (const int a)

Friends

- class **mesh**
- class **snake**

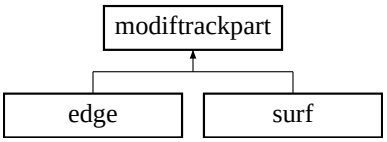
Additional Inherited Members

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

- incl/arraystructures.hpp
- incl/snakstruct_incl.cpp

3.33 modiftrackpart Class Reference

Inheritance diagram for modiftrackpart:



Public Member Functions

- bool **returnIsModif** () const

Protected Attributes

- bool **isModif** =true

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

- incl/arraystructures.hpp

3.34 tetgenmesh::optparameters Class Reference

Public Attributes

- int **max_min_volume**
- int **min_max_aspectratio**
- int **min_max_dihedangle**
- REAL **initval**
- REAL **imprval**
- int **numofsearchdirs**
- REAL **searchstep**
- int **maxiter**
- int **smthiter**

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

- modules/tetgen/tetgen.h

3.35 param::parameters Class Reference

Public Member Functions

- void **PrepareForUse** ()

Public Attributes

- rsvs **rsvs**
- snaking **snak**
- grid **grid**
- files **files**

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

- incl/parameters.hpp
- src/parameters.cpp

3.36 tetgenio::pointparam Struct Reference

Public Attributes

- REAL **uv** [2]
- int **tag**
- int **type**

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

- modules/tetgen/tetgen.h

3.37 tetgenio::polygon Struct Reference

Public Attributes

- int * **vertexlist**
- int **numberofvertices**

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

- modules/tetgen/tetgen.h

3.38 param::rsvs Class Reference

Parameters related to the Velocity calculation and VOS steps.

`#include <parameters.hpp>`

Public Member Functions

- void **PrepareForUse** ()

Public Attributes

- int **solveralgorithm**
- filltype< double > **cstfill**
- filltype< std::string > **filefill**
- filltype< std::string > **makefill**

3.38.1 Detailed Description

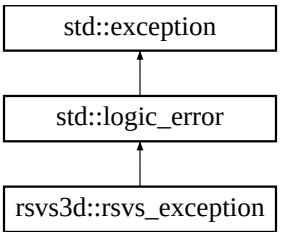
Parameters related to the Velocity calculation and VOS steps.

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

- incl/parameters.hpp
- src/parameters.cpp

3.39 rsvs3d::rsvs_exception Class Reference

Inheritance diagram for rsvs3d::rsvs_exception:



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

- incl/warning.hpp

3.40 RSVScalc Class Reference

Class to handle the RSVS calculation.

`#include <RSVScalc.hpp>`

Public Member Functions

- void BuildMathArrays (int nDv, int nConstr)

  Builds mathematics arrays.
- void BuildConstrMap (const triangulation &triangleRSVS)

  Builds the constraint mapping.
- void BuildConstrMap (const mesh &meshin)

  Builds the constraint mapping.
- int BuildDVMap (const vector< int > &vecin)

  Builds a Design variable map.
- bool SnakDVcond (const triangulation &triRSVS, int ii)

  Returns wether a snaxel is a design variable or not.
- void PrepTriangulationCalc (const triangulation &triRSVS)

  Groups actions needed before the calculation of triangular quantities.
- void CalculateMesh (mesh &meshin)

  Calculates the mesh volumes.
- void CalculateTriangulation (const triangulation &triRSVS, int derivMethod=0)

  Calculates the triangulation volume and area derivatives.
- void CalcTriangle (const triangle &triIn, const triangulation &triRSVS, bool isObj=true, bool isConstr=true, bool isDeriv=true)

  Calculates the properties of single triangle.
- void CalcTriangleFD (const triangle &triIn, const triangulation &triRSVS, bool isObj=true, bool isConstr=true, bool isDeriv=true)

  Calculates the properties of single triangle using Finite difference.
- void CalcTriangleDirectVolume (const triangle &triIn, const triangulation &triRSVS, bool isObj=true, bool is↩ Constr=true, bool isDeriv=true)

  Calculates the properties of single triangle using direct calculation.
- void CalcTriangleEdgeLength (const triangle &triIn, const triangulation &triRSVS, bool isObj=true, bool is↩ Constr=true, bool isDeriv=true)

  Calculates the properties of single triangle for 2D RSVS.
- void ReturnConstrToMesh (triangulation &triRSVS) const

  Returns a constraint to the triangulation::meshDep.
- void ReturnConstrToMesh (mesh &meshin, double volu::∗mp=&volu::volume) const

  Returns a constraint to the mesh.
- void CheckAndCompute (int calcMethod=0)

  Prepare the active arrays for SQP calculation and calculate the SQP step.
- void ComputeSQPstep (int calcMethod, MatrixXd &dConstrAct, RowVectorXd &dObjAct, VectorXd &constr↩ Act, VectorXd &lagMultAct)

  Calculates the next SQP step.
- bool PrepareMatricesForSQP (MatrixXd &dConstrAct, MatrixXd &HConstrAct, MatrixXd &HObjAct, Row↩ VectorXd &dObjAct, VectorXd &constrAct, VectorXd &lagMultAct)

  Prepares the matrices needed for the SQP step calculation.
- void ReturnVelocities (triangulation &triRSVS)

  Returns velocities to the snaxels.
- int numConstr ()

  Getter for the number of constraints.
- void Print2Screen (int outType=0) const

  Prints different amounts of `RSVScalc` owned data to the screen.
- void ConvergenceLog (ofstream &out, int loglvl=3) const

  Print convergence information to file stream.

Public Attributes

- MatrixXd dConstr

  Constraint Jacobian, size: [nConstr, nDv].
- MatrixXd HConstr

  Constraint Hessian, size: [nDv, nDv].
- MatrixXd HObj

  Objective Hessian, size: [nDv, nDv].
- MatrixXd HLag

  Lagrangian Hessian, size: [nDv, nDv].
- RowVectorXd dObj

  Objective Jacobian, size: [1, nDv].
- VectorXd constr

  Constraint value vector, size: [nConstr, 1].
- VectorXd lagMult

  Lagrangian multiplier, size: [nConstr, 1].
- VectorXd deltaDV

  Change in design variable, assigned to snake velocity, size: [nDv, 1].
- VectorXd constrTarg

  Constraint target values, size: [nConstr, 1].
- MatrixXd **dvCallConstr**
- double obj =0.0

  Objective function value.
- double limLag = INFINITY

  Value at which a Lagrangian multiplier is considered problematically large.
- std::vector< bool > isConstrAct

  is the corresponding constraint active?
- std::vector< bool > isDvAct

  Is the corresponding design variable active?
- std::vector< int > subConstrAct

  Vector of subscripts of the active constraints.
- std::vector< int > subDvAct

  Vector of subscripts of the active design variables.
- HashedVector< int, int > dvMap

  Maps the snake indices to the position in the design variable vector.
- HashedMap< int, int, int > constrMap

  maps snakemesh volu onto constr
- std::vector< pair< int, int > > constrList

  keeps pairs with parentindex and voluindex

Protected Attributes

- int nDv =0

  Number of design variables.
- int nConstr =0

  Number of constraints.
- int falseaccess =0

  Number of false access operations.
- bool returnDeriv =true

  Return the derivatives (obsolete/unused)

3.40.1 Detailed Description

Class to handle the RSVS calculation.

This class calculates volume and area metrics in a triangulated snake to update the velocity and volumes. It uses an SQP algorithm to compute the velocities.

3.40.2 Member Function Documentation

3.40.2.1 BuildConstrMap() [1/2]

```
void RSVScalc::BuildConstrMap (
            const triangulation & triangleRSVS )
```

Builds the constraint mapping.

Parameters

in	*triangleRSVS*	Triangulation containing the RSVS.

3.40.2.2 BuildConstrMap() [2/2]

```
void RSVScalc::BuildConstrMap (
            const mesh & meshin )
```

Builds the constraint mapping.

Parameters

in	*meshin*	mesh for constraint building.

3.40.2.3 BuildDVMap()

```
int RSVScalc::BuildDVMap (
            const vector< int > & vecin )
```

Builds a Design variable map.

Parameters

in	*vecin*	The input vector of design variable indices.

Returns

The number of design variable.

3.40.2.4 BuildMathArrays()

```
void RSVScalc::BuildMathArrays (
            int nDv,
            int nConstr )
```

Builds mathematics arrays.

Parameters

in	*nDv*	Number of design variables.
in	*nConstr*	Number of constraints.

3.40.2.5 CalcTriangle()

```
void RSVScalc::CalcTriangle (
            const triangle & triIn,
            const triangulation & triRSVS,
            bool isObj = true,
            bool isConstr = true,
            bool isDeriv = true )
```

Calculates the properties of single triangle.

These values are returned to the class math arrays.

Parameters

in	*triIn*	The triangle to measure.
in	*triRSVS*	The containing triangulation object.
in	*isObj*	Calculate objective?
in	*isConstr*	Calculate constraint?
in	*isDeriv*	Calculate derivatives?

3.40.2.6 CalcTriangleDirectVolume()

```
void RSVScalc::CalcTriangleDirectVolume (
            const triangle & triIn,
            const triangulation & triRSVS,
```

```
            bool isObj = true,
            bool isConstr = true,
            bool isDeriv = true )
```

Calculates the properties of single triangle using direct calculation.

These values are returned to the class math arrays.

Parameters

in	*triIn*	The triangle to measure.
in	*triRSVS*	The containing triangulation object.
in	*isObj*	Calculate objective?
in	*isConstr*	Calculate constraint?
in	*isDeriv*	Calculate derivatives?

