An introduction to

**MeshGems-Hexa V1.1**

A fast, robust, automated hexahedral mesher

For further information, contact:

DISTENE

2, Rue de la Piquetterie, 91680 Bruyères le Châtel FRANCE

Tel: +33 (0) 970 650 219
Fax: +33 (0) 169 269 033
Mail: info@meshgems.com

*Ask for an evaluation copy of MeshGems-Hexa from our web site now!*  
(http://www.meshgems.com)
A. Introduction

A.1. The MeshGems® suite

While numerical simulation has become a vital element now strongly integrated in product development cycles, the need for efficient and reliable meshing algorithms has become even stronger than ever before. Fast and realistic 3D simulations on very complex geometries have become standard in industry, pointing out the weaknesses of many meshing products on the market.

It is today mandatory to be able to transform very efficiently a real-life geometry into a mathematical representation usable by scientific simulation software. This can be achieved only if constraints such as reliability, or high mesh generation speed, can be addressed to reduce considerably the human effort necessary to exploit numerical simulation in a conception cycle.

This constant search for a reduction of costs naturally leads to considering fast and quality solutions to speed up the mesh generation step in a study.

It is for these various reasons that DISTENE and their academic research partners have been working for 20 years on automatic and reliable meshing algorithms. These are now composing the MeshGems® suite, which provides the industry with a product line for an easy and quick generation of high quality unstructured meshes.

A.2. What is MeshGems-Hexa?

MeshGems-Hexa (formerly know as “Hexotic”) is a meshing software component which creates fully hexahedral volumes meshes automatically, starting from a triangulated closed surface defining a volume, with little or no user interaction required. It aims at providing a very efficient and reliable hexahedral meshing component for numerical simulations.

MeshGems-Hexa basically requires a set of triangles and corresponding vertices defining a closed surface. Additional information may be provided such as sharp angles, surface and volumes identification numbers, or the minimum and maximum size of hexahedra.

The result of MeshGems-Hexa is an output mesh composed of connected and conformal hexahedra, each possessing a positive Jacobian number, and with a density and accuracy controlled by the user. In particular, MeshGems-Hexa does not generate other element types like pyramids, pentahedra, tetrahedral etc. It does not create non conformal meshes with hanging nodes either. A new surface made of quadrilaterals is also generated at the same time. MeshGem-Hexa tries to preserve the input surface features like sharp angles, curvatures or material references in order to minimize the distance between both surfaces.

MeshGems-Hexa is a tolerant mesher, as it accepts invalid input meshes (containing self intersections for instance), even though best results are obtained with a valid input triangulation. It comes as an executable or software library, which can be plugged into a third party environment.

A.3. What MeshGems-Hexa is not...
MeshGems-Hexa is not intended to be a general purpose end-user meshing software. In particular it does not have a GUI, and it does not read CAD files, or solver specific mesh formats directly.

MeshGems-Hexa uses meshes as input much the way the other meshing component MeshGems-Tetra does, but unlike it, it cannot enforce the input mesh to be the conformal skin of the resulting hexahedral mesh. The input mesh is only used as a geometric support, like some sort of a discreet CAD model. After generating the volume hexahedral mesh, its quadrilateral boundary is projected onto the input surface, thus preserving the geometry, but not the initial mesh provided.

MeshGems-Hexa is not an anisotropic mesher (except in the very specific case of boundary layers, see C.5), which means that it cannot generate high aspect ratio elements like beams or plates in thin geometries, but only cube-like hexes. In order to capture such geometries, like, say a wing, it has to generate elements small enough to fit inside the wing’s thickness. Consequently, meshing thin geometries requires high densities of elements.

It is not a “hexahedral dominant” mesher either. It cannot handle other kinds of elements like tetrahedra or prisms. Although a pure hex mesher and a hex dominant mesher might look quite alike, the internal structures and algorithms strongly differ. If you really need hybrid meshes or surface conformal meshes, you ought to consider using the upcoming dedicated product MeshGems-Hybrid (see section “Ongoing R&D” below).

