COMPUTING
OBJECTS

CM2 SurfMesh® T3/Q4 for OpenCascade®

Version 5.6

tutorials

Revision February 2025.

https://www.computing-objects.com

Forewords

This manual describes the 3-D parametric surface mesh generators for OpenCascade® Technology (OCCT) **CM2 SurfMesh® T3** and **CM2 SurfMesh® Q4**.

CM2 SurfMesh T3 and **CM2 SurfMesh Q4** generate surface meshes directly onto the CAD surfaces defined in a STEP/IGES/BRep file or by a TopoDS_Shape object (**OpenCascade®** object).

Main features of these parametric surface mesh generators:

- User mesh sizes on vertices, curves and surfaces (or a global default).
- Automatically adjusted mesh sizes according to the curvatures, respecting a given maximal chordal error.
- Isotropic meshes (triangles tend to be equilateral, quadrangles tend to be squares) or anisotropic meshes (triangles and quadrangles can be shrunken in principal curvature directions).
- Support of periodic surfaces (surfaces with non-bijective mapping like some torus, cones, spheres, cylinders...)
- Automatic repairing of some pathologies (merging of vertices and merging of curves if within a given tolerance from each other).
- Quadratic elements (T6, Q8 and Q9) can be generated, with possibly all of their nodes exactly onto the CAD curves/surfaces (curved elements).

Like many other meshers of the library, **CM2 SurfMesh T3/Q4** are multi-threaded (you can select in the settings the maximum number of threads the generator can use).
The generated meshes are reproducible (same mesh with same input data and same mesh with any number of threads).

CM2 SurfMesh T3 requires **CM2 TriaMesh Aniso** and the **OpenCascade®** libs.
CM2 SurfMesh Q4 requires **CM2 TriaMesh Aniso** + **CM2 QuadMesh Aniso** and the **OpenCascade®** libs.
Experienced users with a source code license of **CM2 SurfMesh T3/Q4** can port to a different CAD kernel (**Spatial®**, **Rhino®**, **Hoops®**...)

Data are exchanged through vector and matrix objects. Beginners with the **CM2 MeshTools®** SDK should start by reading the manual **CM2 Math1 - overview** to get first views on these mathematical containers.

For a complete description of the data and settings classes used with these meshers please refer to the **CM2 SurfMesh T3/Q4 - reference manual**.

CM2 SurfMesh T3/Q4 shouldn't be confused with **CM2 SurfRemesh T3/Q4®** which address the problem of *regenerating* a mesh on a 3-D surface defined by an already existing mesh (such as a tessellation).

CM2 SurfMesh T3/Q4 can be complemented with the solid mesh generators **CM2 TetraMesh® Iso**, **CM2 TetraMesh® Aniso** or **CM2 HexaMesh® Iso**.

The source code of the **CM2 MeshTools®** SDK (full library) has been registered with the APP under Inter Deposit number IDDN.FR.001.260002.00.R.P.1998.000.20700 (22/06/1998) and IDDN.FR.001.250030.00.S.P.1999.000.20700 (16/06/1999) is regularly deposited since then.

The source code specific to **CM2 SurfMesh® T3** and **CM2 SurfMesh® Q4** together with their manual have been registered with the APP under Inter Deposit number FR.001.480009.000.S.P.2019.000.20700 (20/11/2019) and is regularly deposited since then.

Table of contents

Forewords 2

1. Getting started. Meshing with CM2 SurfMesh T3 and default settings 5

Some declarations 6

Authorization of the library 6

Settings 6

Running the mesh generator on the CAD model 7

Output information 7

2. Chordal control 9

chordal_control_type 9

max_chordal_error 12

3. Lower bound on chordal control (min_h) 13

4. Meshing with quads (CM2 SurfMesh Q4) 15

5. Healing pathologies (fix_tolerance) 18

6. Specific mesh sizes 26

7. Quadratic elements 29

high_order_type 29

high_order_mode 29

Mesh gallery 31

This manual shows examples of meshing of some CAD models illustrating along the way some of the major options of the meshers.

Each example starts with including the file `stdafx.h` (usually a pre-compiled header) giving access to the required classes and the functions of the CM2 library (API).

The general namespace `cm2` has nested namespaces such as `cm2::vecscal`, `cm2::vecvec`, `cm2::meshtools` or `cm2::surfmesh_t3`. The user can add a `using namespace cm2` directive in this `stdafx.h` file. Keeping namespaces in the user's source code can however be useful to improve the legibility and to avoid name conflicts. In the rest of this document we assume such a `using namespace cm2` directive.

File `stdafx.h`:

```
// CM2 MESHTOOLS
#include "meshtools.h"      // MEDIT mesh visualisation routines (optional).
#include "meshtools2d.h"    // Some 2d mesh management routines (optional).
#include "surfmesh_t3.h"    // CM2 SurfMesh T3.
#include "surfmesh_q4.h"    // CM2 SurfMesh Q4.

using namespace cm2;        // Main cm2 namespace can now be omitted.
```

Required libraries[1]:

- `cm2math1`
- `cm2misc`
- `cm2meshtools` (optional)
- `cm2meshtools2d` (optional)
- `cm2surfmesh_t3`
- `cm2surfmesh_q4`

[1] The lib names end with `_($platform)_($ver)`. For instance `cm2surfmesh_t3_x64_56.dll`.
On Windows, file extensions for the libraries are `.lib` and `.dll`. On Linux/Unix/macOS platforms, file extensions are usually .a (static archive), .so or .dylib (dynamic lib).

1. Getting started. Meshing with CM2 SurfMesh T3 and default settings

The first example is a simple mechanical part (link rod).

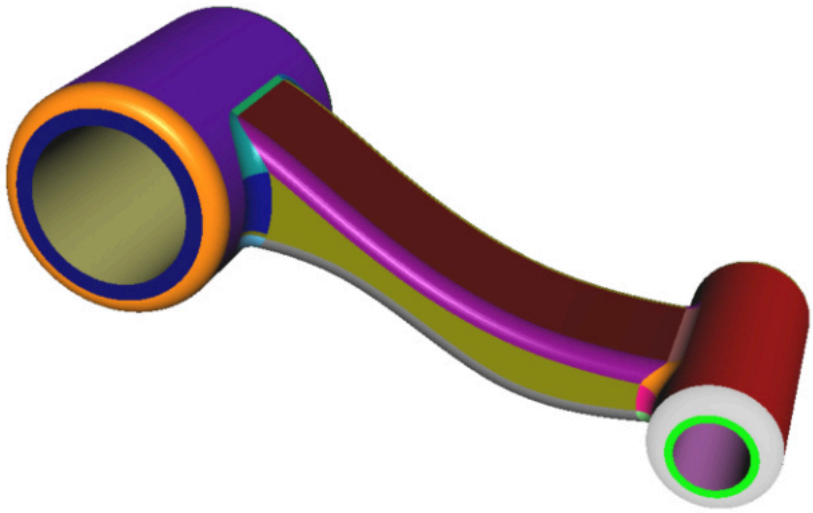


Figure 1 – Link rod.
STEP file (37 surfaces, 108 curves, 74 vertices).

.

The code to mesh this part with **CM2 SurfMesh T3** is given below (we are intersected in this manual only in the meshing of the surfaces, not the volumes):

```
#include "stdafx.h"
#include <iostream>

// Simple optional display handler.
static void display_hdl (void*, unsigned, const char* msg)   { std::cout << msg; }


int main()
{
   surfmesh_t3::mesher              the_mesher;
   surfmesh_t3::mesher::data_type   data;

   // UNLOCK THE DLL.
   surfmesh_t3::registration("Licensed to SMART Inc.", "F5BEA10ABCWX");

   // DEFINE SETTINGS AND RUN THE MESHER.
   the_mesher.settings.target_h = 0.2;
   the_mesher.run("link_rod.step", data);

   // SOME OUTPUT INFO (OPTIONAL).
   data.print_info(&display_hdl);

   // VISUALIZATION (OPTIONAL).
   meshtools::medit_output("out.mesh", data.pos, data.connectM, CM2_FACET3, data.colors);

   return 0;
}   // main
```

Figure 2 shows the generated mesh. The mesh is an all-triangle mesh (**CM2 SurfMesh T3** generates only triangles). See Section 4 below for all-quad and quad-dominant meshes.

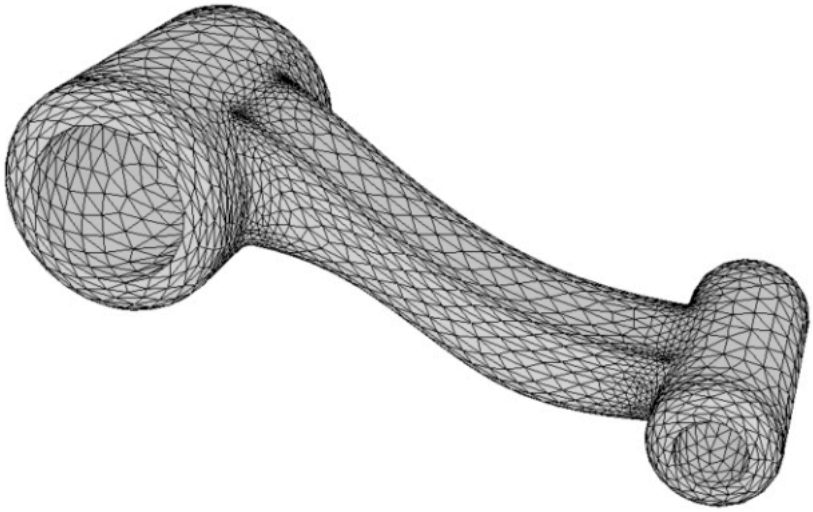


Figure 2 – Link rod. 5 706 triangles.
`target_h` = 0.2 (other settings to default).

Let us explain this program line by line.

Some declarations

`the_mesher` is an instance of `cm2::surfmesh_t3::mesher`.
`data` (type `cm2::surfmesh_t3::mesher::data_type`) is a key structure for the mesher. Upon exit, it stores among others the matrix `pos` of the nodes coordinates and the matrix `connectM` of the connectivity of the elements.
`data.pos` is a `DoubleMat` (variable-sized matrix of doubles)[2]. `data.connectM` is a `UIntMat` (variable-sized matrix of natural integers). `connectM(i, j)` is the i^{th} local node of the j^{th} element. This integer value is the column number in matrix `pos` where the coordinates of this node can be found[3].

Authorization of the library

The library needs to be unlocked through a call to `cm2::surfmesh_t3::registration`. Two strings must be provided for each library: the name of your company or organization that has acquired the license and a secret code[4]. Note that both strings are case sensitive and the registration call must be made each time the library is loaded into memory and before the first run of the mesher.

```
surfmesh_t3::registration("Licensed to SMART Inc.", "F5BEA10ABCWX");
```

Settings

The settings here are very simple and consist only in a single global mesh size (same unit as the model):

```
the_mesher.settings.target_h = 0.2;
```

[2] See manual **CM2 Math1 - overview**.

[3] Recall that array indices are zero based (from 0 to N-1).

[4] Contact `license@computing-objects.com` for any licensing inquiry.

Running the mesh generator on the CAD model

Here, a file name is provided to the `cm2::surfmesh_t3::mesher::run` member together with a data structure (`cm2::surfmesh_t3::mesher::data_type`) that will hold the resulting mesh (and some other auxiliary data):

```
the_mesher.run("link_rod.step", data);
```

The file can be a STEP file (with extension ".STEP", ".STP", ".step" or ".stp"), a IGES file (with extension ".IGES", ".IGS", ".iges" or ".igs") or a B-Rep file (with extension ".BREP" or ".brep").

The mesher accepts also raw OCCT objects (as `TopoDs_Shape*` cast to a `void*` pointer):

```
STEPControl_Reader   reader;
TopoDS_Shape         root;

if (reader.ReadFile("link_rod.step") == IFSelect_RetDone)
{
   reader.TransferRoots();
   root = reader.OneShape();
}
the_mesher.run(static_cast<void*>(&root), data);
```

Output information

Printed information about the generated mesh and a MEDIT[5] output file are obtained with:

```
data.print_info(&display_hdl);
meshtools::medit_output("out.mesh", data.pos, data.connectM, CM2_FACET3, data.colors);
```

Figure 3 shows the output of `data.print_info(&display_hdl)`:

The times spent in the several steps of the meshing process are given in seconds[6]. The first step is the meshing of the curves. The second step is the meshing of the surfaces (multi-threaded).
The read time dominates here the total meshing time. It can be eliminated (or moved) if we use the direct run on a `TopoDS_Shape*`.