<-—Change assignement

<-—Change assignement

3.40.2.7 CalcTriangleEdgeLength()

```
void RSVScalc::CalcTriangleEdgeLength (
            const triangle & triIn,
            const triangulation & triRSVS,
            bool isObj = true,
            bool isConstr = true,
            bool isDeriv = true )
```

Calculates the properties of single triangle for 2D RSVS.

These values are returned to the class math arrays.

Parameters

in	*triIn*	The triangle to measure.
in	*triRSVS*	The containing triangulation object.
in	*isObj*	Calculate objective?
in	*isConstr*	Calculate constraint?
in	*isDeriv*	Calculate derivatives?

3.40.2.8 CalcTriangleFD()

```
void RSVScalc::CalcTriangleFD (
            const triangle & triIn,
            const triangulation & triRSVS,
            bool isObj = true,
            bool isConstr = true,
            bool isDeriv = true )
```

Calculates the properties of single triangle using Finite difference.

These values are returned to the class math arrays.

Parameters

in	*triIn*	The triangle to measure.
in	*triRSVS*	The containing triangulation object.
in	*isObj*	Calculate objective?
in	*isConstr*	Calculate constraint?
in	*isDeriv*	Calculate derivatives?

3.40.2.9 CalculateMesh()

```
void RSVScalc::CalculateMesh (
            mesh & meshin )
```

Calculates the mesh volumes.

Parameters

meshin	The mesh.

3.40.2.10 CalculateTriangulation()

```
void RSVScalc::CalculateTriangulation (
            const triangulation & triRSVS,
            int derivMethod = 0 )
```

Calculates the triangulation volume and area derivatives.

Parameters

in	*triRSVS*	The triangle rsvs
in	*derivMethod*	The differentiation method to use. 1 : Finite Difference, 2 : Direct calculation, all others : differentiation.

3.40.2.11 CheckAndCompute()

```
void RSVScalc::CheckAndCompute (
            int calcMethod = 0 )
```

Prepare the active arrays for SQP calculation and calculate the SQP step.

Parameters

in	*calcMethod*	Calculation method for SQP. Check :meth:RSVScalc::ComputeSQPstep for detail.

3.40.2.12 ComputeSQPstep()

```
void RSVScalc::ComputeSQPstep (
            int calcMethod,
            MatrixXd & dConstrAct,
            RowVectorXd & dObjAct,
            VectorXd & constrAct,
            VectorXd & lagMultAct )
```

Calculates the next SQP step.

In normal operation the constraint should be 0 through 4. With 0 the default. By adding 10 to these values the "constraint only" mode is enabled which performs a gradient descent step based on the constraint.

Parameters

in	*calcMethod*	The calculation method. 10 can be added to all values to enable the "constraint only" mode. Values correspond to the following: `Eigen::HouseholderQR` (1); * `Eigen::ColPivHouseholderQR` (2) - Default; Eigen::LLT<MatrixXd> (3); `Eigen::PartialPivLU` (4);
	dConstrAct	The active constraint Jacobian
	dObjAct	The active objective Jacobian
	constrAct	The active constraint values
	lagMultAct	The active lagrangian multipliers.

3.40.2.13 ConvergenceLog()

```
void RSVScalc::ConvergenceLog (
            ofstream & out,
            int loglvl = 3 ) const
```

Print convergence information to file stream.

Parameters

	out	The output filestream
in	*loglvl*	The logging detail to output. <1 nothing, ==1 Vector statistics, ==2 ...and constraint vectors, >2 ...and snaxel velocity vector.

3.40.2.14 numConstr()

```
int RSVScalc::numConstr ( )  [inline]
```

Getter for the number of constraints.

Returns

The number of constraints.

3.40.2.15 PrepareMatricesForSQP()

```
bool RSVScalc::PrepareMatricesForSQP (
            MatrixXd & dConstrAct,
            MatrixXd & HConstrAct,
            MatrixXd & HObjAct,
            RowVectorXd & dObjAct,
            VectorXd & constrAct,
            VectorXd & lagMultAct )
```

Prepares the matrices needed for the SQP step calculation.

Parameters

dConstrAct	The active constraint Jacobian
HConstrAct	The active constraint hessian
HObjAct	The active objective hessian
dObjAct	The active objective Jacobian
constrAct	The active constraint values
lagMultAct	The active lagrangian multipliers.

Returns

Returns wether the calculation should be performed or not.

3.40.2.16 PrepTriangulationCalc()

```
void RSVScalc::PrepTriangulationCalc (
            const triangulation & triRSVS )
```

Groups actions needed before the calculation of triangular quantities.

Parameters

in	*triRSVS*	The triangulation object.

3.40.2.17 Print2Screen()

```
void RSVScalc::Print2Screen (
            int outType = 0 ) const
```

Prints different amounts of RSVScalc owned data to the screen.

Parameters

in	*outType*	The output type to print, values [2,3,4].

3.40.2.18 ReturnConstrToMesh() [1/2]

```
void RSVScalc::ReturnConstrToMesh (
            triangulation & triRSVS ) const
```

Returns a constraint to the triangulation::meshDep.

Parameters

triRSVS	The triangulation object.

3.40.2.19 ReturnConstrToMesh() [2/2]

```
void RSVScalc::ReturnConstrToMesh (
            mesh & meshin,
            double volu::* mp = &volu::volume ) const
```

Returns a constraint to the mesh.

Parameters

	meshin	The input mesh.
in	*volu*	The volumetric field that data needs to be returned to. It is a member point of class volu.

3.40.2.20 ReturnVelocities()

```
void RSVScalc::ReturnVelocities (
            triangulation & triRSVS )
```

Returns velocities to the snaxels.

Returns velocities to the snake in the triangulation object.

Parameters

triRSVS	The triangulation object, affects the triangulation::snakeDep attribute.
triRSVS	The triangulation object of the RSVS

3.40.2.21 SnakDVcond()

```
bool RSVScalc::SnakDVcond (
            const triangulation & triRSVS,
            int ii )
```

Returns wether a snaxel is a design variable or not.

If the snaxel is frozen and all its neighbours are frozen, it is not a design variable.

Parameters

in	*triRSVS*	The triangulation which is being calculated
in	*ii*	the snaxel subscript.

Returns

wether the snaxel is design variable or not.

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

- incl/RSVScalc.hpp
- src/rsvs/RSVScalc.cpp
- src/rsvs/RSVScalc_core.cpp
- src/rsvs/RSVScalc_SQP.cpp

3.41 integrate::RSVSclass Class Reference

Public Attributes

- param::parameters **paramconf**
- tecplotfile **outSnake**
- snake **rsvsSnake**
- mesh **snakeMesh**
- mesh **voluMesh**
- triangulation **rsvsTri**
- RSVScalc **calcObj**

- std::ofstream **logFile**
- std::ofstream **coutFile**
- std::ofstream **cerrFile**

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

- incl/RSVSclass.hpp

3.42 selfint_event Class Reference

Public Attributes

- int **e_type**
- int **f_marker1**
- int **s_marker1**
- int **f_vertices1** [3]
- int **f_marker2**
- int **s_marker2**
- int **f_vertices2** [3]
- REAL **int_point** [3]

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

- modules/tetgen/tetgen.h

3.43 snake Class Reference

Public Member Functions

- void **disp** () const
- void **displight** () const
- bool **isready** () const
- void **PrepareForUse** (bool needOrder=true)
- void **Init** (mesh ∗snakemesh, int nSnax, int nEdge, int nSurf, int nVolu)
- void **reserve** (int nSnax, int nEdge, int nSurf, int nVolu)
- void **GetMaxIndex** (int ∗nVert, int ∗nEdge, int ∗nSurf, int ∗nVolu) const
- void **HashArray** ()
- void **HashArrayNM** ()
- void **HashParent** ()
- void **SetMaxIndex** ()
- void **SetMaxIndexNM** ()
- void **Concatenate** (const snake &other, int isInternal=0)
- bool **Check3D** () const
- void **MakeCompatible_inplace** (snake &other) const
- snake **MakeCompatible** (snake other) const
- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu)
- void **ChangeIndicesSnakeMesh** (int nVert, int nEdge, int nSurf, int nVolu)
- void **ForceCloseContainers** ()

- void **UpdateDistance** (double dt, double maxDstep=1.0)
- void **UpdateDistance** (const vector< double > &dt, double maxDstep=1.0)
- void **CalculateTimeStep** (vector< double > &dt, double dtDefault, double distDefault=1.0)
- void **SnaxImpactDetection** (vector< int > &isImpact)
- void **SnaxAlmostImpactDetection** (vector< int > &isImpact, double dDlim)
- void **UpdateCoord** ()
- void **Flip** ()
- grid::limits **Scale** (const grid::limits &newSize)
- void **OrderEdges** ()
- void **SetSnaxSurfs** ()
- void **OrientFaces** ()
- int **FindBlockSnakeMeshVerts** (vector< int > &vertBlock) const
- void **AssignInternalVerts** ()
- void **CheckConnectivity** () const
- void **VertIsIn** (int vertInd, bool isIn=true)
- void **VertIsIn** (vector< int > vertInd, bool isIn=true)
- bool **ReturnFlip** () const
- void **read** (FILE *fid)
- void **write** (FILE *fid) const
- int **read** (const char *str)
- int **write** (const char *str) const

Public Attributes

- snaxarray **snaxs**
- snaxedgearray **snaxedges**
- snaxsurfarray **snaxsurfs**
- mesh **snakeconn**
- mesh * **snakemesh** =NULL
- vector< bool > **isMeshVertIn**

Private Member Functions

- void **SetLastIndex** ()
- void **OrientSurfaceVolume** ()
- void **OrientEdgeSurface** ()

Private Attributes

- bool **is3D** =true
- bool **isFlipped** =false

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

- incl/snake.hpp
- src/snake/snake.cpp

3.44 param::snaking Class Reference

Parameters controlling tuning parameters for the stepping of the restricted surface.