Although MeshGems-Hexa is fault tolerant, the MeshGems suite offers other components for these needs, such as MeshGems-Cleaner, which heals surface meshes by removing self–intersections, gaps, holes, non-conformities, badly shaped elements, etc. - even if this can alter locally - but in a very limited way - the geometry.

B. Overview of MeshGems-Hexa V1.1

MeshGems-Hexa is a volume mesher based on the Octree method. It goes through several steps to produce a volume mesh:

- Analysis of the input surface and computation of the feature sizes in every parts of the geometry,
- generation of a so-called “background mesh”, which is a pure hexahedral mesh encompassing the object’s bounding box with its element sizes matching locally the sizes of the input surface features,
- removal of elements lying outside the object,
- computation of a mapping between the output quad surface and the input triangulated surface,
- projection of the output surface nodes onto the input geometry and smoothing of the internal nodes in order to maintain a good quality of cells.

Here is below an example of the method for the pure hexahedral meshing of a sphere

```
```

“background mesh”

in red: elements to be removed
projection of the output surface nodes onto the input geometry

smoothing of the internal nodes

All previous features are kept in this latest release, including:

- fully automated 100% hex generation with size control and variable density,
- processing of complex multiple domains, including non manifold geometries,
- parallel mesh generation,
- adaptation capabilities, thanks to a user defined size map

C. What’s new with respect to the previous version ?

The release 1.1 of **MeshGems-Hexa** includes several new features which were not available in Hexotic. The main ones are presented here.

C.1. Application programming interface (API)

MeshGems-Hexa offers a comprehensive and stable API, which allows integrators to exploit fully and simply all the features supported. This API is fully described in the user documentation.

C.2. more control

More controls were added to **MeshGems-Hexa** to make it much easier to use and control.

- **Curvature driven mesh refinement**
  
  This basically controls the density so as to get finer meshes in high curvature areas, using the angle that two underlying triangles describing the geometry make. Specifically, one has simply to specify a threshold angle which triggers the refinement: if the local angles are greater than the threshold, the mesh will be locally refined until the threshold angle is met.

  This is illustrated in the example below, where a curvature refinement with a threshold angle of 5° was required.
• Controlling volume subdomain numbering
  It is now possible to control the way the volume regions are numbered.

• Hex mesh size controlled by memory settings
  MeshGems-Hexa is capable of stopping and saving the mesh obtained so far when the
  available memory is exhausted. The next version will allow even more control of this sort.

• Watertightness enforcement
  When the input is not watertight, MeshGems-Hexa may fail. However, a new option is capable
  of automatically filling holes.

C.3. more information
MeshGems-Hexa also now provides additional information which may prove helpful to integrators,
  such as:
  • Dual way mapping surface information: each input triangle is linked to a single generated
    quad on the surface, and each quadrangle generated is linked to a single surface triangle.
    This makes sure that it is possible to map the original surface onto the generated surface,
    and vice-versa.
  • Edge references: if input edges are provided, their attributes will be transferred to the
    resulting edges belonging to the generated volume.
  • Geometry accuracy information: this highlights areas where the mesher could not reach its
    subdivision level (because of user requirements or memory limits), by providing the list of
    coarse hexes.

C.4. better cell quality
There may be configurations where the geometry accuracy may impose strong constraints on the shape
  of elements, resulting in locally poor quality elements. MeshGems-Hexa now proposes a new
  optimisation process which flattens the sides of the hexahedra, to improve the numerical behaviour of
  solvers on these elements.

This is illustrated on the example below:
C.5. Boundary layers

A new outstanding feature of MeshGems-Hexa is its ability to generate boundary layers automatically. Boundary layers are sets of very anisotropic elements particularly useful to capture specific physical phenomena occurring very close to bodies and evolving rapidly in the direction perpendicular to the surface (like for example in CFD or computational electromagnetics). Given local parameters on surfaces, like the number of layers, the growth rate and the height of the first layer, MeshGems-Hexa is able to generate boundary layers automatically, as well as the blending layers between anisotropic and isotropic areas.

The user has control over the number, size, growth factor as well as the “stem surface references”, that is, the references of the surfaces from which the layers will be extruded.

Additional blending layers may be inserted between the first set of physical layers and the internal hexahedral mesh in order to mitigate the size transition so that no element is more than twice larger than its neighbours.