[5] MEDIT is a free visualization program. Other available output formats: Ensight, FEMAP, Nastran, STL (ASCII or binary), VTK, Wavefront OBJ.

[6] All runs were done with x64 CM2 libs (VS 2010 MD build) on Windows® 8.1 with Intel® Xeon® E3-1270 V2 3.5 Ghz (8 threads, turbo boost disabled).

```
************************************************
*          CM2 SurfMesh T3(R) (5.6.0.0)          *
************************************************
Nodes           : 2851
Triangles       : 5706
Vertices        : 74 (merged: 0)
Curves          : 108 (merged: 0)
Surfaces        : 37
Area            : 3.206425E+001
Qmin            : 2.224215E-001
Read time       : 0.81 s.
Curvs mesh time : 0.04 s.
Surfs mesh time : 0.10 s.
Total mesh time : 0.14 s. (39351.73 t/s.)

************ HISTOGRAM QS *************
Total number of bins     :           11
Total number of counts   :         5706
Number of larger values  :            0
Number of smaller values :            0
V max                    : 9.979977E-001
V mean                   : 6.692706E-001
V min                    : 2.224215E-001

Bin number       -- Bin boundaries --          Hits

   10            0.90            1.00           855
    9            0.80            0.90           907
    8            0.70            0.80           840
    7            0.60            0.70           823
    6            0.50            0.60           876
    5            0.40            0.50           938
    4            0.30            0.40           435
    3            0.20            0.30            32
    2            0.10            0.20             0
    1            0.01            0.10             0
    0            0.00            0.01             0
```

Figure 3 – Link rod. Output info (triangles).
`target_h` = 0.2 (other settings to default).

You can notice that despite a global unique mesh size (`target_h` = 0.2), the elements do not have all the same size. Moreover, some are shrunken in some directions. This is because of the default settings (illustrated in subsequent sections) regarding the *chordal error control*. It defaults to the *anisotropic* chordal control (allowing shrinking along high curvature principal directions) and to a maximum chordal error of 2% of the local curvature radii.

2. Chordal control

The maximal chordal error (i.e. maximal distance between the flat elements and the exact CAD curves/surfaces) is controlled by two parameters: `settings_type::chordal_control_type` and `settings_type::max_chordal_error`.

chordal_control_type

Chordal control is disabled when `chordal_control_type` = 0. The elements have uniform size (as much as possible) specified by `settings_type::target_h`. The number of elements is minimal but the more curved the surfaces (and the larger `target_h`), the larger the chordal error.

```
#include "stdafx.h"
#include <iostream>

// Simple optional display handler.
static void display_hdl (void*, unsigned, const char* msg)   { std::cout << msg; }

int main()
{
   surfmesh_t3::mesher              the_mesher;
   surfmesh_t3::mesher::data_type   data;

   // UNLOCK THE DLL.
   surfmesh_t3::registration("Licensed to SMART Inc.", "F5BEA10ABCWX");

   // DEFINE SETTINGS AND RUN THE MESHER.
   the_mesher.settings.target_h = 0.2;
   the_mesher.settings.chordal_control_type = 0;    // Disable chordal error control.
   the_mesher.run("link_rod.step", data);

   // SOME OUTPUT INFO (OPTIONAL).
   data.print_info(&display_hdl);

   // VISUALIZATION (OPTIONAL).
   meshtools::medit_output("out.mesh", data.pos, data.connectM, CM2_FACET3);

   return 0;
}   // main
```

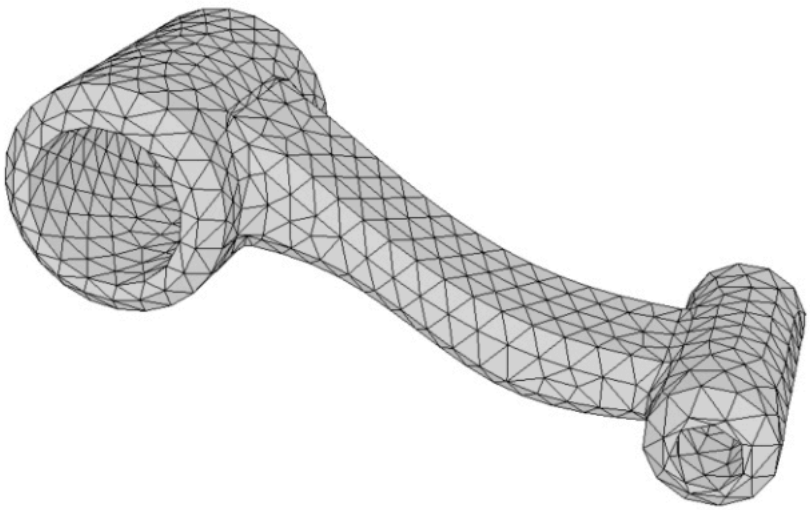


Figure 4 – Link rod. 1 934 triangles
`target_h` = 0.2, chordal control disabled (`chordal_control_type` = 0).

☞ Notice that, due to the dimensions of the surfaces, some elements may still have size smaller than `target_h`

(elements are generated surface by surface and cannot span over or be optimized across the boundary curves of the surfaces).

```
************************************************
*          CM2 SurfMesh T3(R) (5.6.0.0)          *
************************************************
Nodes            : 965
Triangles        : 1934
Vertices         : 74 (merged: 0)
Curves           : 108 (merged: 0)
Surfaces         : 37
Area             : 3.146739E+001
Qmin             : 1.988903E-001
Read time        : 0.81 s.
Curvs mesh time : 0.02 s.
Surfs mesh time : 0.08 s.
Total mesh time : 0.10 s. (18596.14 t/s.)

************ HISTOGRAM QS *************
Total number of bins     :           11
Total number of counts   :         1934
Number of larger values  :            0
Number of smaller values :            0
V max                    : 9.995265E-001
V mean                   : 8.882759E-001
V min                    : 1.988903E-001

Bin number       -- Bin boundaries --          Hits

   10            0.90            1.00          1187
    9            0.80            0.90           440
    8            0.70            0.80           155
    7            0.60            0.70            64
    6            0.50            0.60            19
    5            0.40            0.50            29
    4            0.30            0.40            19
    3            0.20            0.30            20
    2            0.10            0.20             1
    1            0.01            0.10             0
    0            0.00            0.01             0
```

Figure 5 – Link rod. Output info (triangles).
`target_h` = 0.2, chordal control disabled (`chordal_control_type` = 0).

Chordal control is *isotropic* when `chordal_control_type` = 3 or 1 (see Reference Manual). The elements have size given by `settings_type::target_h` or size computed by the chordal formula (below) whatever the lowest. The number of elements can be much larger than with no chordal control but the maximum chordal error is kept anywhere below `settings_type::max_chordal_error` (2% of local smallest radius by default).

$$h = 2\,R\,\sqrt{\varepsilon(2-\varepsilon)} \;\approx 2\sqrt{2}\;R\,\sqrt{\varepsilon}$$

Formula 1 – Mesh size h for a given curvature radius R and a chordal error ε.

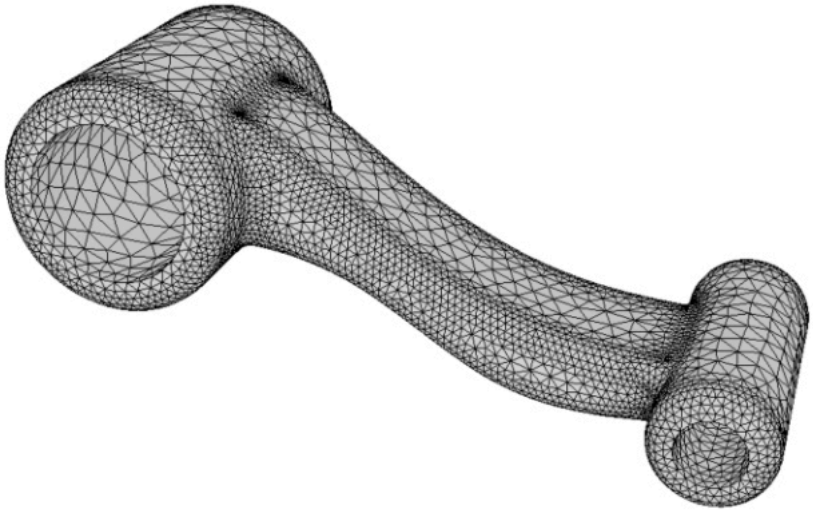


Figure 6 – Link rod. 11 272 triangles.
`target_h` = 0.2, isotropic chordal control (`chordal_control_type` = 3)

```
************************************************
*          CM2 SurfMesh T3(R) (5.6.0.0)        *
************************************************
Nodes           : 5634
Triangles       : 11272
Vertices        : 74 (merged: 0)
Curves          : 108 (merged: 0)
Surfaces        : 37
Area            : 3.206579E+001
Qmin            : 4.266428E-001
Read time       : 0.81 s.
Curvs mesh time : 0.03 s.
Surfs mesh time : 0.19 s.
Total mesh time : 0.22 s. (50774.75 t/s.)

************ HISTOGRAM QS *************
Total number of bins     :          11
Total number of counts   :       11272
Number of larger values  :           0
Number of smaller values :           0
V max                    : 9.996927E-001
V mean                   : 8.847562E-001
V min                    : 4.266428E-001

Bin number       -- Bin boundaries --          Hits

   10            0.90            1.00          5838
    9            0.80            0.90          3652
    8            0.70            0.80          1586
    7            0.60            0.70           184
    6            0.50            0.60            10
    5            0.40            0.50             2
    4            0.30            0.40             0
    3            0.20            0.30             0
    2            0.10            0.20             0
    1            0.01            0.10             0
    0            0.00            0.01             0
```

Figure 7 – Link rod. Output info (triangles).
`target_h` = 0.2, isotropic chordal control (`chordal_control_type` = 3).

Chordal control is *anisotropic* when `chordal_control_type` = 4 (the default), or 2 (see reference manual). The elements have size given by `settings_type::target_h` or anisotropic sizes computed by the chordal formula (Formula 1) in each principal curvature direction, whatever the lowest.
Since the sizes are reduced only in the directions of curvature, the number of elements can be much lower than with isotropic chordal control (but still higher than with no chordal control at all) with a similar maximum chordal error (5 706 triangles, Figure 2).

max_chordal_error

The maximum chordal error δ is governed by `settings_type::max_chordal_error`. The mesh size is reduced locally to limit the chordal error between the mesh and the curves or the surfaces:

- If negative, this value (- ε) is relative to the local radii *R* (max chordal error δ = - ε *R*).
- If positive, this value is absolute ($\delta = \varepsilon$).

Formula 1 shows that the mesh size depends linearly on the local curvature radius *R* and as the square root of the relative chordal error ε. This gives for instance:

- with `max_chordal_error` = -0.05 (ε = 5%), a complete circle is meshed with approximately 10 elements.
- with `max_chordal_error` = -0.03 (ε = 3%), a complete circle is meshed with approximately 13 elements.
- with `max_chordal_error` = -0.02 (ε = 2%), a complete circle is meshed with approximately 16 elements.
- with `max_chordal_error` = -0.01 (ε = 1%), a complete circle is meshed with approximately 22 elements.

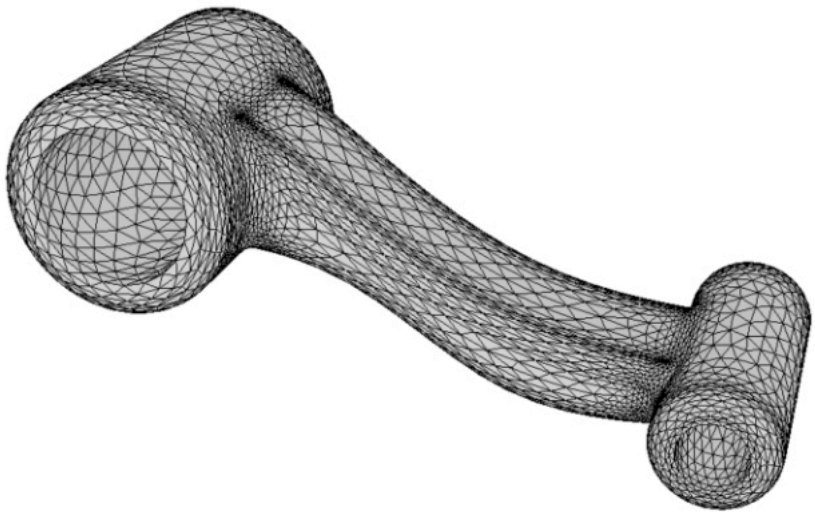


Figure 8 – Link rod. 8 402 triangles.
`target_h` = 0.2, `max_chordal_error` = -0.01 (1% of radius),
default anisotropic chordal control.