`#include <parameters.hpp>`

Public Member Functions

- void **PrepareForUse** ()

Public Attributes

- double **arrivaltolerance**
- double **multiarrivaltolerance**
- double **snaxtimestep**
- double **snaxdiststep**
- int **initboundary**
- int **maxsteps**

3.44.1 Detailed Description

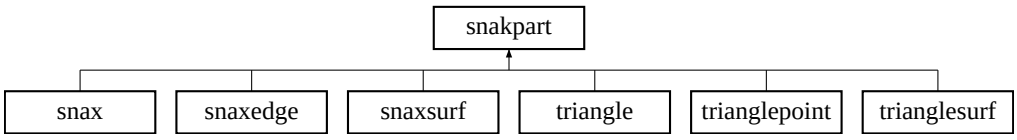
Parameters controlling tuning parameters for the stepping of the restricted surface.

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

- incl/parameters.hpp
- src/parameters.cpp

3.45 snakpart Class Reference

Inheritance diagram for snakpart:



Public Member Functions

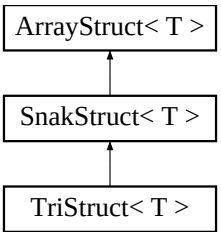
- virtual int **KeyParent** () const =0

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

- incl/arraystructures.hpp

3.46 SnakStruct< T > Class Template Reference

Inheritance diagram for SnakStruct< T >:



Public Member Functions

- int **findparent** (int key) const
- void **findsiblings** (int key, vector< int > &siblings) const
- int **countparent** (int key) const
- void **HashParent** ()
- void **DeHashParent** (const int pos)
- bool **memberIsHashParent** (const int pos) const
- void **Init** (int n)
- void **push_back** (T &newelem)
- void **clear** ()
- bool **checkready** ()
- void **ForceArrayReady** ()
- void **PrepareForUse** ()
- void **Concatenate** (const SnakStruct< T > &other)
- void **remove** (const vector< int > &sub)
- T & **operator[ ]** (const int a)

Protected Attributes

- unordered_multimap< int, int > **hashParent**
- int **isHashParent** =0

Friends

- class **snake**

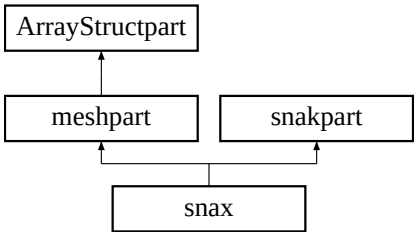
Additional Inherited Members

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

- incl/arraystructures.hpp
- incl/snakstruct_incl.cpp

3.47 snax Class Reference

Inheritance diagram for snax:



Public Member Functions

- void **disp** () const
- void **disptree** (const mesh &meshin, int n) const
- void **disptree** (const snake &snakein, int n) const
- int **Key** () const
- int **KeyParent** () const
- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu)
- void **ChangeIndicesSnakeMesh** (int nVert, int nEdge, int nSurf, int nVolu)
- void **PrepareForUse** ()
- bool **isready** (bool isInMesh) const
- void **read** (FILE ∗fid)
- void **write** (FILE ∗fid) const
- void **set** (int index, double d, double v, int fromvert, int tovert, int edgeind, int isfreeze, int orderedge)
- void **SwitchIndex** (int typeInd, int oldInd, int newInd)
- void **TightenConnectivity** ()

Public Attributes

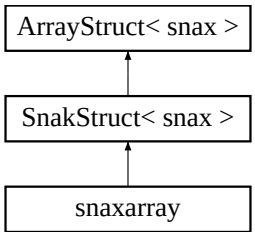
- double **d** =0.0
- double **v** =0.0
- int **fromvert** =0
- int **tovert** =0
- int **edgeind** =0
- int **isfreeze** =0
- int **orderedge** =0

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

- incl/snake.hpp
- src/snake/snake.cpp

3.48 snaxarray Class Reference

Inheritance diagram for snaxarray:



Public Member Functions

- void **ReorderOnEdge** ()
- void **OrderOnEdge** ()
- void **CalculateTimeStepOnEdge** (vector< double > &dt, vector< bool > &isSnaxDone, int edgeInd)
- void **DetectImpactOnEdge** (vector< int > &isImpact, vector< bool > &isSnaxDone, int edgeInd)
- bool **checkready** ()
- void **ForceArrayReady** ()
- void **PrepareForUse** ()
- void **Concatenate** (const snaxarray &other)
- snax & **operator[ ]** (const int a)

Protected Attributes

- int **isOrderedOnEdge** =0

Friends

- class **snake**
- void **SpawnArrivedSnaxeIsDir** (snake &fullsnake, snake &partSnake, const vector< int > &isImpact, int dir)

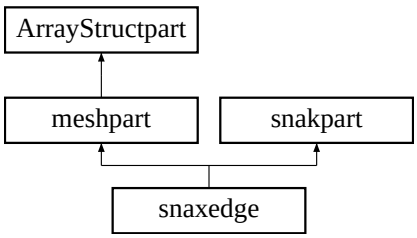
Additional Inherited Members

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

- incl/snake.hpp
- src/snake/snake.cpp

3.49 snaxedge Class Reference

Inheritance diagram for snaxedge:



Public Member Functions

- void **PrepareForUse** ()
- void **disp** () const
- void **disptree** (const mesh &meshin, int n) const
- void **disptree** (const snake &snakein, int n) const
- int **Key** () const
- int **KeyParent** () const
- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu)
- void **ChangeIndicesSnakeMesh** (int nVert, int nEdge, int nSurf, int nVolu)
- bool **isready** (bool isInMesh) const
- void **read** (FILE ∗fid)
- void **write** (FILE ∗fid) const
- void **SwitchIndex** (int typeInd, int oldInd, int newInd)
- void **TightenConnectivity** ()

Public Attributes

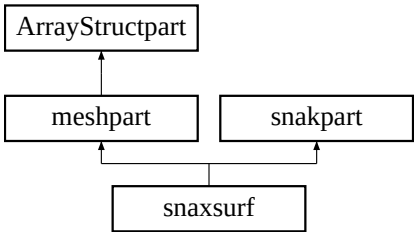
- int **surfind** =0
- coordvec **normvector**

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

- incl/snake.hpp
- src/snake/snake.cpp

3.50 snaxsurf Class Reference

Inheritance diagram for snaxsurf:



Public Member Functions

- void **PrepareForUse** ()
- void **disp** () const
- void **disptree** (const mesh &meshin, int n) const
- void **disptree** (const snake &snakein, int n) const
- int **Key** () const
- int **KeyParent** () const
- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu)
- void **ChangeIndicesSnakeMesh** (int nVert, int nEdge, int nSurf, int nVolu)
- bool **isready** (bool isInMesh) const
- void **read** (FILE ∗fid)
- void **write** (FILE ∗fid) const
- void **SwitchIndex** (int typeInd, int oldInd, int newInd)
- void **TightenConnectivity** ()

Public Attributes

- int **voluind** =0
- coordvec **normvector**

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

- incl/snake.hpp
- src/snake/snake.cpp

3.51 dbg::StackFrame Struct Reference

Public Attributes

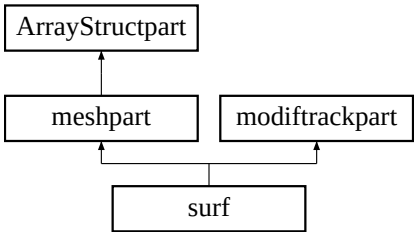
- DWORD64 **address**
- std::string **name**
- std::string **module**
- unsigned int **line**
- std::string **file**

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

- modules/external/data.h

3.52 surf Class Reference

Inheritance diagram for surf:



Public Member Functions

- void **disp** () const
- void **disptree** (const mesh &meshin, int n) const
- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu)
- void **PrepareForUse** ()
- bool **isready** (bool isInMesh) const
- void **read** (FILE ∗fid)
- void **write** (FILE ∗fid) const
- int **OrderEdges** (mesh ∗meshin)
- int **SplitSurface** (mesh &meshin, const vector< int > &fullEdgeInd)
- void **OrderedVerts** (const mesh ∗meshin, vector< int > &vertList) const
- void **TightenConnectivity** ()
- void **FlipVolus** ()
- bool **edgeconneq** (const surf &other, bool recurse=true) const
- **surf** (const surf &oldSurf)
- void **operator=** (const surf ∗other)
- int **Key** () const

Public Attributes

- friend **surfarray**
- double **fill**
- double **target**
- double **error**
- double **area**
- vector< int > **edgeind**
- vector< int > **voluind**

Protected Attributes

- bool **isordered**

Friends

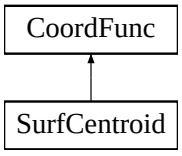
- class **mesh**

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

- incl/mesh.hpp
- src/grid/mesh.cpp

3.53 SurfCentroid Class Reference

Inheritance diagram for SurfCentroid:



Public Member Functions

- void **Disp** ()
- void **Calc** () override
- void **assigncentroid** (const vector< double > &vecin)
- **SurfCentroid** (int a)

Protected Attributes

- vector< double > **centroid**
- double **edgeLength** =0.0
- vector< vector< double > const * > **coords**
- double **fun**
- ArrayVec< double > **jac**
- ArrayVec< double > **hes**
- int **nCoord**

Additional Inherited Members

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

- incl/RSVSmath.hpp
- src/rsvs/RSVSmath.cpp

3.54 tecplotfile Class Reference

Public Member Functions

- int **OpenFile** (const char *str, const char *mode="w")
- void **CloseFile** ()
- int **ZoneNum** () const
- int **PrintMesh** (const mesh &meshout, int strandID=0, double timeStep=0, int forceOutType=0, const vector< int > &vertList={})
- int **PrintSnakeInternalPts** (const snake &snakein, int strandID=0, double timeStep=0)
- int **VolDataBlock** (const mesh &meshout, int nVert, int nVolu, int nVertDat, const std::vector< int > &voluList={}, const std::vector< int > &vertList={})
- int **SurfDataBlock** (const mesh &meshout, int nVert, int nSurf, int nVertDat)
- int **LineDataBlock** (const mesh &meshout, int nVert, int nEdge, int nVertDat, int nCellDat)
- int **VertDataBlock** (const mesh &meshout, int nVert, int nVertDat, int nCellDat, const vector< int > &vertList={})
- int **VolFaceMap** (const mesh &meshout, int nSurf)
- int **VolFaceMap** (const mesh &meshout, const std::vector< int > &surfList, const std::vector< int > &voluList, const std::vector< int > &vertList)
- int **SurfFaceMap** (const mesh &meshout, int nEdge)
- int **LineFaceMap** (const mesh &meshout, int nEdge)
- int **PrintVolumeDat** (const mesh &meshout, int shareZone, int strandID, double timeStep)
- int **DefShareZoneVolume** (int shareZone, int nVertDat)
- int **VolDataBlock** (const triangulation &triout, triarray triangulation::*mp, int nVert, int nVolu, int nVertDat)
- int **SurfDataBlock** (const triangulation &triout, triarray triangulation::*mp, int nVert, int nSurf, int nVertDat)

- int **LineDataBlock** (const triangulation &triout, triarray triangulation::∗mp, int nVert, int nEdge, int nVertDat, int nCellDat)
- int **LineDataBlock** (const triangulation &triout, triarray triangulation::∗mp, int nVert, int nEdge, int nVertDat, int nCellDat, const vector< int > &triList)
- int **SurfFaceMap** (const triangulation &triout, triarray triangulation::∗mp)
- int **LineFaceMap** (const triangulation &triout, triarray triangulation::∗mp)
- int **LineFaceMap** (const vector< int > &triList)
- int **VolFaceMap** (const triangulation &triout, triarray triangulation::∗mp, int nSurf)
- int **PrintTriangulation** (const triangulation &triout, triarray triangulation::∗mp, int strandID=0, double time↩ Step=0, int forceOutType=0, const vector< int > &triList={})
- int **VolDataBlock** (const triangulation &triout, trisurfarray triangulation::∗mp, int nVert, int nVolu, int nVertDat)
- int **SurfDataBlock** (const triangulation &triout, trisurfarray triangulation::∗mp, int nVert, int nSurf, int nVertDat)
- int **LineDataBlock** (const triangulation &triout, trisurfarray triangulation::∗mp, int nVert, int nEdge, int nVert↩ Dat, int nCellDat)
- int **SurfFaceMap** (const triangulation &triout, trisurfarray triangulation::∗mp)
- int **LineFaceMap** (const triangulation &triout, trisurfarray triangulation::∗mp)
- int **VolFaceMap** (const triangulation &triout, trisurfarray triangulation::∗mp, int nSurf)
- int **PrintTriangulation** (const triangulation &triout, trisurfarray triangulation::∗mp, int strandID=0, double timeStep=0, int forceOutType=0)
- int **SnakeDataBlock** (const snake &snakeout, int nVert, int nVertDat)
- int **PrintSnake** (const snake &snakeout, int strandID=0, double timeStep=0, int forceOutType=0, const vector< int > &vertList={})
- void **ZoneHeaderPolyhedron** (int nVert, int nVolu, int nSurf, int totNumFaceNode, int nVertDat, int nCellDat)
- void **ZoneHeaderPolygon** (int nVert, int nEdge, int nSurf, int nVertDat, int nCellDat)
- void **ZoneHeaderFelineseg** (int nVert, int nEdge, int nVertDat, int nCellDat)
- void **ZoneHeaderOrdered** (int nVert, int nVertDat, int nCellDat)
- void **ZoneHeaderPolyhedronSnake** (int nVert, int nVolu, int nSurf, int totNumFaceNode, int nVertDat, int nCellDat)
- void **ZoneHeaderPolygonSnake** (int nVert, int nEdge, int nSurf, int nVertDat, int nCellDat)
- void **ZoneHeaderFelinesegSnake** (int nVert, int nEdge, int nVertDat, int nCellDat)
- void **ZoneHeaderOrderedSnake** (int nVert, int nVertDat, int nCellDat)
- void **NewZone** ()
- void **StrandTime** (int strandID, double timeStep)
- int **Print** (const char ∗format,...)
- void **ResetLine** ()

Private Attributes

- FILE ∗ **fid**
- int **lengthLine**
- int **nZones** =0

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

- incl/postprocessing.hpp
- src/postprocessing.cpp

3.55 tetgenbehavior Class Reference

Public Types

- enum **objecttype** {
  NODES, **POLY**, **OFF**, **PLY**,
  STL, **MEDIT**, **VTK**, **MESH**,
  NEU_MESH }

Public Member Functions

- void **syntax** ()
- void **usage** ()
- bool **parse_commandline** (int argc, const char ∗∗argv)
- bool **parse_commandline** (const char ∗switches)

Public Attributes

- int **plc**
- int **psc**
- int **refine**
- int **quality**
- int **nobisect**
- int **coarsen**
- int **weighted**
- int **brio_hilbert**
- int **incrflip**
- int **flipinsert**
- int **metric**
- int **varvolume**
- int **fixedvolume**
- int **regionattrib**
- int **cdtrefine**
- int **use_equatorial_lens**
- int **insertaddpoints**
- int **diagnose**
- int **convex**
- int **nomergefacet**
- int **nomergevertex**
- int **noexact**
- int **nostaticfilter**
- int **zeroindex**
- int **facesout**
- int **edgesout**
- int **neighout**
- int **voroout**
- int **meditview**
- int **vtkview**
- int **nobound**
- int **nonodewritten**
- int **noelewritten**
- int **nofacewritten**
- int **noiterationnum**
- int **nojettison**
- int **docheck**
- int **quiet**
- int **verbose**
- int **vertexperblock**
- int **tetrahedraperblock**
- int **shellfaceperblock**
- int **nobisect_nomerge**
- int **supsteiner_level**
- int **addsteiner_algo**

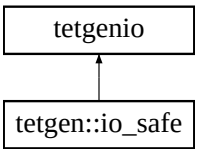
- int **coarsen_param**
- int **weighted_param**
- int **fliplinklevel**
- int **flipstarsize**
- int **fliplinklevelinc**
- int **reflevel**
- int **optlevel**
- int **optscheme**
- int **delmaxfliplevel**
- int **order**
- int **reversetetori**
- int **steinerleft**
- int **no_sort**
- int **hilbert_order**
- int **hilbert_limit**
- int **brio_threshold**
- REAL **brio_ratio**
- REAL **facet_separate_ang_tol**
- REAL **facet_overlap_ang_tol**
- REAL **facet_small_ang_tol**
- REAL **maxvolume**
- REAL **minratio**
- REAL **mindihedral**
- REAL **optmaxdihedral**
- REAL **optminsmtdihed**
- REAL **optminslidihed**
- REAL **epsilon**
- REAL **coarsen_percent**
- char **commandline** [1024]
- char **infilename** [1024]
- char **outfilename** [1024]
- char **addinfilename** [1024]
- char **bgmeshfilename** [1024]
- int **hole_mesh**
- char **hole_mesh_filename** [1024]
- int **apply_flow_bc**
- enum tetgenbehavior::objecttype **object**

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

- modules/tetgen/tetgen.h
- modules/tetgen/tetgen.cpp
- modules/tetgen/tetgen.cxx

3.56 tetgenio Class Reference

Inheritance diagram for tetgenio:



Classes

- struct facet
- struct pointparam
- struct polygon
- struct voroedge
- struct vorofacet

Public Types

- typedef REAL(∗ **GetVertexParamOnEdge**) (void ∗, int, int)
- typedef void(∗ **GetSteinerOnEdge**) (void ∗, int, REAL, REAL ∗)
- typedef void(∗ **GetVertexParamOnFace**) (void ∗, int, int, REAL ∗)
- typedef void(∗ **GetEdgeSteinerParamOnFace**) (void ∗, int, REAL, int, REAL ∗)
- typedef void(∗ **GetSteinerOnFace**) (void ∗, int, REAL ∗, REAL ∗)
- typedef bool(∗ **TetSizeFunc**) (REAL ∗, REAL ∗, REAL ∗, REAL ∗, REAL ∗, REAL)

Public Member Functions

- bool **load_node_call** (FILE ∗infile, int markers, int uvflag, char ∗)
- bool **load_node** (const char ∗)
- bool **load_edge** (const char ∗)
- bool **load_face** (const char ∗)
- bool **load_tet** (const char ∗)
- bool **load_vol** (const char ∗)
- bool **load_var** (const char ∗)
- bool **load_mtr** (const char ∗)
- bool **load_pbc** (const char ∗)
- bool **load_poly** (const char ∗)
- bool **load_off** (const char ∗)
- bool **load_ply** (const char ∗)
- bool **load_stl** (const char ∗)
- bool **load_vtk** (const char ∗)
- bool **load_medit** (const char ∗, int)
- bool **load_neumesh** (const char ∗, int)
- bool **load_plc** (const char ∗, int)
- bool **load_tetmesh** (const char ∗, int)
- void **save_nodes** (const char ∗)
- void **save_elements** (const char ∗)
- void **save_faces** (const char ∗)
- void **save_edges** (const char ∗)
- void **save_neighbors** (const char ∗)
- void **save_poly** (const char ∗)
- void **save_faces2smesh** (char ∗)
- char ∗ **readline** (char ∗string, FILE ∗infile, int ∗linenumber)
- char ∗ **findnextfield** (char ∗string)
- char ∗ **readnumberline** (char ∗string, FILE ∗infile, char ∗infilename)
- char ∗ **findnextnumber** (char ∗string)
- void **initialize** ()
- void **deinitialize** ()

Static Public Member Functions

- static void **init** (polygon *p)
- static void **init** (facet *f)

Public Attributes

- int **firstnumber**
- int **mesh_dim**
- int **useindex**
- REAL * **pointlist**
- REAL * **pointattributelist**
- REAL * **pointmtrlist**
- int * **pointmarkerlist**
- int * **point2tetlist**
- pointparam * **pointparamlist**
- int **numberofpoints**
- int **numberofpointattributes**
- int **numberofpointmtrs**
- int * **tetrahedronlist**
- REAL * **tetrahedronattributelist**
- REAL * **tetrahedronvolumelist**
- int * **neighborlist**
- int * **tet2facelist**
- int * **tet2edgelist**
- int **numberoftetrahedra**
- int **numberofcorners**
- int **numberoftetrahedronattributes**
- facet * **facetlist**
- int * **facetmarkerlist**
- int **numberoffacets**
- REAL * **holelist**
- int **numberofholes**
- REAL * **regionlist**
- int **numberofregions**
- REAL * **facetconstraintlist**
- int **numberoffacetconstraints**
- REAL * **segmentconstraintlist**
- int **numberofsegmentconstraints**
- int * **trifacelist**
- int * **trifacemarkerlist**
- int * **o2facelist**
- int * **face2tetlist**
- int * **face2edgelist**
- int **numberoftrifaces**
- int * **edgelist**
- int * **edgemarkerlist**
- int * **o2edgelist**
- int * **edge2tetlist**
- int **numberofedges**
- REAL * **vpointlist**
- voroedge * **vedgelist**
- vorofacet * **vfacetlist**
- int ** **vcelllist**

- int **numberofvpoints**
- int **numberofvedges**
- int **numberofvfacets**
- int **numberofvcells**
- void * **geomhandle**
- GetVertexParamOnEdge **getvertexparamonedge**
- GetSteinerOnEdge **getsteineronedge**
- GetVertexParamOnFace **getvertexparamonface**
- GetEdgeSteinerParamOnFace **getedgesteinerparamonface**
- GetSteinerOnFace **getsteineronface**
- TetSizeFunc **tetunsuitable**

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

- modules/tetgen/tetgen.h
- modules/tetgen/tetgen.cpp
- modules/tetgen/tetgen.cxx

3.57 tetgenmesh Class Reference

Classes

- class arraypool
- class badface
- class face
- class flipconstraints
- class insertvertexflags
- class memorypool
- class optparameters
- class triface

Public Types

- enum **verttype** {
  UNUSEDVERTEX, **DUPLICATEDVERTEX**, **RIDGEVERTEX**, **ACUTEVERTEX**,
  FACETVERTEX, **VOLVERTEX**, **FREESEGVERTEX**, **FREEFACETVERTEX**,
  FREEVOLVERTEX, **NREGULARVERTEX**, **DEADVERTEX** }
- enum **interresult** {
  DISJOINT, **INTERSECT**, **SHAREVERT**, **SHAREEDGE**,
  SHAREFACE, **TOUCHEDGE**, **TOUCHFACE**, **ACROSSVERT**,
  ACROSSEDGE, **ACROSSFACE** }
- enum **locateresult** {
  UNKNOWN, **OUTSIDE**, **INTETRAHEDRON**, **ONFACE**,
  ONEDGE, **ONVERTEX**, **ENCVERTEX**, **ENCSEGMENT**,
  ENCSUBFACE, **NEARVERTEX**, **NONREGULAR**, **INSTAR**,
  BADELEMENT }
- typedef REAL ** **tetrahedron**
- typedef REAL ** **shellface**
- typedef REAL * **point**

Public Member Functions

- void **inittables** ()
- tetrahedron **encode** (triface &t)
- tetrahedron **encode2** (tetrahedron *ptr, int ver)
- void **decode** (tetrahedron ptr, triface &t)
- void **bond** (triface &t1, triface &t2)
- void **dissolve** (triface &t)
- void **esym** (triface &t1, triface &t2)
- void **esymself** (triface &t)
- void **enext** (triface &t1, triface &t2)
- void **enextself** (triface &t)
- void **eprev** (triface &t1, triface &t2)
- void **eprevself** (triface &t)
- void **enextesym** (triface &t1, triface &t2)
- void **enextesymself** (triface &t)
- void **eprevesym** (triface &t1, triface &t2)
- void **eprevesymself** (triface &t)
- void **eorgoppo** (triface &t1, triface &t2)
- void **eorgoppoself** (triface &t)
- void **edestoppo** (triface &t1, triface &t2)
- void **edestoppoself** (triface &t)
- void **fsym** (triface &t1, triface &t2)
- void **fsymself** (triface &t)
- void **fnext** (triface &t1, triface &t2)
- void **fnextself** (triface &t)
- point **org** (triface &t)
- point **dest** (triface &t)
- point **apex** (triface &t)
- point **oppo** (triface &t)
- void **setorg** (triface &t, point p)
- void **setdest** (triface &t, point p)
- void **setapex** (triface &t, point p)
- void **setoppo** (triface &t, point p)
- REAL **elemattribute** (tetrahedron *ptr, int attnum)
- void **setelemattribute** (tetrahedron *ptr, int attnum, REAL value)
- REAL **volumebound** (tetrahedron *ptr)
- void **setvolumebound** (tetrahedron *ptr, REAL value)
- int **elemindex** (tetrahedron *ptr)
- void **setelemindex** (tetrahedron *ptr, int value)
- int **elemmarker** (tetrahedron *ptr)
- void **setelemmarker** (tetrahedron *ptr, int value)
- void **infect** (triface &t)
- void **uninfect** (triface &t)
- bool **infected** (triface &t)
- void **marktest** (triface &t)
- void **unmarktest** (triface &t)
- bool **marktested** (triface &t)
- void **markface** (triface &t)
- void **unmarkface** (triface &t)
- bool **facemarked** (triface &t)
- void **markedge** (triface &t)
- void **unmarkedge** (triface &t)
- bool **edgemarked** (triface &t)
- void **marktest2** (triface &t)

- void **unmarktest2** (triface &t)
- bool **marktest2ed** (triface &t)
- int **elemcounter** (triface &t)
- void **setelemcounter** (triface &t, int value)
- void **increaseelemcounter** (triface &t)
- void **decreaseelemcounter** (triface &t)
- bool **ishulltet** (triface &t)
- bool **isdeadtet** (triface &t)
- void **sdecode** (shellface sptr, face &s)
- shellface **sencode** (face &s)
- shellface **sencode2** (shellface *sh, int shver)
- void **spivot** (face &s1, face &s2)
- void **spivotself** (face &s)
- void **sbond** (face &s1, face &s2)
- void **sbond1** (face &s1, face &s2)
- void **sdissolve** (face &s)
- point **sorg** (face &s)
- point **sdest** (face &s)
- point **sapex** (face &s)
- void **setsorg** (face &s, point pointptr)
- void **setsdest** (face &s, point pointptr)
- void **setsapex** (face &s, point pointptr)
- void **sesym** (face &s1, face &s2)
- void **sesymself** (face &s)
- void **senext** (face &s1, face &s2)
- void **senextself** (face &s)
- void **senext2** (face &s1, face &s2)
- void **senext2self** (face &s)
- REAL **areabound** (face &s)
- void **setareabound** (face &s, REAL value)
- int **shellmark** (face &s)
- void **setshellmark** (face &s, int value)
- void **sinfect** (face &s)
- void **suninfect** (face &s)
- bool **sinfected** (face &s)
- void **smarktest** (face &s)
- void **sunmarktest** (face &s)
- bool **smarktested** (face &s)
- void **smarktest2** (face &s)
- void **sunmarktest2** (face &s)
- bool **smarktest2ed** (face &s)
- void **smarktest3** (face &s)
- void **sunmarktest3** (face &s)
- bool **smarktest3ed** (face &s)
- void **setfacetindex** (face &f, int value)
- int **getfacetindex** (face &f)
- void **tsbond** (triface &t, face &s)
- void **tsdissolve** (triface &t)
- void **stdissolve** (face &s)
- void **tspivot** (triface &t, face &s)
- void **stpivot** (face &s, triface &t)
- void **tssbond1** (triface &t, face &seg)
- void **sstbond1** (face &s, triface &t)
- void **tssdissolve1** (triface &t)
- void **sstdissolve1** (face &s)

- void **tsspivot1** (triface &t, face &s)
- void **sstpivot1** (face &s, triface &t)
- void **ssbond** (face &s, face &edge)
- void **ssbond1** (face &s, face &edge)
- void **ssdissolve** (face &s)
- void **sspivot** (face &s, face &edge)
- int **pointmark** (point pt)
- void **setpointmark** (point pt, int value)
- enum verttype **pointtype** (point pt)
- void **setpointtype** (point pt, enum verttype value)
- int **pointgeomtag** (point pt)
- void **setpointgeomtag** (point pt, int value)
- REAL **pointgeomuv** (point pt, int i)
- void **setpointgeomuv** (point pt, int i, REAL value)
- void **pinfect** (point pt)
- void **puninfect** (point pt)
- bool **pinfected** (point pt)
- void **pmarktest** (point pt)
- void **punmarktest** (point pt)
- bool **pmarktested** (point pt)
- void **pmarktest2** (point pt)
- void **punmarktest2** (point pt)
- bool **pmarktest2ed** (point pt)
- void **pmarktest3** (point pt)
- void **punmarktest3** (point pt)
- bool **pmarktest3ed** (point pt)
- tetrahedron **point2tet** (point pt)
- void **setpoint2tet** (point pt, tetrahedron value)
- shellface **point2sh** (point pt)
- void **setpoint2sh** (point pt, shellface value)
- point **point2ppt** (point pt)
- void **setpoint2ppt** (point pt, point value)
- tetrahedron **point2bgmtet** (point pt)
- void **setpoint2bgmtet** (point pt, tetrahedron value)
- void **setpointinsradius** (point pt, REAL value)
- REAL **getpointinsradius** (point pt)
- bool **issteinerpoint** (point pt)
- void **point2tetorg** (point pt, triface &t)
- void **point2shorg** (point pa, face &s)
- point **farsorg** (face &seg)
- point **farsdest** (face &seg)
- void **tetrahedrondealloc** (tetrahedron *)
- tetrahedron * **tetrahedrontraverse** ()
- tetrahedron * **alltetrahedrontraverse** ()
- void **shellfacedealloc** (memorypool *, shellface *)
- shellface * **shellfacetraverse** (memorypool *)
- void **pointdealloc** (point)
- point **pointtraverse** ()
- void **makeindex2pointmap** (point *&)
- void **makepoint2submap** (memorypool *, int *&, face *&)
- void **maketetrahedron** (triface *)
- void **makeshellface** (memorypool *, face *)
- void **makepoint** (point *, enum verttype)
- void **initializepools** ()
- REAL **insphere_s** (REAL *, REAL *, REAL *, REAL *, REAL *)

- REAL **orient4d_s** (REAL *, REAL *, REAL *, REAL *, REAL *, REAL, REAL, REAL, REAL, REAL)
- int **tri_edge_2d** (point, point, point, point, point, point, int, int *, int *)
- int **tri_edge_tail** (point, point, point, point, point, point, REAL, REAL, int, int *, int *)
- int **tri_edge_test** (point, point, point, point, point, point, int, int *, int *)
- int **tri_edge_inter_tail** (point, point, point, point, point, REAL, REAL)
- int **tri_tri_inter** (point, point, point, point, point, point)
- REAL **dot** (REAL *v1, REAL *v2)
- void **cross** (REAL *v1, REAL *v2, REAL *n)
- bool **lu_decmp** (REAL lu[4][4], int n, int *ps, REAL *d, int N)
- void **lu_solve** (REAL lu[4][4], int n, int *ps, REAL *b, int N)
- REAL **incircle3d** (point pa, point pb, point pc, point pd)
- REAL **orient3dfast** (REAL *pa, REAL *pb, REAL *pc, REAL *pd)
- REAL **norm2** (REAL x, REAL y, REAL z)
- REAL **distance** (REAL *p1, REAL *p2)
- void **facenormal** (point pa, point pb, point pc, REAL *n, int pivot, REAL *lav)
- REAL **shortdistance** (REAL *p, REAL *e1, REAL *e2)
- REAL **triarea** (REAL *pa, REAL *pb, REAL *pc)
- REAL **interiorangle** (REAL *o, REAL *p1, REAL *p2, REAL *n)
- void **projpt2edge** (REAL *p, REAL *e1, REAL *e2, REAL *prj)
- void **projpt2face** (REAL *p, REAL *f1, REAL *f2, REAL *f3, REAL *prj)
- bool **tetalldihedral** (point, point, point, point, REAL *, REAL *, REAL *)
- void **tetallnormal** (point, point, point, point, REAL N[4][3], REAL *volume)
- REAL **tetaspectratio** (point, point, point, point)
- bool **circumsphere** (REAL *, REAL *, REAL *, REAL *, REAL *cent, REAL *radius)
- bool **orthosphere** (REAL *, REAL *, REAL *, REAL *, REAL, REAL, REAL, REAL, REAL *, REAL *)
- void **tetcircumcenter** (point tetorg, point tetdest, point tetfapex, point tettapex, REAL *circumcenter, REAL *radius)
- void **planelineint** (REAL *, REAL *, REAL *, REAL *, REAL *, REAL *, REAL *)
- int **linelineint** (REAL *, REAL *, REAL *, REAL *, REAL *, REAL *, REAL *, REAL *)
- REAL **tetprismvol** (REAL *pa, REAL *pb, REAL *pc, REAL *pd)
- bool **calculateabovepoint** (arraypool *, point *, point *, point *)
- void **calculateabovepoint4** (point, point, point, point)
- void **report_overlapping_facets** (face *, face *, REAL dihedang=0.0)
- int **report_selfint_edge** (point, point, face *sedge, triface *searchtet, enum interresult)
- int **report_selfint_face** (point, point, point, face *sface, triface *iedge, int intflag, int *types, int *poss)
- void **flip23** (triface *, int, flipconstraints *fc)
- void **flip32** (triface *, int, flipconstraints *fc)
- void **flip41** (triface *, int, flipconstraints *fc)
- int **flipnm** (triface *, int n, int level, int, flipconstraints *fc)
- int **flipnm_post** (triface *, int n, int nn, int, flipconstraints *fc)
- int **insertpoint** (point, triface *, face *, face *, insertvertexflags *)
- void **insertpoint_abort** (face *, insertvertexflags *)
- void **transfernodes** ()
- void **hilbert_init** (int n)
- int **hilbert_split** (point *vertexarray, int arraysize, int gc0, int gc1, REAL, REAL, REAL, REAL, REAL, REAL)
- void **hilbert_sort3** (point *vertexarray, int arraysize, int e, int d, REAL, REAL, REAL, REAL, REAL, REAL, int depth)
- void **brio_multiscale_sort** (point *, int, int threshold, REAL ratio, int *depth)
- unsigned long **randomnation** (unsigned int choices)
- void **randomsample** (point searchpt, triface *searchtet)
- enum locateresult **locate** (point searchpt, triface *searchtet, int chkencflag=0)
- void **flippush** (badface *&, triface *)
- int **incrementalflip** (point newpt, int, flipconstraints *fc)
- void **initialdelaunay** (point pa, point pb, point pc, point pd)
- void **incrementaldelaunay** (clock_t &)

- void **flipshpush** (face ∗)
- void **flip22** (face ∗, int, int)
- void **flip31** (face ∗, int)
- long **lawsonflip** ()
- int **sinsertvertex** (point newpt, face ∗, face ∗, int iloc, int bowywat, int)
- int **sremovevertex** (point delpt, face ∗, face ∗, int lawson)
- enum locateresult **slocate** (point, face ∗, int, int, int)
- enum interresult **sscoutsegment** (face ∗, point, int, int, int)
- void **scarveholes** (int, REAL ∗)
- int **triangulate** (int, arraypool ∗, arraypool ∗, int, REAL ∗)
- void **unifysegments** ()
- void **identifyinputedges** (point ∗)
- void **mergefacets** ()
- void **removesmallangles** ()
- void **meshsurface** ()
- void **interecursive** (shellface ∗∗subfacearray, int arraysize, int axis, REAL, REAL, REAL, REAL, REAL, REAL, int ∗internum)
- void **detectinterfaces** ()
- void **makesegmentendpointsmap** ()
- enum interresult **finddirection** (triface ∗searchtet, point endpt)
- enum interresult **scoutsegment** (point, point, face ∗, triface ∗, point ∗, arraypool ∗)
- int **getsteinerptonsegment** (face ∗seg, point refpt, point steinpt)
- void **delaunizesegments** ()
- int **scoutsubface** (face ∗searchsh, triface ∗searchtet, int shflag)
- void **formregion** (face ∗, arraypool ∗, arraypool ∗, arraypool ∗)
- int **scoutcrossedge** (triface &crosstet, arraypool ∗, arraypool ∗)
- bool **formcavity** (triface ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗)
- void **delaunizecavity** (arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗)
- bool **fillcavity** (arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗, triface ∗crossedge)
- void **carvecavity** (arraypool ∗, arraypool ∗, arraypool ∗)
- void **restorecavity** (arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗)
- void **flipcertify** (triface ∗chkface, badface ∗∗pqueue, point, point, point)
- void **flipinsertfacet** (arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗)
- int **insertpoint_cdt** (point, triface ∗, face ∗, face ∗, insertvertexflags ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗)
- void **refineregion** (face &, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗, arraypool ∗)
- void **constrainedfacets** ()
- void **constraineddelaunay** (clock_t &)
- int **checkflipeligibility** (int fliptype, point, point, point, point, point, int level, int edgepivot, flipconstraints ∗fc)
- int **removeedgebyflips** (triface ∗, flipconstraints ∗)
- int **removefacebyflips** (triface ∗, flipconstraints ∗)
- int **recoveredgebyflips** (point, point, face ∗, triface ∗, int fullsearch)
- int **add_steinerpt_in_schoenhardtpoly** (triface ∗, int, int chkencflag)
- int **add_steinerpt_in_segment** (face ∗, int searchlevel)
- int **addsteiner4recoversegment** (face ∗, int)
- int **recoversegments** (arraypool ∗, int fullsearch, int steinerflag)
- int **recoverfacebyflips** (point, point, point, face ∗, triface ∗)
- int **recoversubfaces** (arraypool ∗, int steinerflag)
- int **getvertexstar** (int, point searchpt, arraypool ∗, arraypool ∗, arraypool ∗)
- int **getedge** (point, point, triface ∗)
- int **reduceedgesatvertex** (point startpt, arraypool ∗endptlist)
- int **removevertexbyflips** (point steinerpt)
- int **suppressbdrysteinerpoint** (point steinerpt)
- int **suppresssteinerpoints** ()
- void **recoverboundary** (clock_t &)

- void **carveholes** ()
- void **reconstructmesh** ()
- int **search_face** (point p0, point p1, point p2, triface &tetloop)
- int **search_edge** (point p0, point p1, triface &tetloop)
- int **scoutpoint** (point, triface *, int randflag)
- REAL **getpointmeshsize** (point, triface *, int iloc)
- void **interpolatemeshsize** ()
- void **out_points_to_cells_map** ()
- void **insertconstrainedpoints** (point *insertarray, int arylen, int rejflag)
- void **insertconstrainedpoints** (tetgenio *addio)
- void **collectremovepoints** (arraypool *remptlist)
- void **meshcoarsening** ()
- void **makefacetverticesmap** ()
- int **segsegadjacent** (face *, face *)
- int **segfacetadjacent** (face *checkseg, face *checksh)
- int **facetfacetadjacent** (face *, face *)
- void **save_segmentpoint_insradius** (point segpt, point parentpt, REAL r)
- void **save_facetpoint_insradius** (point facpt, point parentpt, REAL r)
- void **enqueuesubface** (memorypool *, face *)
- void **enqueuetetrahedron** (triface *)
- int **checkseg4encroach** (point pa, point pb, point checkpt)
- int **checkseg4split** (face *chkseg, point &, int &)
- int **splitsegment** (face *splitseg, point encpt, REAL, point, point, int, int)
- void **repairencsegs** (int chkencflag)
- int **checkfac4encroach** (point, point, point, point checkpt, REAL *, REAL *)
- int **checkfac4split** (face *chkfac, point &encpt, int &qflag, REAL *ccent)
- int **splitsubface** (face *splitfac, point, point, int qflag, REAL *ccent, int)
- void **repairencfacs** (int chkencflag)
- int **checktet4split** (triface *chktet, int &qflag, REAL *ccent)
- int **splittetrahedron** (triface *splittet, int qflag, REAL *ccent, int)
- void **repairbadtets** (int chkencflag)
- void **delaunayrefinement** ()
- long **lawsonflip3d** (flipconstraints *fc)
- void **recoverdelaunay** ()
- int **gettetrahedron** (point, point, point, point, triface *)
- long **improvequalitybyflips** ()
- int **smoothpoint** (point smtpt, arraypool *, int ccw, optparameters *opm)
- long **improvequalitybysmoothing** (optparameters *opm)
- int **splitsliver** (triface *, REAL, int)
- long **removeslivers** (int)
- void **optimizemesh** ()
- int **checkmesh** (int topoflag)
- int **checkshells** ()
- int **checksegments** ()
- int **checkdelaunay** (int perturb=1)
- int **checkregular** (int)
- int **checkconforming** (int)
- void **printfcomma** (unsigned long n)
- void **qualitystatistics** ()
- void **memorystatistics** ()
- void **statistics** ()
- void **jettisonnodes** ()
- void **highorder** ()
- void **indexelements** ()
- void **numberedges** ()

- void **outnodes** (tetgenio ∗)
- void **outmetrics** (tetgenio ∗)
- void **outelements** (tetgenio ∗)
- void **outfaces** (tetgenio ∗)
- void **outhullfaces** (tetgenio ∗)
- void **outsubfaces** (tetgenio ∗)
- void **outedges** (tetgenio ∗)
- void **outsubsegments** (tetgenio ∗)
- void **outneighbors** (tetgenio ∗)
- void **outvoronoi** (tetgenio ∗)
- void **outsmesh** (char ∗)
- void **outmesh2medit** (char ∗)
- void **outmesh2vtk** (char ∗)
- void **initializetetgenmesh** ()
- void **freememory** ()

Public Attributes

- tetgenio ∗ **in**
- tetgenio ∗ **addin**
- tetgenbehavior ∗ **b**
- tetgenmesh ∗ **bgm**
- memorypool ∗ **tetrahedrons**
- memorypool ∗ **subfaces**
- memorypool ∗ **subsegs**
- memorypool ∗ **points**
- memorypool ∗ **tet2subpool**
- memorypool ∗ **tet2segpool**
- memorypool ∗ **badtetrahedrons**
- memorypool ∗ **badsubfacs**
- memorypool ∗ **badsubsegs**
- memorypool ∗ **flippool**
- arraypool ∗ **unflipqueue**
- badface ∗ **flipstack**
- arraypool ∗ **cavetetlist**
- arraypool ∗ **cavebdrylist**
- arraypool ∗ **caveoldtetlist**
- arraypool ∗ **cavetetshlist**
- arraypool ∗ **cavetetseglist**
- arraypool ∗ **cavetetvertlist**
- arraypool ∗ **caveencshlist**
- arraypool ∗ **caveencseglist**
- arraypool ∗ **caveshlist**
- arraypool ∗ **caveshbdlist**
- arraypool ∗ **cavesegshlist**
- arraypool ∗ **subsegstack**
- arraypool ∗ **subfacstack**
- arraypool ∗ **subvertstack**
- arraypool ∗ **encseglist**
- arraypool ∗ **encshlist**
- int ∗ **idx2facetlist**
- point ∗ **facetverticeslist**
- point ∗ **segmentendpointslist**
- point **dummypoint**

- triface **recenttet**
- face **recentsh**
- point * **highordertable**
- int **numpointattrib**
- int **numelemattrib**
- int **sizeoftensor**
- int **pointmtrindex**
- int **pointparamindex**
- int **point2simindex**
- int **pointmarkindex**
- int **pointinsradiusindex**
- int **elemattribindex**
- int **volumeboundindex**
- int **elemmarkerindex**
- int **shmarkindex**
- int **areaboundindex**
- int **checksubsegflag**
- int **checksubfaceflag**
- int **checkconstraints**
- int **nonconvex**
- int **autofliplinklevel**
- int **useinsertradius**
- long **samples**
- unsigned long **randomseed**
- REAL **cosmaxdihed**
- REAL **cosmindihed**
- REAL **cossmtdihed**
- REAL **cosslidihed**
- REAL **minfaceang**
- REAL **minfacetdihed**
- REAL **tetprism_vol_sum**
- REAL **longest**
- REAL **minedgelength**
- REAL **xmax**
- REAL **xmin**
- REAL **ymax**
- REAL **ymin**
- REAL **zmax**
- REAL **zmin**
- long **insegments**
- long **hullsize**
- long **meshedges**
- long **meshhulledges**
- long **steinerleft**
- long **dupverts**
- long **unuverts**
- long **nonregularcount**
- long **st_segref_count**
- long **st_facref_count**
- long **st_volref_count**
- long **fillregioncount**
- long **cavitycount**
- long **cavityexpcount**
- long **flip14count**
- long **flip26count**

- long **flipn2ncount**
- long **flip23count**
- long **flip32count**
- long **flip44count**
- long **flip41count**
- long **flip31count**
- long **flip22count**
- unsigned long **totalworkmemory**
- int **transgc** [8][3][8]
- int **tsb1mod3** [8]

Static Public Attributes

- static REAL **PI** = 3.14159265358979323846264338327950288419716939937510582
- static int **bondtbl** [12][12] = {{0,},}
- static int **fsymtbl** [12][12] = {{0,},}
- static int **esymtbl** [12] = {9, 6, 11, 4, 3, 7, 1, 5, 10, 0, 8, 2}
- static int **enexttbl** [12] = {0,}
- static int **eprevtbl** [12] = {0,}
- static int **enextesymtbl** [12] = {0,}
- static int **eprevesymtbl** [12] = {0,}
- static int **eorgoppotbl** [12] = {0,}
- static int **edestoppotbl** [12] = {0,}
- static int **facepivot1** [12] = {0,}
- static int **facepivot2** [12][12] = {{0,},}
- static int **orgpivot** [12] = {7, 7, 5, 5, 6, 4, 4, 6, 5, 6, 7, 4}
- static int **destpivot** [12] = {6, 4, 4, 6, 5, 6, 7, 4, 7, 7, 5, 5}
- static int **apexpivot** [12] = {5, 6, 7, 4, 7, 7, 5, 5, 6, 4, 4, 6}
- static int **oppopivot** [12] = {4, 5, 6, 7, 4, 5, 6, 7, 4, 5, 6, 7}
- static int **tsbondtbl** [12][6] = {{0,},}
- static int **stbondtbl** [12][6] = {{0,},}
- static int **tspivottbl** [12][6] = {{0,},}
- static int **stpivottbl** [12][6] = {{0,},}
- static int **ver2edge** [12] = {0, 1, 2, 3, 3, 5, 1, 5, 4, 0, 4, 2}
- static int **edge2ver** [6] = {0, 1, 2, 3, 8, 5}
- static int **epivot** [12] = {4, 5, 2, 11, 4, 5, 2, 11, 4, 5, 2, 11}
- static int **sorgpivot** [6] = {3, 4, 4, 5, 5, 3}
- static int **sdestpivot** [6] = {4, 3, 5, 4, 3, 5}
- static int **sapexpivot** [6] = {5, 5, 3, 3, 4, 4}
- static int **snextpivot** [6] = {2, 5, 4, 1, 0, 3}

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

- modules/tetgen/tetgen.h
- modules/tetgen/tetgen.cpp
- modules/tetgen/tetgen.cxx

3.58 tri2mesh Class Reference

Public Attributes

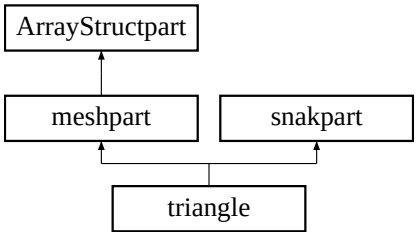
- vector< int > **celltarg**
- vector< double > **constrinfluence**

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

- incl/triangulate.hpp

3.59 triangle Class Reference

Inheritance diagram for triangle:



Public Member Functions

- void **disp** () const override
- void **disptree** (const mesh &meshin, int n) const override
- int **Key** () const override
- int **KeyParent** () const override
- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu) override
- void **PrepareForUse** () override
- bool **isready** (bool isInMesh) const override
- void **SwitchIndex** (int typeInd, int oldInd, int newInd)
- void **read** (FILE *fid) override
- void **write** (FILE *fid) const override
- void **TightenConnectivity** () override
- void **SetPointType** (int a, int b, int c)

Public Attributes

- vector< int > **pointtype**
- vector< int > **pointind**
- int **parentsurf** =0
- int **parenttype** =0
- tri2mesh **connec**

Private Attributes

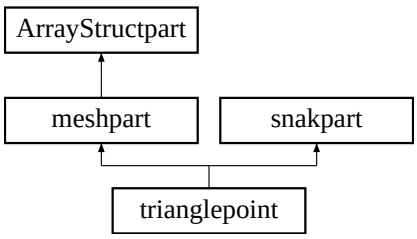
- bool **isTriangleReady** =false

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

- incl/triangulate.hpp
- src/snake/triangulate.cpp

3.60 trianglepoint Class Reference

Inheritance diagram for trianglepoint:



Public Member Functions

- void **disp** () const override
- void **disptree** (const mesh &meshin, int n) const override
- int **Key** () const override
- int **KeyParent** () const override
- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu) override
- void **ChangeIndicesSnakeMesh** (int nVert, int nEdge, int nSurf, int nVolu)
- void **PrepareForUse** () override
- bool **isready** (bool isInMesh) const override
- void **SwitchIndex** (int typeInd, int oldInd, int newInd)
- void **read** (FILE ∗fid) override
- void **write** (FILE ∗fid) const override
- void **TightenConnectivity** () override

Public Attributes

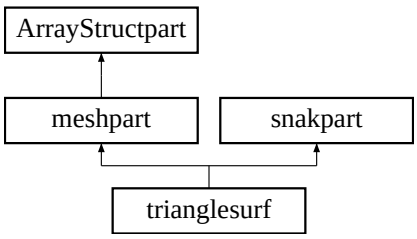
- coordvec **coord**
- int **parentsurf** =0
- int **parentType** =0
- int **nInfluences** =0

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

- incl/triangulate.hpp
- src/snake/triangulate.cpp

3.61 trianglesurf Class Reference

Inheritance diagram for trianglesurf:



Public Member Functions

- void **disp** () const override
- void **disptree** (const mesh &meshin, int n) const override
- int **Key** () const override
- int **KeyParent** () const override
- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu) override
- void **ChangeIndicesSnakeMesh** (int nVert, int nEdge, int nSurf, int nVolu)
- void **PrepareForUse** () override
- bool **isready** (bool isInMesh) const override
- void **SwitchIndex** (int typeInd, int oldInd, int newInd)
- void **read** (FILE ∗fid) override
- void **write** (FILE ∗fid) const override
- void **TightenConnectivity** () override

Public Attributes

- vector< int > **indvert**
- vector< int > **typevert**
- vector< int > **voluind**
- int **parentsurfmesh** =0

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

- incl/triangulate.hpp
- src/snake/triangulate.cpp

3.62 triangulation Class Reference

Public Member Functions

- void **disp** () const
- void **PrepareForUse** ()
- void **CleanDynaTri** ()
- void **CalcTriVertPosDyna** (int ii)
- void **CalcTriVertPosDyna** ()
- void **CalcTriVertPos** (int ii)
- void **CalcTriVertPos** ()
- void **SetActiveStaticTri** ()
- void **SetConnectivity** ()
- void **SetConnectivityStat** (int ii)
- void **SetConnectivityInter** (int ii)
- void **SetConnectivityDyna** (int ii)
- **triangulation** (mesh &meshin)

Public Attributes

- vector< int > **acttri**
- triarray **stattri**
- triarray **dynatri**
- triarray **intertri**
- tripointarray **trivert**
- trisurfarray **trisurf**
- snake * **snakeDep** =NULL
- mesh * **meshDep** =NULL

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

- incl/triangulate.hpp
- src/snake/triangulate.cpp

3.63 tetgenmesh::triface Class Reference

Public Member Functions

- triface & **operator=** (const triface &t)

Public Attributes

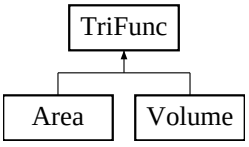
- tetrahedron * **tet**
- int **ver**

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

- modules/tetgen/tetgen.h

3.64 TriFunc Class Reference

Inheritance diagram for TriFunc:



Public Member Functions

- bool **CheckValid** ()
- bool **MakeValid** ()
- void **PreCalc** ()
- void **assign** (const vector< double > &in0, const vector< double > &in1, const vector< double > &in2)
- void **assign** (const vector< double > *in0, const vector< double > *in1, const vector< double > *in2)
- void **assign** (int pRepI, const vector< double > &pRep)
- void **ReturnDatPoint** (double **a, ArrayVec< double > **b, ArrayVec< double > **c)
- virtual void **Calc** ()=0
- **TriFunc** (int a)

Protected Member Functions

- bool **MakeValidField** (vector< double > *TriFunc::*mp)

Protected Attributes

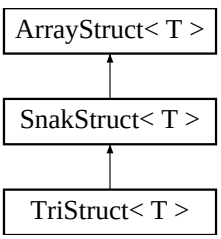
- vector< double > const * **p0** =NULL
- vector< double > const * **p1** =NULL
- vector< double > const * **p2** =NULL
- double **fun**
- ArrayVec< double > **jac**
- ArrayVec< double > **hes**
- bool **isReady**
- bool **isCalc**
- int **nTarg**

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

- incl/RSVSmath.hpp
- src/rsvs/RSVSmath.cpp

3.65 TriStruct< T > Class Template Reference

Inheritance diagram for TriStruct< T >:



Friends

- class **triangulation**

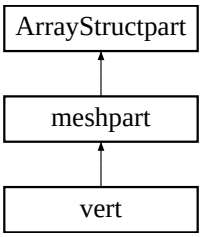
Additional Inherited Members

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

- incl/triangulate.hpp

3.66 vert Class Reference

Inheritance diagram for vert:



Public Member Functions

- void **disp** () const
- void **disptree** (const mesh &meshin, int n) const
- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu)
- void **PrepareForUse** ()
- bool **isready** (bool isInMesh) const
- void **read** (FILE ∗fid)
- void **write** (FILE ∗fid) const
- void **TightenConnectivity** ()
- **vert** (const vert &oldEdge)
- void **operator=** (const vert ∗other)
- int **Key** () const

Public Attributes

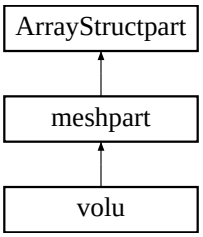
- vector< int > **edgeind**
- vector< double > **coord**

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

- incl/mesh.hpp
- src/grid/mesh.cpp

3.67 volu Class Reference

Inheritance diagram for volu:



Public Member Functions

- void **ChangeIndices** (int nVert, int nEdge, int nSurf, int nVolu)
- void **disp** () const
- void **disptree** (const mesh &meshin, int n) const
- void **PrepareForUse** ()
- bool **isready** (bool isInMesh) const
- void **read** (FILE ∗fid)
- void **write** (FILE ∗fid) const
- void **TightenConnectivity** ()
- **volu** (const volu &oldVolu)
- void **operator=** (const volu ∗other)
- int **Key** () const

Public Attributes

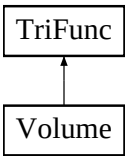
- double **fill**
- double **target**
- double **error**
- double **volume**
- vector< int > **surfind**

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

- incl/mesh.hpp
- src/grid/mesh.cpp

3.68 Volume Class Reference

Inheritance diagram for Volume:



Public Member Functions

- void **Calc** () override
- void **CalcFD** ()

Private Member Functions

- **TriFunc** ()
- **TriFunc** (int a)
- void **PreCalc** ()

Private Attributes

- vector< double > const * **p0**
- vector< double > const * **p1**
- vector< double > const * **p2**
- double **fun**
- ArrayVec< double > **jac**
- ArrayVec< double > **hes**

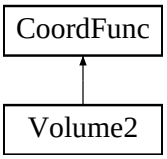
Additional Inherited Members

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

- incl/RSVSmath.hpp
- src/rsvs/RSVSmath.cpp

3.69 Volume2 Class Reference

Inheritance diagram for Volume2:



Public Member Functions

- void **Calc** () override

Private Member Functions

- void **PreCalc** ()

Private Attributes

- vector< vector< double > const * > **coords**
- double **fun**
- ArrayVec< double > **jac**
- ArrayVec< double > **hes**

Additional Inherited Members

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

- incl/RSVSmath.hpp
- src/rsvs/RSVSmath.cpp

3.70 tetgenio::voroedge Struct Reference

Public Attributes

- int **v1**
- int **v2**
- REAL **vnormal** [3]

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

- modules/tetgen/tetgen.h

3.71 tetgenio::vorofacet Struct Reference

Public Attributes

- int **c1**
- int **c2**
- int * **elist**

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

- modules/tetgen/tetgen.h

3.72 param::voronoi Class Reference

Public Member Functions

- void **PrepareForUse** ()
- void **ReadPoints** ()

Public Attributes

- std::vector< double > **inputpoints**
- double **distancebox**
- std::string **pointfile**
- double **snakecoarseness**

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

- incl/parameters.hpp
- src/parameters.cpp

3.73 param::voxel Class Reference

Parameters controlling grid properties.

`#include <parameters.hpp>`

Public Member Functions

- void **PrepareForUse** ()

Public Attributes

- std::array< int, 3 > **gridsizebackground**
- std::array< int, 3 > **gridsizesnake**

3.73.1 Detailed Description

Parameters controlling grid properties.

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

- incl/parameters.hpp
- src/parameters.cpp

Index