An example of hexahedral mesh with boundary layers is illustrated below:
Hexahedral mesh with boundary and blending layers (30 boundary layers, with a first height of 0.001, a geometric progression factor of 1.1, and 8 blending layers)

Hexahedral mesh generated (view of the airplane body)

Boundary and blending layers around wing

Boundary layers between wing and main body
D. The features of MeshGems-Hexa V1.1

D.1. Hexahedral mesh generation out of a triangulated surface

The volume mesh is generated automatically and requires no user’s intervention. The user has however control over the generated volume through two different ways:

- **Element size control**: if no parameters are given, MeshGems-Hexa automatically computes the hexes sizes according to the local geometry features sizes. Since it is based on the octree method, which splits elements recursively in two, the hexes sizes are multiples of power of two relative to the object bounding box. Default minimum and maximum subdivision levels are 6 and 10 respectively, which means that the smallest hexes are $1/2^{10} = 1/1024$th of the bounding box and the biggest allowed are $1/64$th.

Various levels of refinement

- **Volume subdomains recovery**: default mode is to mesh every concentric subdomains and assign them an increasing reference ID. You can also tell the mesher to keep only the outmost subdomain, or to fuse all of them in one piece, or to handle non-manifold geometries.

Standard processing : all cavities are meshed  
Non-manifold specific processing: separate subdomains are created

D.2. Respect of input geometrical features

Even though MeshGems-Hexa only uses the input triangulation as a support describing the geometry; it is capable of identifying surface sharp edges based on a threshold angle. Any pair of adjacent triangles for which their dihedral angle is sharper than the threshold is considered a “ridge”, which is a geometrical feature that must be preserved in the final hexahedral mesh surface.
If two triangles sharing a common edge have different attributes, this edge is considered as a reference-ridge, regardless of the angle between the triangles. Consequently, the boundary between different materials will be enforced in the output mesh and surface quadrilaterals produced will inherit the triangle reference they are projected on. In any case, attributes, which are usually used to differentiate components of a geometry assembly, physical properties or to set boundary conditions for the solver, are transferred from the input surface geometry onto the volume mesh.

Volume ID may also be provided to control finely the enumeration of inner subdomains.

**D.3. Mesh Adaptation**

*MeshGems-Hexa* can be used in an “a posteriori“ adaptation scheme. If it is provided with a size map, a set of nodes with associated required local sizes, the background mesh generated in the second meshing step will match both the geometrical features sizes and the sizes required by the computation (if the two required sizes differ locally, the smallest one is selected).

Note that MeshGems-Hexa uses a global adaptation scheme, and not a local modification scheme. It regenerates the whole mesh for each adaptation step. The algorithm is fast enough to regenerate the whole mesh at each adaptation step without penalizing the whole computation.
E. **MeshGems-Hexa, a software product of industrial quality**

**E.1. Application Programming Interface (API)**

MeshGems-Hexa offers a comprehensive and stable API, which allows integrators to exploit fully and simply all the features supported. This API is described in the user documentation.

**E.2. Reliability**

Thanks to a very robust method, MeshGems-Hexa is very reliable and always produces an output, provided the input mesh is valid and describes a closed volume. As are all components of the MeshGems suite, MeshGems-Hexa benefits from a thorough validation on a very large and representative database.

**E.3. Speed and quality**

Depending on the options, the size of the generated mesh and the hardware, one can reasonably expect to complete MeshGems-Hexa on a PC in a few minutes.

MeshGems-Hexa algorithm is fast and generates typically about 4 million cells per minute on recent quad-core machines (i7 Intel Core based for example). This is achieved thanks to a multithreaded parallel implementation of the code.

Its memory footprint is around 1.5 million elements per GigaByte.

When projecting the nodes on the real surface, the algorithm tries to balance between quality and geometrical accuracy. Using the scaled Jacobian quality criterion, any node displacement that would lead to a negative ratio is refused. This approach may very locally move nodes off the input geometry for the sake of cell quality.

We show below some test cases and the corresponding measurements (quality and speed) obtained when running MeshGems-Hexa.