The triangles are more shrunken along the curved regions but remain mostly governed by the `target_h` mesh size (unchanged at 0.2) on flat/shallow surfaces.

3. Lower bound on chordal control (min_h)

The security lower bound for the size of the elements.

It is well known that CAD models are prone to local geometric errors almost invisible for the human eye. These errors can be small gaps between surfaces, small overlaps or tiny highly-curved fillers connecting surfaces or curves. This can translate in very high localized curvatures and, through Formula 1, to very small elements.
Even in cases of very good CAD models, users may want to avoid element smaller than a given value (relaxing somewhat the chordal error).
This is done with the `settings_type::min_h` parameter.

The mesh sizes will not be reduced (by `max_chordal_error`) beyond this limit h_{min}:

- If negative, this value is relative to `target_h` (h_{min} = - `min_h` * `target_h`).
- If positive, this value is absolute (h_{min} = `min_h`).

On the link rod model above with 1% anisotropic chordal control we reduce the number of elements from 8402 to 8110 if we set min_h = -0.1 (instead of the default -0.001).

```
#include "stdafx.h"
#include <iostream>

// Simple optional display handler.
static void display_hdl (void*, unsigned, const char* msg)   { std::cout << msg; }


int main()
{
   surfmesh_t3::mesher              the_mesher;
   surfmesh_t3::mesher::data_type   data;

   // UNLOCK THE DLL.
   surfmesh_t3::registration("Licensed to SMART Inc.", "F5BEA10ABCWX");

   // DEFINE SETTINGS AND RUN THE MESHER.
   the_mesher.settings.target_h = 0.2;
   the_mesher.settings.max_chordal_error = -0.01;
   the_mesher.settings.min_h = -0.1;
   the_mesher.run("link_rod.step", data);

   // SOME OUTPUT INFO (OPTIONAL).
   data.print_info(&display_hdl);

   // VISUALIZATION (OPTIONAL).
   meshtools::medit_output("out.mesh", data.pos, data.connectM, CM2_FACET3);

   return 0;
}   // main
```

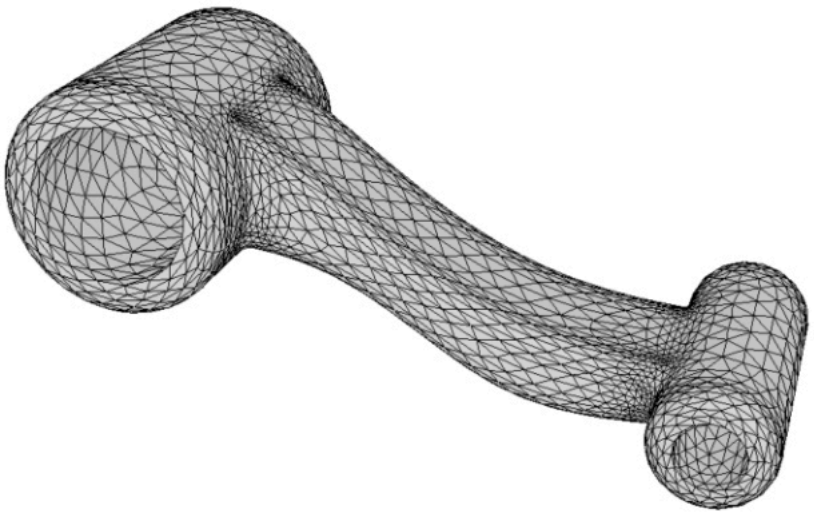


Figure 9 – Link rod. 8 110 triangles
`target_h` = 0.2, `max_chordal_error` = -0.01 (1% of radius), `min_h` = -0.1,
default anisotropic chordal control.

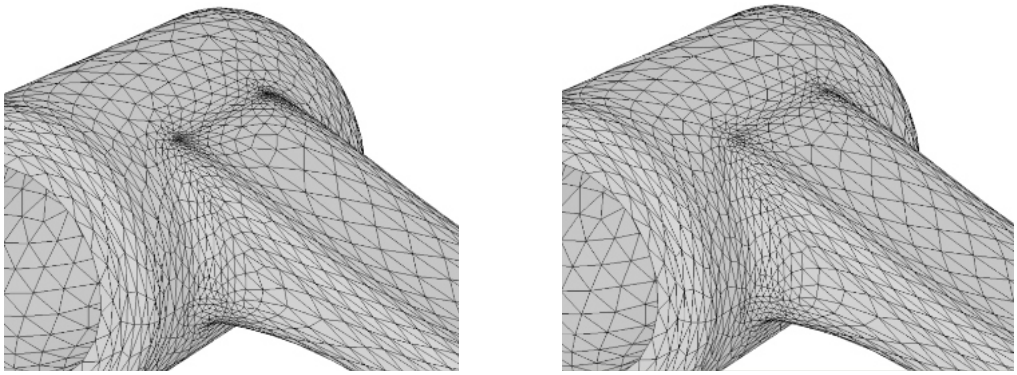


Figure 10 – Link rod. Detail.
`min_h` = -0.001 (default, left), `min_h` = -0.1 (right).

4. Meshing with quads (CM2 SurfMesh Q4)

To mesh with quads, it's just as easy as to replace the namespace `surfmesh_t3` with `surfmesh_q4`. All the settings seen so far (`target_h`, `chordal_control_type`, `max_chordal_error`, `min_h`) are also available in **CM2 SurfMesh Q4**.

```
#include "stdafx.h"
#include <iostream>

// Simple optional display handler.
static void display_hdl (void*, unsigned, const char* msg)   { std::cout << msg; }


int main()
{
   surfmesh_q4::mesher              the_mesher;
   surfmesh_q4::mesher::data_type   data;

   // UNLOCK THE DLL.
   surfmesh_q4::registration("Licensed to SMART Inc.", "F5BEA10ABCWX");

   // DEFINE SETTINGS AND RUN THE MESHER.
   the_mesher.settings.target_h = 0.2;
   the_mesher.settings.max_chordal_error = -0.01;
   the_mesher.settings.min_h = -0.1;
   the_mesher.run("link_rod.step", data);

   // SOME OUTPUT INFO (OPTIONAL).
   data.print_info(&display_hdl);

   // VISUALIZATION (OPTIONAL).
   meshtools::medit_output("out.mesh", data.pos, data.connectM, CM2_FACE_MIX);

   return 0;
}   // main
```

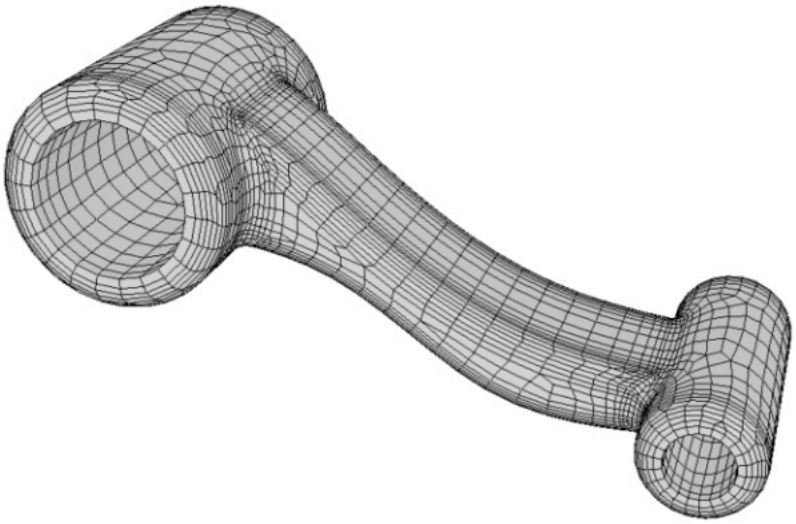


Figure 11 – Link rod. Quad-dominant mesh. 802 quads, 100 triangles. `target_h` = 0.2, `max_chordal_error` = -0.01 (1% of radius), `min_h` = -0.1, default anisotropic chordal control.

```
************************************************
*          CM2 SurfMesh Q4(R) (5.6.0.0)          *
************************************************
Nodes           : 3850
Elements        : 3902
  Quadrangles   : 3802 (97.44 %, 99.23 %)
  Triangles     : 100 (2.56 %, 0.77 %)
Vertices        : 74 (merged: 0)
Curves          : 108 (merged: 0)
Surfaces        : 37
Area            : 3.207294E+001
Qmin            : 1.749600E-001
Read time       : 0.81 s.
Curvs mesh time : 0.04 s.
Surfs mesh time : 0.15 s.
Total mesh time : 0.20 s. (19908.15 t/s.)

************ HISTOGRAM QS *************
Total number of bins     :           11
Total number of counts   :         3902
Number of larger values  :            0
Number of smaller values :            0
V max                    : 9.999616E-001
V mean                   : 6.362755E-001
V min                    : 1.749600E-001

Bin number       -- Bin boundaries --          Hits

   10            0.90            1.00           524
    9            0.80            0.90           517
    8            0.70            0.80           524
    7            0.60            0.70           641
    6            0.50            0.60           607
    5            0.40            0.50           457
    4            0.30            0.40           438
    3            0.20            0.30           186
    2            0.10            0.20             8
    1            0.01            0.10             0
    0            0.00            0.01             0
```

Figure 12 – Link rod. Output info (quad-dominant).
`target_h` = 0.2, `max_chordal_error` = -0.01 (1% of radius), `min_h` = -0.1,
default anisotropic chordal control.

By default, **CM2 SurfMesh Q4** generates *quad-dominant* meshes. To get an *all-quad* mesh, one needs to set the `settings_type::all_quad_flag` = true (default = false).

```
#include "stdafx.h"
#include <iostream>

// Simple optional display handler.
static void display_hdl (void*, unsigned, const char* msg)   { std::cout << msg; }


int main()
{
   surfmesh_q4::mesher               the_mesher;
   surfmesh_q4::mesher::data_type    data;

   // UNLOCK THE DLL.
   surfmesh_q4::registration("Licensed to SMART Inc.", "F5BEA10ABCWX");

   // DEFINE SETTINGS AND RUN THE MESHER.
   the_mesher.settings.target_h = 0.2;
   the_mesher.settings.max_chordal_error = -0.01;
   the_mesher.settings.all_quad_flag = true;
   the_mesher.settings.min_h = -0.1;
   the_mesher.run("link_rod.step", data);

   // SOME OUTPUT INFO (OPTIONAL).
   data.print_info(&display_hdl);

   // VISUALIZATION (OPTIONAL).
   meshtools::medit_output("out.mesh", data.pos, data.connectM, CM2_FACEQ4);

   return 0;
}   // main
```

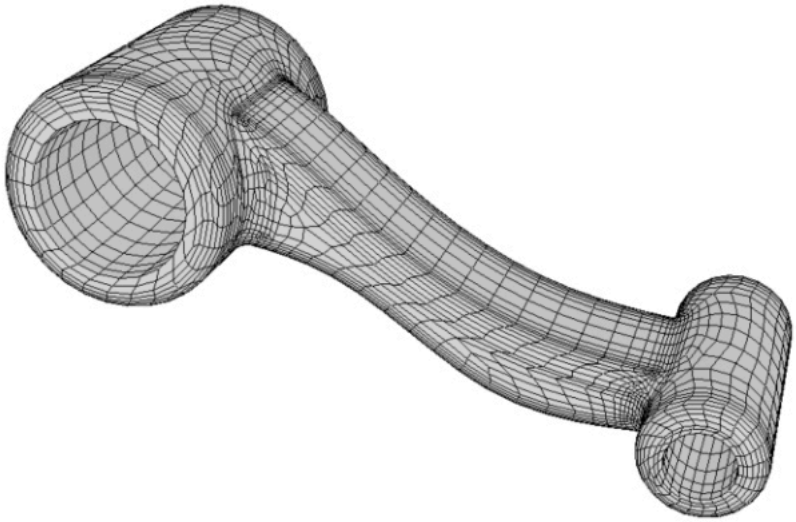


Figure 13 – Link rod. All-quad mesh. 4 092 quadrangles.
`target_h` = 0.2, `max_chordal_error` = -0.01 (1% of radius), `min_h` = -0.1,
default anisotropic chordal control.

```
************************************************
*          CM2 SurfMesh Q4(R) (5.6.0.0)        *
************************************************
Nodes           : 4090
Elements        : 4092
  Quadrangles   : 4092 (100.00 %, 100.00 %)
  Triangles     : 0 (0.00 %, 0.00 %)
Vertices        : 74 (merged: 0)
Curves          : 108 (merged: 0)
Surfaces        : 37
Area            : 3.208350E+001
Qmin            : 1.230279E-001
Read time       : 0.81 s.
Curvs mesh time : 0.04 s.
Surfs mesh time : 0.19 s.
Total mesh time : 0.23 s. (17562.23 t/s.)

************ HISTOGRAM QS *************
Total number of bins     :           11
Total number of counts   :         4092
Number of larger values  :            0
Number of smaller values :            0
V max                    : 9.995153E-001
V mean                   : 6.078816E-001
V min                    : 1.230279E-001

Bin number       -- Bin boundaries --          Hits

   10            0.90            1.00           525
    9            0.80            0.90           440
    8            0.70            0.80           438
    7            0.60            0.70           679
    6            0.50            0.60           594
    5            0.40            0.50           532
    4            0.30            0.40           620
    3            0.20            0.30           229
    2            0.10            0.20            35
    1            0.01            0.10             0
    0            0.00            0.01             0
```

Figure 14 – Link rod. Output info (all-quad).
`target_h` = 0.2, `max_chordal_error` = -0.01 (1% of radius), `min_h` = -0.1,
default anisotropic chordal control.

☞ If possible, prefer the quad-dominant mode over the all-quad mode. The former gives usually meshes of better quality. Moreover, all-quad mode can fail at present (version 5.6.0) on some non-bijective (aka. periodic) surfaces, returning a mesh with a few triangles and a `CM2_FAILED_SURFACES_WARNING`. This can usually be fixed a posteriori with a local remeshing with the help of **CM2 SurfRemesh Q4** (see **CM2 SurfRemesh T3/Q4 - tutorial** and **reference manual**).

5. Healing pathologies (fix_tolerance)

On top of (local) geometric pathologies (such as spurious high-curvatures which can be mitigated by the `min_h` parameter), CAD models often exhibit topological errors such as gaps between surfaces. This is usually caused by a boundary between two surfaces defined by two different curves (almost coincident) instead of sharing the same curve. **CM2 SurfMesh T3** and **CM2 SurfMesh Q4** can repair this kind of error of modeling.

The following example has many discontinuities between surfaces:

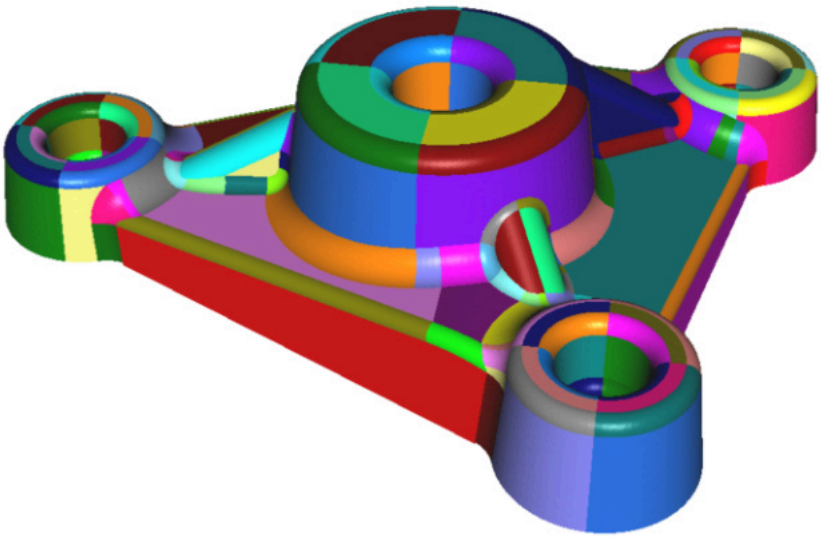


Figure 15 – Bearing.
IGES file. 925 vertices, 9 041 curves, 213 surfaces.

We first proceed with repairing disabled to show the initial CAD errors (with the help of some meshtools code to extract the boundaries of the mesh). Disabling the repairing merges is done by affecting the special value `DBL_MAX` to `fix_tolerance` (`fix_tolerance` = 0 would merge the vertices and the curves that has *exactly* the same positions).

The `fix_tolerance` parameter obeys the same rule as `max_chordal_error` and `min_h`:

- If negative, this value is relative to `target_h` (*merge tol* = - `fix_tolerance` * `target_h`).
- If positive, this value is absolute (*merge tol* = `fix_tolerance`).

Vertices are merged (vertices with highest ids into lowest ones) if their distance is below or equal to *merge tol*. Curves are merged (curves with highest ids into lowest ones) if their (approximated) maximal distance is below or equal to *merge tol*.

☞ **CM2 SurfMesh T3/Q4** cannot repair all pathologies:

- At present (version 5.6), curves are merged only when *any point* of one is within the tolerance distance of the second. This implies that two connected and successive small curves are not merged into a third larger one, even if the group of the two curves is coincident with the large curve.
- At present (version 5.6), surfaces cannot be merged (only their contour curves can be merged).

```
#include "stdafx.h"
#include <iostream>

// Simple optional display handler.
static void display_hdl (void*, unsigned, const char* msg)   { std::cout << msg; }


int main()
{
   surfmesh_t3::mesher              the_mesher;
   surfmesh_t3::mesher::data_type   data;
   UIntMat                          connectV, connectB;

   // UNLOCK THE DLL.
   surfmesh_t3::registration("Licensed to SMART Inc.", "F5BEA10ABCWX");

   // DEFINE SETTINGS AND RUN THE MESHER.
   the_mesher.settings.fix_tolerance = DBL_MAX;   // Disable merging.
   the_mesher.settings.target_h = 0.003;
   the_mesher.run("bearing.iges", data);

   // SOME OUTPUT INFO (OPTIONAL).
   data.print_info(&display_hdl);

   // EXTRACT THE BOUNDARIES (TO SHOW GAPS).
   meshtools::get_neighbors(data.connectM, CM2_FACET3, false, connectV);
   meshtools::get_mesh_boundaries(data.connectM, connectV, CM2_FACET3, connectB);

   // VISUALIZATION (OPTIONAL).
   meshtools::medit_output("out.mesh", data.pos, data.connectM, CM2_FACET3, data.colors);
   meshtools::medit_output("boundaries.mesh", data.pos, connectB, CM2_EDGE2);

   return 0;
}   // main
```

`target_h` is set to 0.003. The other parameters are left to their default (i.e. 2% anisotropic chordal control).

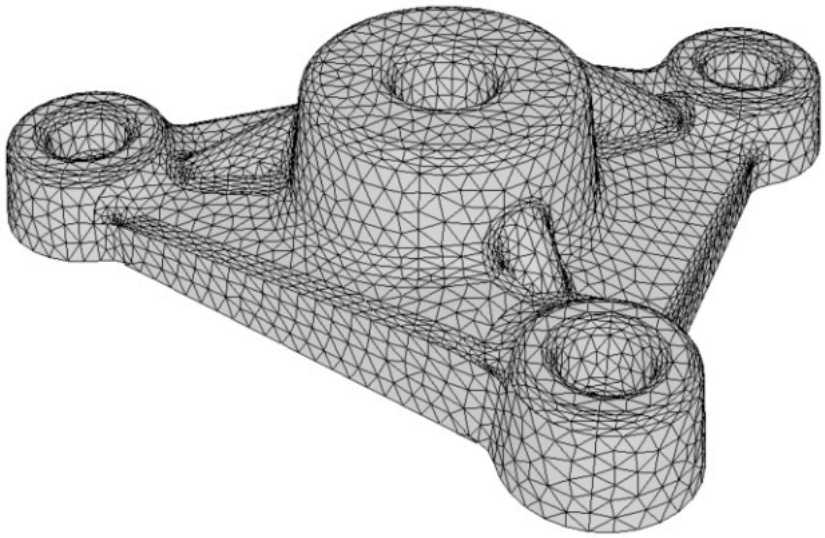


Figure 16 – Bearing. Mesh with no repairing.. 6 909 triangles.
`target_h` = 0.03, default 2% anisotropic chordal control.

```
************************************************
*          CM2 SurfMesh T3(R) (5.6.0.0)          *
************************************************
Nodes           : 5354
Triangles       : 6909
Vertices        : 925 (merged: 0)
Curves          : 941 (merged: 0)
Surfaces        : 213
Area            : 1.336568E-002
Qmin            : 1.125807E-001
Read time       : 0.26 s.
Curvs mesh time : 0.16 s.
Surfs mesh time : 0.23 s.
Total mesh time : 0.40 s. (17272.51 t/s.)

************ HISTOGRAM QS *************
Total number of bins     :           11
Total number of counts   :         6909
Number of larger values  :            0
Number of smaller values :            0
V max                    : 9.993214E-001
V mean                   : 6.647028E-001
V min                    : 1.125807E-001

Bin number       -- Bin boundaries --          Hits

   10            0.90            1.00           929
    9            0.80            0.90          1304
    8            0.70            0.80          1006
    7            0.60            0.70          1015
    6            0.50            0.60           884
    5            0.40            0.50           947
    4            0.30            0.40           668
    3            0.20            0.30           141
    2            0.10            0.20            15
    1            0.01            0.10             0
    0            0.00            0.01             0
```

Figure 17 – Bearing. Output info (triangles).
target_h = 0.003, default 2% anisotropic chordal control, no repairing.

The problems of connectivity between many surfaces appear clearly with a zoom:

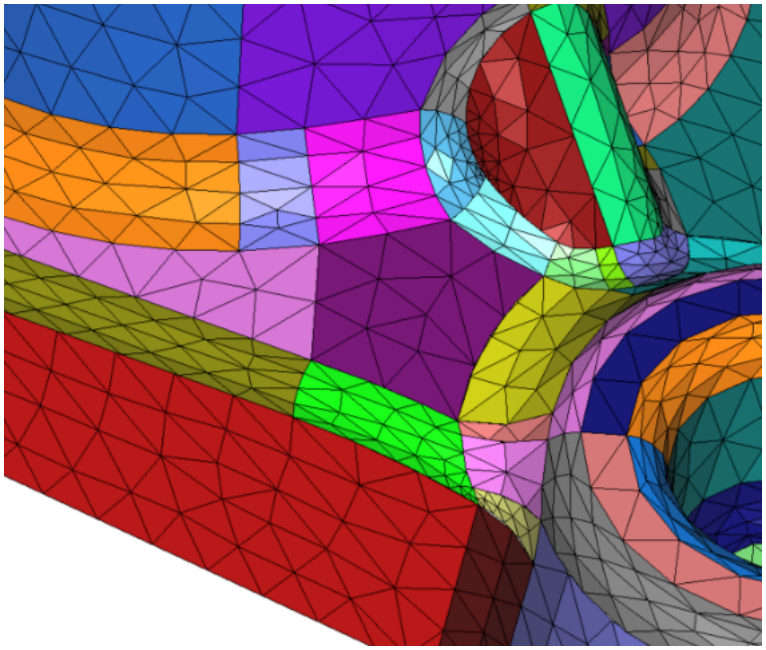


Figure 18 – Bearing. Zoom.
Meshes between many surfaces are not connected.

Figure 19 shows all the boundary edges (i.e. edges that are not shared by exactly two triangles).

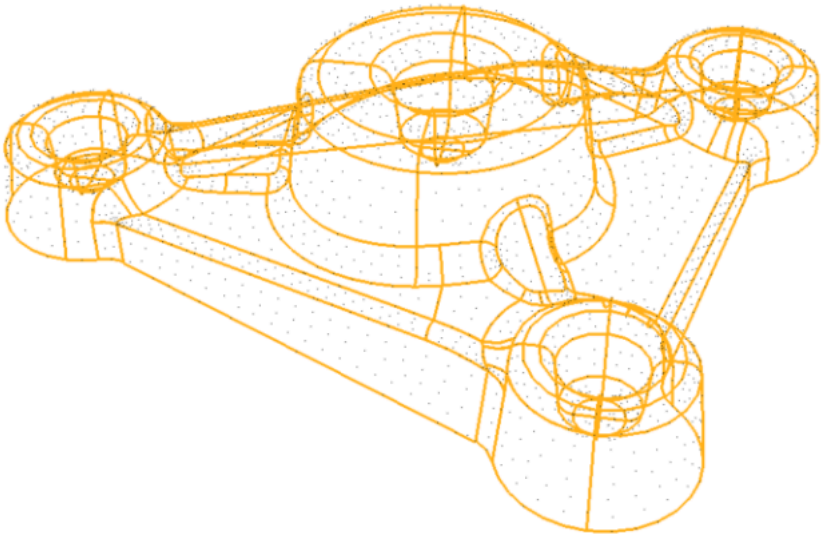


Figure 19 – Bearing.
Boundary edges (no repairing).

With the default value `fix_tolerance` = -1E-4 (giving here an absolute tolerance of 3E-7), almost all the gaps are repaired. But not all.

Figure 20 – Bearing.
Boundary edges with default `fix_tolerance` = - 1E-4.

With this model, with have to push `fix_tolerance` to +5E-6 (absolute value) to merge all bad curves:

Figure 21 – Bearing.
Boundary edges with `fix_tolerance` = + 5E-6.

```
************************************************
*          CM2 SurfMesh T3(R) (5.6.0.0)        *
************************************************
Nodes           : 5357
Triangles       : 9254
Vertices        : 925 (merged: 662)
Curves          : 941 (merged: 450)
Surfaces        : 213
Area            : 1.336575E-002
Qmin            : 8.545688E-002
Read time       : 0.26 s.
Curvs mesh time : 0.14 s.
Surfs mesh time : 0.26 s.
Total mesh time : 0.41 s. (22570.73 t/s.)

************ HISTOGRAM QS *************
Total number of bins     :           11
Total number of counts   :         9254
Number of larger values  :            0
Number of smaller values :            0
V max                    : 9.993164E-001
V mean                   : 6.677958E-001
V min                    : 8.545688E-002

Bin number       -- Bin boundaries --          Hits

   10            0.90            1.00          1096
    9            0.80            0.90          1519
    8            0.70            0.80          1626
    7            0.60            0.70          1600
    6            0.50            0.60          1404
    5            0.40            0.50          1231
    4            0.30            0.40           644
    3            0.20            0.30           119
    2            0.10            0.20            14
    1            0.01            0.10             1
    0            0.00            0.01             0
```

Figure 22 – Bearing. Output info (triangles).
`target_h` = 0.003, `fix_tolerance`= + 5E-6,
default 2% anisotropic chordal control.

662 vertices have been merged (out of 925). 450 curves have been merged (out of 941). The boundary edges are now limited to the external boundary of the model. We have now a manifold global surface:

We can improve the mesh a bit further by raising the `min_h` parameter to -1E-1 (from default -1E-3) as in Section 3:

```
#include "stdafx.h"
#include <iostream>

// Simple optional display handler.
static void display_hdl (void*, unsigned, const char* msg)   { std::cout << msg; }


int main()
{
   surfmesh_t3::mesher              the_mesher;
   surfmesh_t3::mesher::data_type   data;

   // UNLOCK THE DLL.
   surfmesh_t3::registration("Licensed to SMART Inc.", "F5BEA10ABCWX");

   // DEFINE SETTINGS AND RUN THE MESHER.
   the_mesher.settings.fix_tolerance = 5E-6;   // Absolute val.
   the_mesher.settings.min_h = -0.1;           // Relative val.
   the_mesher.settings.target_h = 0.003;
   the_mesher.run("bearing.iges", data);

   // SOME OUTPUT INFO (OPTIONAL).
   data.print_info(&display_hdl);

   // VISUALIZATION (OPTIONAL).
   meshtools::medit_output("out.mesh", data.pos, data.connectM, CM2_FACET3);

   return 0;
}   // main
```

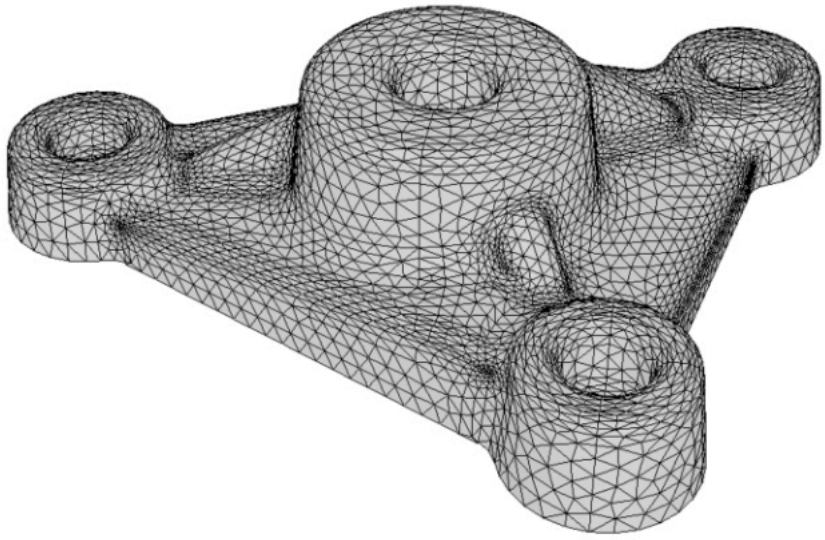


Figure 23 – Bearing.
target_h = 0.003, fix_tolerance= + 5E-6, min_h = - 0.1,
default 2% anisotropic chordal control,

```
************************************************
*          CM2 SurfMesh T3(R) (5.6.0.0)          *
************************************************
Nodes           : 5312
Triangles       : 9164
Vertices        : 925 (merged: 662)
Curves          : 941 (merged: 450)
Surfaces        : 213
Area            : 1.336565E-002
Qmin            : 8.545688E-002
Read time       : 0.26 s.
Curvs mesh time : 0.14 s.
Surfs mesh time : 0.27 s.
Total mesh time : 0.42 s. (22028.84 t/s.)

************ HISTOGRAM QS *************
Total number of bins     :           11
Total number of counts   :         9164
Number of larger values  :            0
Number of smaller values :            0
V max                    : 9.997505E-001
V mean                   : 6.664721E-001
V min                    : 8.545688E-002

Bin number       -- Bin boundaries --          Hits

   10            0.90            1.00          1080
    9            0.80            0.90          1466
    8            0.70            0.80          1622
    7            0.60            0.70          1597
    6            0.50            0.60          1390
    5            0.40            0.50          1237
    4            0.30            0.40           647
    3            0.20            0.30           110
    2            0.10            0.20            14
    1            0.01            0.10             1
    0            0.00            0.01             0
```

Figure 24 – Bearing. Output info (triangles).
`target_h` = 0.003, `fix_tolerance`= + 5E-6, `min_h` = - 0.1,
default 2% anisotropic chordal control.

The number of triangles is reduced from 9254 to 9164.

CM2 SurfMesh Q4 in quad-dominant mode with same settings gives:

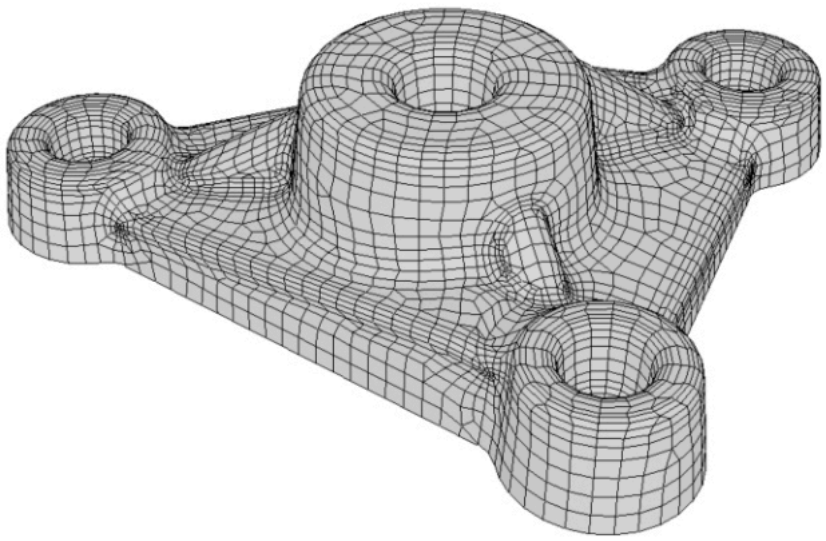


Figure 25 – Bearing. 4 209 quads, 288 triangles.
`target_h` = 0.003, `fix_tolerance`= + 5E-6, `min_h` = - 0.1,
default 2% anisotropic chordal control.

```
************************************************
*          CM2 SurfMesh Q4(R) (5.6.0.0)          *
************************************************
Nodes           : 5083
Elements        : 4497
  Quadrangles   : 4209 (93.60 %, 98.17 %)
  Triangles     : 288 (6.40 %, 1.83 %)
Vertices        : 925 (merged: 662)
Curves          : 941 (merged: 450)
Surfaces        : 213
Area            : 1.336572E-002
Qmin            : 1.395770E-001
Read time       : 0.24 s.
Curvs mesh time : 0.13 s.
Surfs mesh time : 0.32 s.
Total mesh time : 0.46 s. (9818.78 t/s.)

************ HISTOGRAM QS *************
Total number of bins     :           11
Total number of counts   :         4497
Number of larger values  :            0
Number of smaller values :            0
V max                    : 9.999652E-001
V mean                   : 6.586261E-001
V min                    : 1.395770E-001

Bin number       -- Bin boundaries --          Hits

   10            0.90            1.00           571
    9            0.80            0.90           542
    8            0.70            0.80           744
    7            0.60            0.70           874
    6            0.50            0.60           828
    5            0.40            0.50           548
    4            0.30            0.40           265
    3            0.20            0.30           112
    2            0.10            0.20            13
    1            0.01            0.10             0
    0            0.00            0.01             0
```

Figure 26 – Bearing. Output info (quad-dominant).
`target_h` = 0.003, `fix_tolerance`= + 5E-6, `min_h` = - 0.1,
default 2% anisotropic chordal control.

6. Specific mesh sizes

`settings_type::target_h` is the *global* default mesh size. As seen before, **CM2 SurfMesh T3/Q4** may reduce automatically the sizes on curved lines and surfaces, either in all directions (isotropic) or in selected directions (anisotropic) to honor the `settings_type::max_chordal_error` criterion.

However, **CM2 SurfMesh T3/Q4** allows more control over mesh sizes. Users can set *local* mesh sizes at specific CAD vertices, curves or surfaces (see fields `vertex_H`, `curve_H` and `surface_H` of struct `data_type`).

To illustrate this, let's consider this simple model for which we want specific mesh sizes near vertex #66, along curve #38 and on top and bottom surfaces (ids 2and 5):

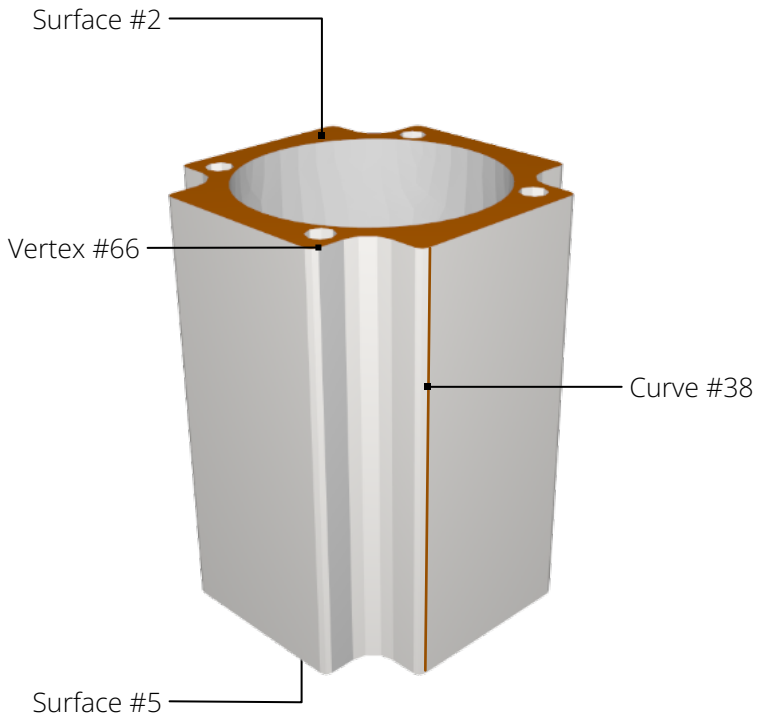


Figure 27 – Body.
IGES model (39 surfaces, 108 curves, 72 vertices).

☞ The vertex, curve and surface ids must follow the CAD indexing-scheme.
With OCCT, we use the CAD indexing scheme given by `TopTools_IndexedMapOfShape`.
Example:

```
TopTools_IndexedMapOfShape       vertices;
TopExp::MapShapes(root, TopAbs_VERTEX, vertices);  // Index all vertices of root
                                                   // consecutively starting from 1.
unsigned i = vertices.FindIndex(some_vertex);      // `i` is the index we use to refer
                                                   // to `some_vertex`.
```

☞ As OCCT follows a 1-base indexing scheme, `vertex_H[0]`, `curve_H[0]` and `surface_H[0]` are discarded.

☞ The index of an edge or face is also its *color*.

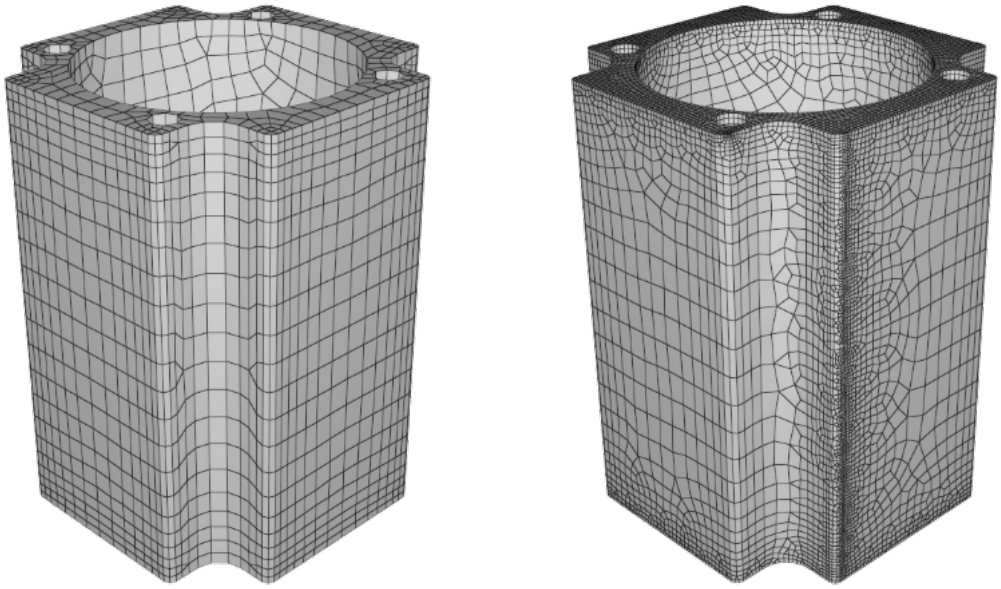


Figure 28 – Body. Quad-dominant mesh.
`target_h` = 7.0 only (left),
+ finer mesh sizes on top and bottom surfaces, along a curve and near a vertex (right),
`max_chordal_error` = -0.10 (i.e. 10% of radius), default anisotropic chordal control.

If we mesh only with a global `target_h` (we choose here value = 7.0) we get the mesh Figure 28 (left).

Note that **CM2 SurfMesh T3/Q4** reduces by itself the global target mesh size (7.0) where curvatures are high in order to respect the `settings_type::max_chordal_error` criterion. Here we chose a maximum of 10% of local radii and anisotropic control (the default control).

To specify special mesh sizes on the top surface (id = 2) and on the bottom surface (id = 5), we resize the `surface_H` array (empty by default) to at least 6 entries and populate the indices #2 and #5 with the specific mesh sizes (here a mesh size of 1.0 for both). The others, left to 0.0 or out-of-bound of the array will default to `target_h` = 7.0.

```
    the_mesher.settings.target_h = 7.;     // Default target size = 7.
    data.surface_H.resize(6, 0.);          // Reserve space in the array to control
                                           // the mesh sizes of the first five surfaces.
    data.surface_H[2] = 1.;                // Overwrite the default size for surface #1.
    data.surface_H[5] = 1.;                // Overwrite the default size for surface #4.
```

We can do the same on curves and vertices:

```
    data.curve_H.resize(39, 0.);           // Reserve space in the array to control
                                           // the mesh sizes of the first 38 curves.
    data.curve_H[38] = 0.50;               // Overwrite the default size for curve #38.

    data.vertex_H.resize(67, 0.);          // Reserve space in the array to control
                                           // the mesh sizes of the first 66 vertices.
    data.vertex_H[66] = 0.25;              // Overwrite the default size for vertex #66.
```

The resulting mesh is shown Figure 28 (right).

Another strategy, followed in the code below, is to resize the arrays to the number of surfaces, curves and vertices in the model and fill all of them with the same target size (here again 7.0), then overwrite the selected entity ids with whatever specific mesh sizes.

```
#include "stdafx.h"
#include <iostream>

// Simple optional display handler.
static void display_hdl (void*, unsigned, const char* msg)   { std::cout << msg; }


int main()
{
   surfmesh_q4::mesher              the_mesher;
   surfmesh_q4::mesher::data_type   data;

   // UNLOCK THE DLL.
   surfmesh_q4::registration("Licensed to SMART Inc.", "F5BEA10ABCWX");

   // DEFINE SETTINGS, DATA AND RUN THE MESHER.
   the_mesher.settings.max_chordal_error = -0.10;  // Max distance of 10% of radii.
   the_mesher.settings.max_gradation = 0.2;        // Low gradation for nicer rendering.
   data.surface_H.resize(40, 7.0);                 // Size 7.0 on all surfaces.
   data.curve_H.resize(109, 7.0);                  // Size 7.0 on all curves.
   data.vertex_H.resize(73, 7.0);                  // Size 7.0 on all vertices.
   data.surface_H[2] = data.surface_H[5] = 1.0;    // Change size to 1.0 for surfaces #2 and #5.
   data.curve_H[38] = 0.50;                        // Change size to 0.5 for curve #38.
   data.vertex_H[66] = 0.25;                       // Change size to 0.25 for vertex #66.

   the_mesher.run(“body.iges”, data);

   // SOME OUTPUT INFO (OPTIONAL).
   data.print_info(&display_hdl);

   // VISUALIZATION (OPTIONAL).
   meshtools::medit_output("out.mesh", data.pos, connectM_, CM2_FACE_MIX, data.colors);

   return 0;
}   // main
```

☞ Setting value 7.0 by default to all curves and vertices isn't strictly necessary since the values affected to surfaces are transferred to their children curves and recursively to their grand-children vertices, unless valid mesh sizes are provided for them in `curve_H` or `vertex_H`.

☞ This assumes that the ids of the CAD entities are known (using `TopTools_IndexedMapOfShape` indexing).

7. Quadratic elements

Beyond the `meshtools2d::convert_into_high_order` function (and its simplified version `meshtools2d::convert_into_quadratic`, see CM2 MeshTools HTML reference manual), **CM2 SurfMesh T3/Q4** have embedded facilities to generate directly quadratic elements (6-node triangles and 8-node or 9-node quadrangles).

With **CM2 SurfMesh T3/Q4**, high-order nodes can be exactly positioned onto the curves and surfaces, whereas `meshtools2d::convert_into_high_order` (and friends) generate these nodes through linear interpolation of the base nodes (though boundary edges can use specific curved high-order nodes).

The generation of quadratic elements is governed by two parameters: `settings_type::high_order_type` and `settings_type::high_order_mode`.

high_order_type

Controls the type of generated elements. The value can be:

- 0: Elements remain linear (3-node triangles and 4-node quadrangles). This is the default.
- 1: Elements are 6-node triangles and 8-node (so-called serendipity) quadrangles.
- 2+: Elements are 6-node triangles and 9-node (full) quadrangles.

high_order_mode

Controls the position of the high-order nodes (when `high_order_type` > 0). The value can be:

- 0: The high-order nodes are interpolated between the linear 3D nodes. Elements remain straight.
  This is equivalent to a call to `meshtools2d::convert_into_high_order`.
- 1: The high-order nodes are on the curves/surfaces and have reference coordinates in the middle between the linear nodes but may not be equidistant between the linear 3D nodes. Elements are curved.
  2+: The high-order nodes are on the curves/surfaces and equidistant between the linear 3D nodes but may not have reference values in the middle between the linear nodes. Elements are curved. This is the default.

☞ The considered chordal error is always the maximum distance between the *linear* elements and the curves/surfaces.

With linear elements of size h along a curve of local radius R, the chordal error is approximately given by

$$\delta = \frac{h^2}{8\,R}$$

Formula 2 – Chordal error between linear element of size *h* and a circle of radius *R*.

The chordal error δ is divided by 4 when the size h is divided by 2.

With quadratic elements, the maximum distance to the curve (abusively called also chordal error) is approximately given by

$$\delta = \frac{h^4}{512\,R^3}$$

Formula 3 – Chordal error between quadratic element of size *h* and a circle of radius *R*.

The chordal error δ is divided by 16 when the size h is divided by 2.
A linear relative chordal error of 40% translates into a quadratic relative chordal error of 2%.

```
#include "stdafx.h"
#include <iostream>

// Simple optional display handler.
static void display_hdl (void*, unsigned, const char* msg)   { std::cout << msg; }


int main()
{
   surfmesh_q4::mesher              the_mesher;
   surfmesh_q4::mesher::data_type   data;

   // UNLOCK THE DLL.
   surfmesh_q4::registration("Licensed to SMART Inc.", "F5BEA10ABCWX");

   // DEFINE SETTINGS AND RUN THE MESHER.
   the_mesher.settings.high_order_type = 2;          // T6/Q9.
   the_mesher.settings.max_chordal_error = -0.40;    // To get a max distance of 2%.
   the_mesher.settings.target_h = 0.006;             // Elements twice larger.
   the_mesher.run(“bearing.iges”, data);

   // SOME OUTPUT INFO (OPTIONAL).
   data.print_info(&display_hdl);

   // VISUALIZATION (OPTIONAL).
   UIntMat   connectM_;
   connectM_.copy(data.connectM);
   meshtools2d::convert_into_linear(connectM_);      // MEDIT can’t process quadratic elements.
   meshtools::medit_output("out.mesh", data.pos, connectM_, CM2_FACE_MIX, data.colors);

   return 0;
}   // main
```

COMPUTING OBJECTS

CM2 SurfMesh® T3/Q4 for OpenCascade®

Version 5.6

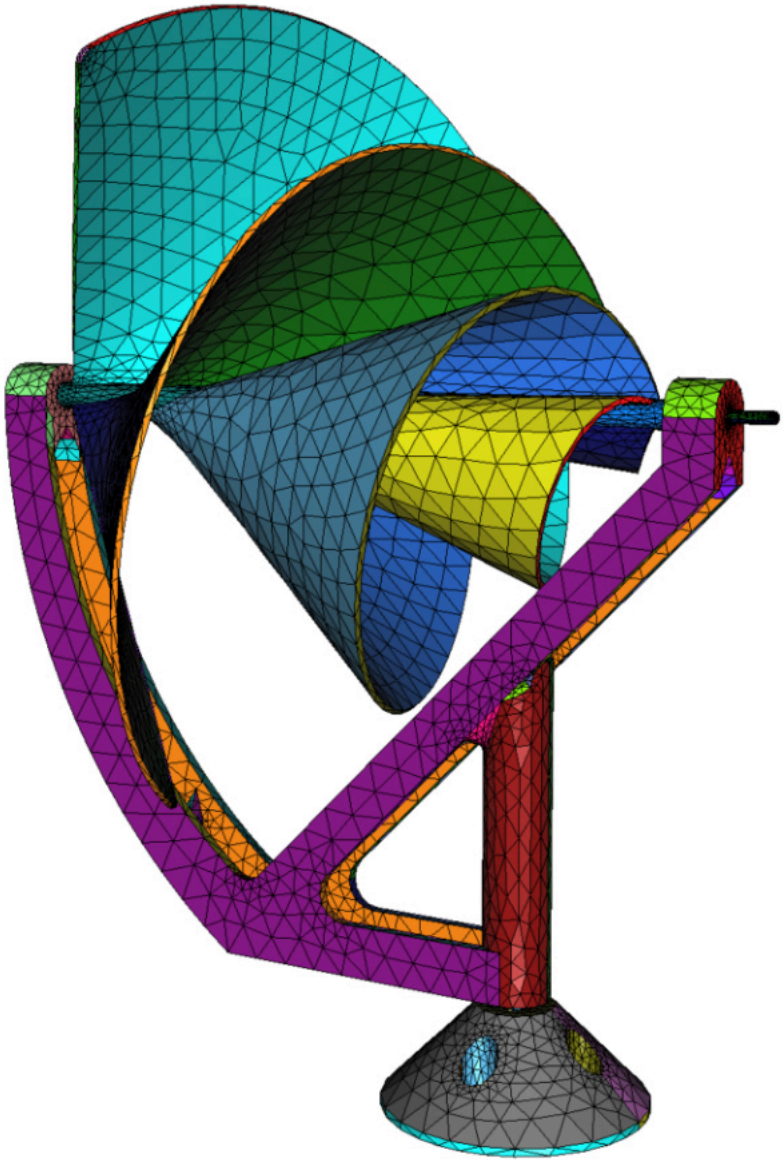
Mesh gallery

Revision February 2025.

https://www.computing-objects.com

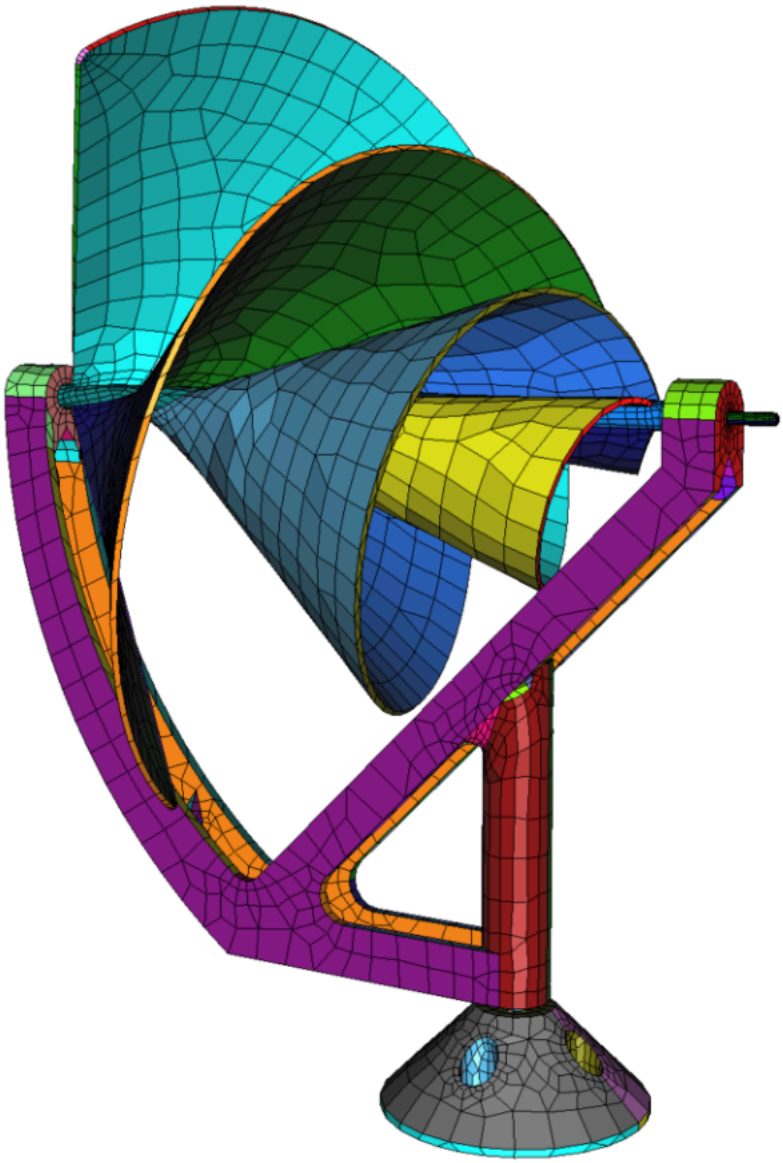
Archimedes turbine

`target_h` = 10, all other settings to their default.



```
*******************************************************
*             CM2 SurfMesh T3(R) (5.6.0.0)            *
*******************************************************
Nodes           : 10694
Triangles       : 21528
Vertices        : 394 (merged: 8)
Curves          : 622 (merged: 12)
Surfaces        : 258
Area            : 2.119694E+05
Qmin            : 2.314685E-02
Read time       : 5.59 s.
Curvs mesh time : 0.16 s.
Surfs mesh time : 0.48 s.
Total mesh time : 0.65 s. (32967.85 t/s.)
```

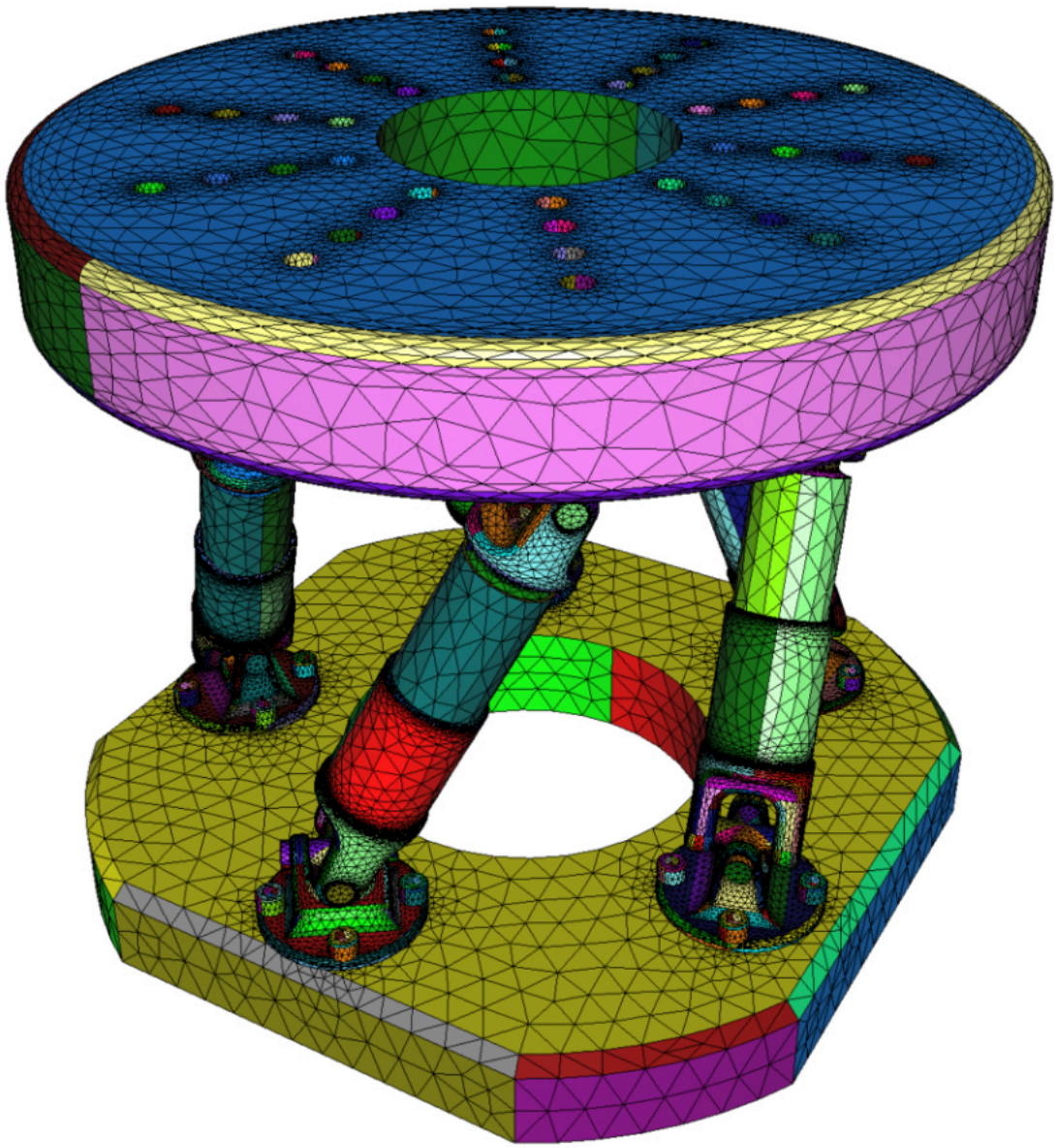
The output information given here are only indicative. All runs were done with x64 CM2 libs (Visual Studio 2022 MD build) on Windows 8.1 x64 with Intel Xeon E3-1270 V2 3.5 Ghz (4 cores with hyper-threading, turbo boost disabled).



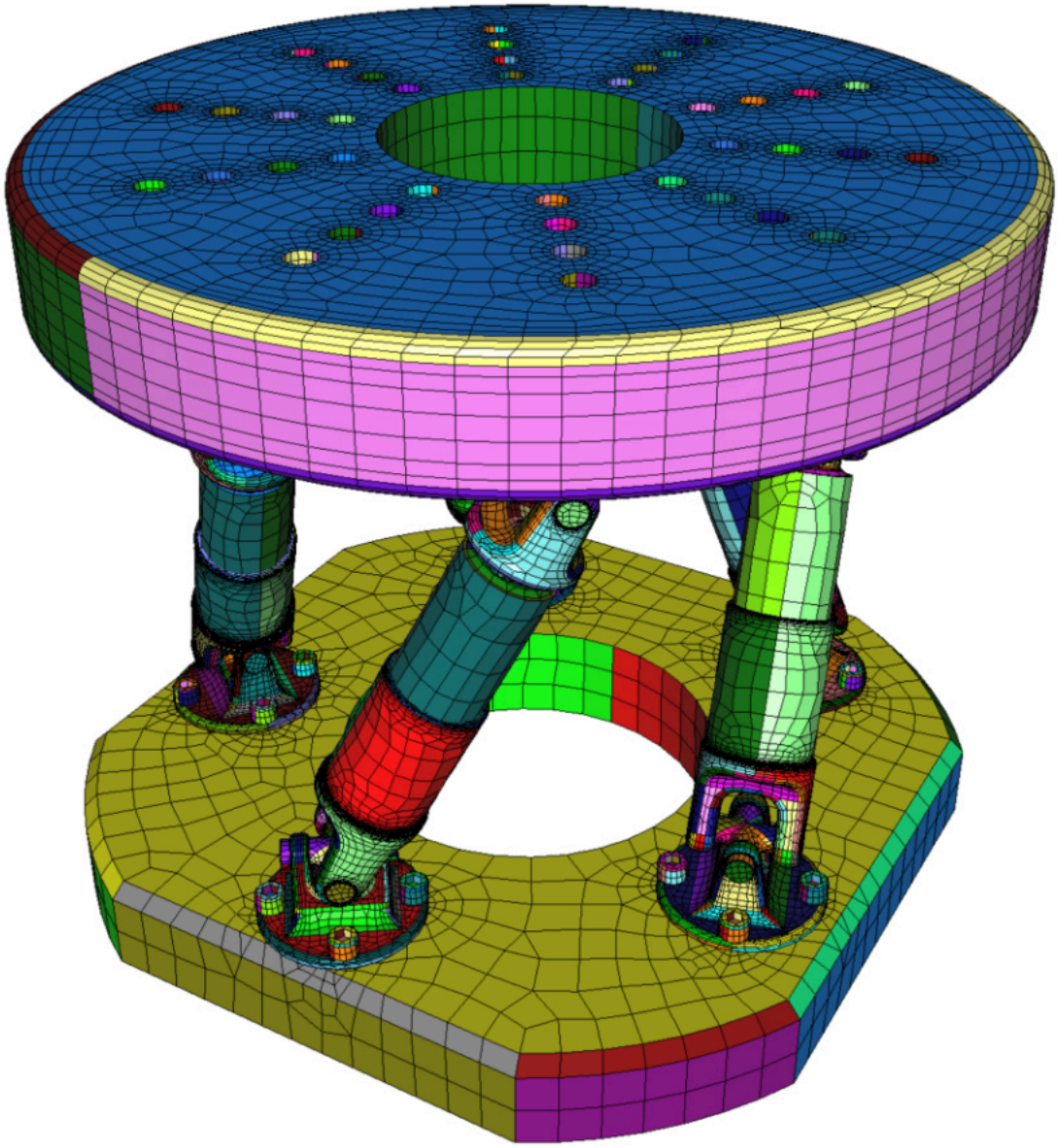
```
*******************************************************
*            CM2 SurfMesh Q4(R) (5.6.0.0)             *
*******************************************************
Nodes           : 9818
Elements        : 10199
  Quadrangles   : 9577 (93.90 %, 97.64 %)
  Triangles     : 622 (6.10 %, 2.36 %)
Vertices        : 394 (merged: 8)
Curves          : 622 (merged: 12)
Surfaces        : 258
Area            : 2.120434E+05
Qmin            : 4.838189E-02
Read time       : 1.30 s.
Curvs mesh time : 0.17 s.
Surfs mesh time : 0.53 s.
Total mesh time : 0.71 s. (14385.05 t/s.)
```

Hexapod

`target_h` = 20, all other settings to their default.



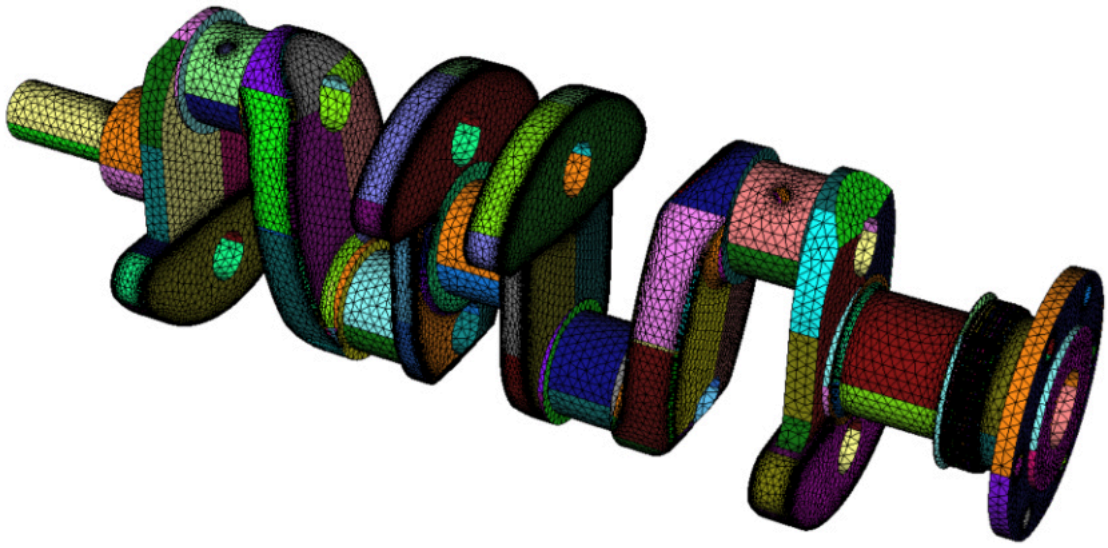
```
*********************************************************
*             CM2 SurfMesh T3(R) (5.6.0.0)              *
*********************************************************
Nodes           : 152178
Triangles       : 303116
Vertices        : 4348 (merged: 80)
Curves          : 7307 (merged: 80)
Surfaces        : 3198
Area            : 1.784679E+06
Qmin            : 3.554735E-03
Read time       : 1.59 s.
Curvs mesh time : 1.75 s.
Surfs mesh time : 6.76 s.
Total mesh time : 8.54 s. (35506.15 t/s.)
```



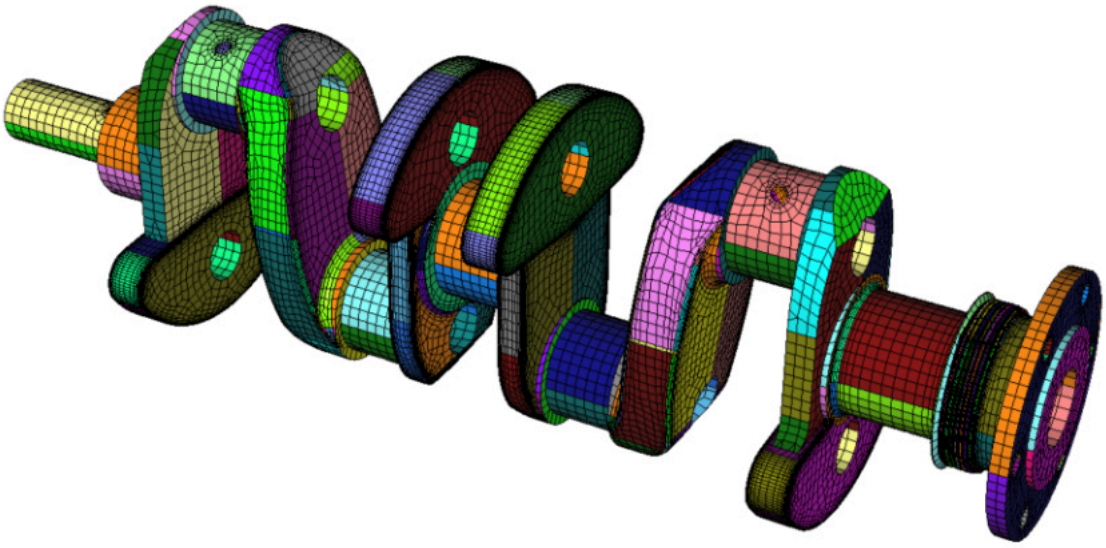
```
*******************************************************
*            CM2 SurfMesh Q4(R) (5.6.0.0)             *
*******************************************************
Nodes           : 137917
Elements        : 149544
  Quadrangles   : 125050 (83.62 %, 97.83 %)
  Triangles     : 24494 (16.38 %, 2.17 %)
Vertices        : 4348 (merged: 80)
Curves          : 7307 (merged: 80)
Surfaces        : 3198
Area            : 1.784513E+06
Qmin            : 8.804998E-03
Read time       : 1.39 s.
Curvs mesh time : 1.76 s.
Surfs mesh time : 8.05 s.
Total mesh time : 9.84 s. (15202.20 t/s.)
```

Krank Mili

`target_h` = 5, `fix_tolerance` = -0.05, all other settings to their default.



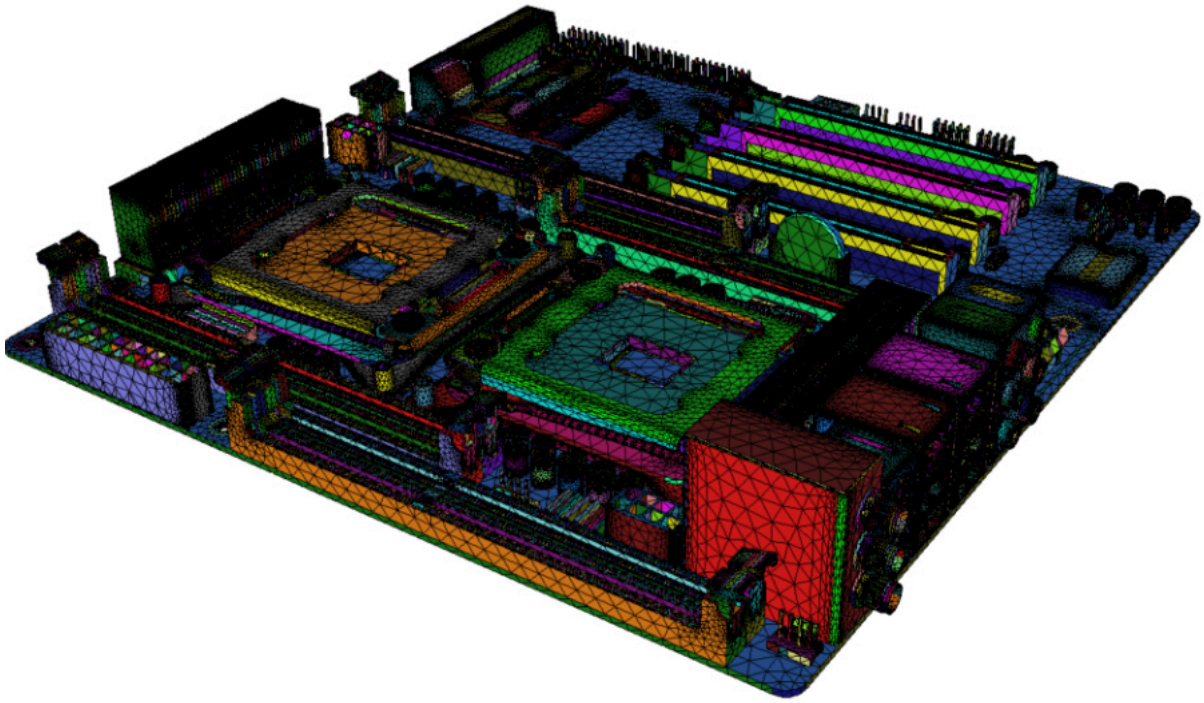
```
*******************************************************
*            CM2 SurfMesh T3(R) (5.6.0.0)             *
*******************************************************
Nodes           : 48959
Triangles       : 94568
Vertices        : 2369 (merged: 1638)
Curves          : 2370 (merged: 1183)
Surfaces        : 466
Area            : 3.002293E+05
Qmin            : 1.080618E-02
Read time       : 1.07 s.
Curvs mesh time : 0.44 s.
Surfs mesh time : 1.49 s.
Total mesh time : 1.94 s. (48646.09 t/s.)
```



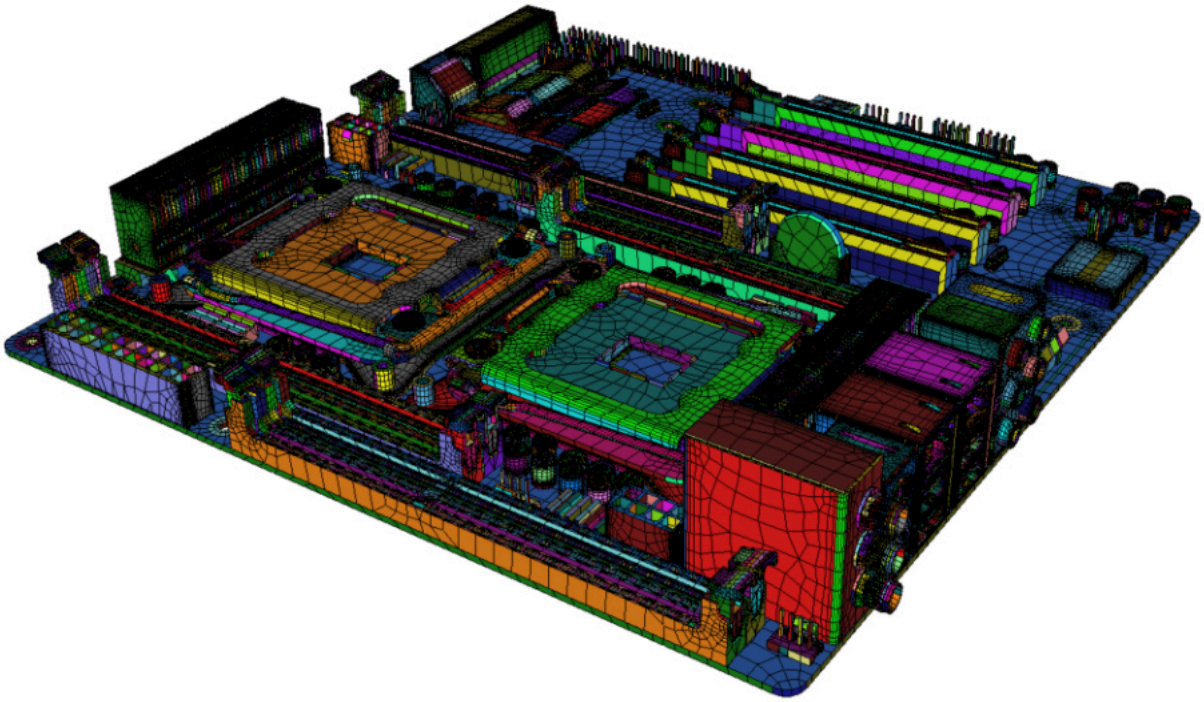
```
*******************************************************
*            CM2 SurfMesh Q4(R) (5.6.0.0)             *
*******************************************************
Nodes           : 44929
Elements        : 48039
  Quadrangles   : 38469 (80.08 %, 97.59 %)
  Triangles     : 9570 (19.92 %, 2.41 %)
Vertices        : 2369 (merged: 1638)
Curves          : 2370 (merged: 1183)
Surfaces        : 466
Area            : 3.001658E+05
Qmin            : 1.208474E-02
Read time       : 1.01 s.
Curvs mesh time : 0.45 s.
Surfs mesh time : 1.82 s.
Total mesh time : 2.28 s. (21069.74 t/s.)
```

Micro ATX

`target_h` = 5,, all other settings to their default.



```
*******************************************************
*            CM2 SurfMesh T3(R) (5.6.0.0)             *
*******************************************************
Nodes           : 652530
Triangles       : 1308244
Vertices        : 70720 (merged: 830)
Curves          : 107249 (merged: 968)
Surfaces        : 40748
Area            : 5.459773E+05
Qmin            : 4.812106E-04
Read time       : 24.51 s.
Curvs mesh time : 26.09 s.
Surfs mesh time : 31.39 s.
Total mesh time : 57.94 s. (22577.73 t/s.)
```



```
*******************************************************
*            CM2 SurfMesh Q4(R) (5.6.0.0)             *
*******************************************************
Nodes           : 623724
Elements        : 659020
  Quadrangles   : 591612 (89.77 %, 97.39 %)
  Triangles     : 67408 (10.23 %, 2.61 %)
Vertices        : 70720 (merged: 830)
Curves          : 107249 (merged: 968)
Surfaces        : 40748
Area            : 5.459561E+05
Qmin            : 4.529884E-04
Read time       : 24.21 s.
Curvs mesh time : 25.98 s.
Surfs mesh time : 40.94 s.
Total mesh time : 67.38 s. (9780.50 t/s.)
```

COMPUTING
OBJECTS