The first table below illustrates quality measurements on a set of industrial test cases meshed with MeshGems-Hexa, and default values for mesh density and accuracy. Note that the quality $q$ computed currently by MeshGems-Hexa is based on:
• the minimal volume $V_{\text{min}}$ of the 10 elementary tetrahedral contained in the hexahedron considered
• the length $l_i$ of the edges of the hexahedron

Using the formula:

$$q = 24\sqrt[3]{\frac{V_{\text{min}}}{\left(\sum_{i=1}^{12} l_i\right)^{3/2}}}$$

We show in particular in this table:

• The number of hexahedra generated by MeshGems-Hexa
• The average quality of the hexahedra produced.
• The quality of the worst hexahedron produced.

<table>
<thead>
<tr>
<th>Test case</th>
<th>number of hexahedra</th>
<th>Average quality</th>
<th>Worst quality</th>
<th>elapsed time (s)</th>
<th>speed (million hexes/mn)</th>
</tr>
</thead>
<tbody>
<tr>
<td>retaining_arm</td>
<td>726 108</td>
<td>0.628</td>
<td>0.052</td>
<td>9.4</td>
<td>4.6</td>
</tr>
<tr>
<td>actuator</td>
<td>1 276 672</td>
<td>0.692</td>
<td>0.054</td>
<td>17.4</td>
<td>4.4</td>
</tr>
<tr>
<td>gripper_wheel</td>
<td>51 201</td>
<td>0.677</td>
<td>0.063</td>
<td>0.7</td>
<td>4.2</td>
</tr>
<tr>
<td>mech05</td>
<td>505 246</td>
<td>0.703</td>
<td>0.051</td>
<td>6.2</td>
<td>4.9</td>
</tr>
<tr>
<td>micro-processor</td>
<td>219 768</td>
<td>0.659</td>
<td>0.096</td>
<td>4.4</td>
<td>3.0</td>
</tr>
<tr>
<td>motor</td>
<td>318 478</td>
<td>0.604</td>
<td>0.061</td>
<td>6.2</td>
<td>3.1</td>
</tr>
<tr>
<td>radiator</td>
<td>7 872 444</td>
<td>0.751</td>
<td>0.054</td>
<td>31.7</td>
<td>14.9</td>
</tr>
<tr>
<td>fillet</td>
<td>8 964 642</td>
<td>0.727</td>
<td>0.051</td>
<td>25.3</td>
<td>21.3</td>
</tr>
<tr>
<td>filter</td>
<td>292 638</td>
<td>0.640</td>
<td>0.056</td>
<td>4.0</td>
<td>4.4</td>
</tr>
<tr>
<td>gear_part</td>
<td>215 329</td>
<td>0.634</td>
<td>0.054</td>
<td>3.8</td>
<td>3.4</td>
</tr>
<tr>
<td>propeller</td>
<td>513 071</td>
<td>0.637</td>
<td>0.062</td>
<td>10.4</td>
<td>3.0</td>
</tr>
<tr>
<td>air_filter</td>
<td>12 893 008</td>
<td>0.658</td>
<td>0.053</td>
<td>26.5</td>
<td>29.2</td>
</tr>
<tr>
<td>crank</td>
<td>11 738</td>
<td>0.669</td>
<td>0.091</td>
<td>0.2</td>
<td>3.1</td>
</tr>
<tr>
<td>steering_wheel</td>
<td>96 778</td>
<td>0.707</td>
<td>0.124</td>
<td>1.7</td>
<td>3.3</td>
</tr>
<tr>
<td>fastener</td>
<td>32 261</td>
<td>0.646</td>
<td>0.095</td>
<td>0.4</td>
<td>4.4</td>
</tr>
<tr>
<td>valve</td>
<td>49 429</td>
<td>0.805</td>
<td>0.104</td>
<td>0.7</td>
<td>4.0</td>
</tr>
</tbody>
</table>

MeshGems-Hexa’s quality and speed on several test cases (pictures pages 13 and 14)

The next table illustrates speed measurements on the set of test cases, in particular:

• The number of hexahedra generated by MeshGems-Hexa
• The speed (in number of elements per minute)
• The elapsed time (in seconds)
• Default options are used for mesh density and geometry accuracy
The hardware used is a PC equipped with Intel I7-2600s@2.8GHz (4 physical cores used in parallel)

<table>
<thead>
<tr>
<th>Test case</th>
<th>number of hexahedra</th>
<th>elapsed time (s)</th>
<th>speed (million hexes/mn)</th>
</tr>
</thead>
<tbody>
<tr>
<td>retaining_arm</td>
<td>726 108</td>
<td>9.4</td>
<td>4.62</td>
</tr>
<tr>
<td>actuator</td>
<td>1 276 672</td>
<td>17.4</td>
<td>4.41</td>
</tr>
<tr>
<td>gripper_wheel</td>
<td>51 201</td>
<td>0.7</td>
<td>4.18</td>
</tr>
<tr>
<td>mech05</td>
<td>505 246</td>
<td>6.2</td>
<td>4.15</td>
</tr>
<tr>
<td>microprocessor</td>
<td>219 768</td>
<td>4.4</td>
<td>4.87</td>
</tr>
<tr>
<td>motor</td>
<td>318 478</td>
<td>6.2</td>
<td>3.03</td>
</tr>
<tr>
<td>radiator</td>
<td>7 872 444</td>
<td>31.7</td>
<td>3.11</td>
</tr>
<tr>
<td>fillet</td>
<td>8 964 642</td>
<td>25.3</td>
<td>14.90</td>
</tr>
<tr>
<td>filter</td>
<td>292 638</td>
<td>4.0</td>
<td>21.27</td>
</tr>
<tr>
<td>gear_part</td>
<td>215 329</td>
<td>3.8</td>
<td>4.45</td>
</tr>
<tr>
<td>propeller</td>
<td>513 071</td>
<td>10.4</td>
<td>3.44</td>
</tr>
<tr>
<td>air_filter</td>
<td>12 893 008</td>
<td>26.5</td>
<td>2.96</td>
</tr>
<tr>
<td>crank</td>
<td>11 738</td>
<td>0.2</td>
<td>29.22</td>
</tr>
<tr>
<td>steering_wheel</td>
<td>96 778</td>
<td>1.7</td>
<td>3.06</td>
</tr>
<tr>
<td>fastener</td>
<td>32 261</td>
<td>0.4</td>
<td>3.34</td>
</tr>
<tr>
<td>valve</td>
<td>49 429</td>
<td>0.7</td>
<td>4.40</td>
</tr>
</tbody>
</table>

MeshGems-Hexa’s speed on some test cases (pictures pages 13 and 14)
Pump case: cut planes

Pump case: output quadrilateral surface

Actuator case: cut planes

Actuator case: output quadrilateral surface

Microprocessor case: cut planes

Microprocessor case: output quadrilateral surface
Mech05 case

Propeller case

Crank case

Gear_part case

Gripper_wheel case

Retaining_arm case

Valve case

Filter case
E.4. Platforms

MeshGems-Hexa is available as dynamic libraries or executables on these operating systems:

- Windows (2000, XP, Vista, Seven, 32 bit or 64 bit operating systems),
- Linux (32 bit and 64 bit operating systems),
- Linux (32 bit and 64 bit operating systems).

Other operating systems are also available upon request.

F. Ongoing R&D

Several topics of interest are currently being investigated for automatic hexahedral meshing. These include:

- Optimising the code of MeshGems-Hexa for better results and geometry recovery accuracy.
- Reducing the cell count of MeshGems-Hexa
- Optimizing the parallel performance: a prototype of MeshGems-Hexa already takes advantage of the new graphics cards computing capabilities (GPUs). It pushes the software speed even higher than the multithreaded version.
- Hybrid meshing, when geometry accuracy, and even more, the input surface mesh, must be kept as it is and should not be altered in any way while keeping the mesh conformal. A prototype of a new product (MeshGems-Hybrid) is now capable of creating hybrid meshes containing a mixture of tetrahedra, pyramids, pentahedra, and hexahedra. This prototype is capable of dealing with mixed triangular/quadrilateral surfaces and combines several meshing methods in order to maximize the number of pentahedra and hexahedra in the volume.

This is illustrated below: