User Manual

GiD

The universal, adaptative and user
friendly pre and postprocessing
system for computer analysis

Developers

Miguel Pasenau de Riera

Enrique Escolano Tercero

Abel Coll Sans

Adrià Melendo Ribera

Anna Monros Bellart

Javier Gárate Vidiella

Mª Rosa Peyrau Rubio

For further information please contact

International Center for Numerical Methods in Engineering
Edificio C1, Campus Norte UPC
Gran Capitán s/n, 08034 Barcelona, Spain
http://www.gidsimulation.com
gid@cimne.upc.edu

Contents

INTRODUCTION

This manual contains a collection of tutorials and practical information for learning the basics and advanced features of GiD, covering full flow of GiD user from preprocessing to postprocessing going through meshing, analysis and introduction to customization.

Models used in this manual

In order to follow some of the tutorials included in this manual some files should be downloaded.
The models are located in GiD official webpage www.gidsimulation.com in Support->Manuals section.
The models are packed in a zip file and classified by chapters.

INITIATION TO GiD

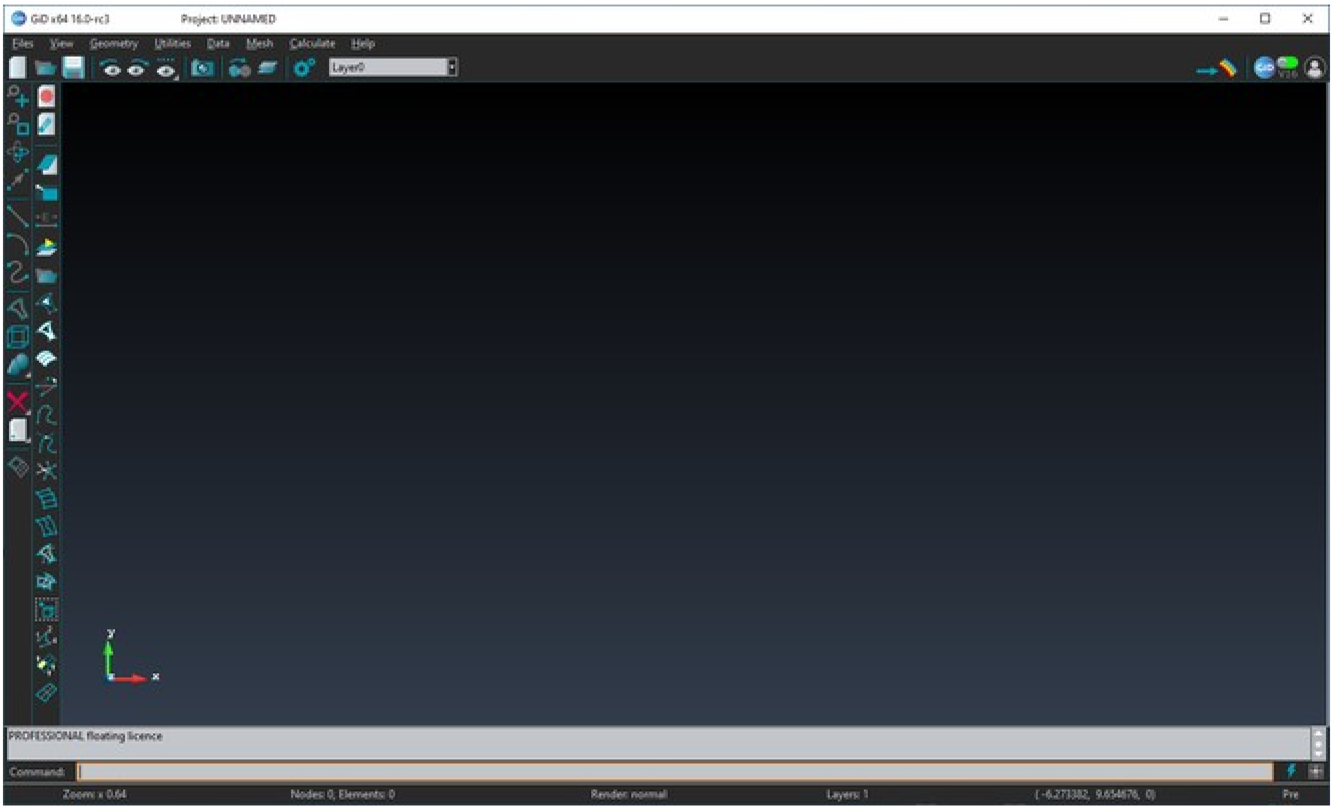
The philosophy of this tutorial is to get familiarized with GiD: how to change the views of the model, how to manage the Layers, and other basic features. Some of these features are both in the preprocessing and the postprocessing parts of GiD, although the examples shown are from the preprocessing one.

Many times, the text will refer to 'entities'. Almost all the options explained in this tutorial are valid both for geometrical and mesh entities, although the examples used are often geometrical ones.

The topics in this tutorial are further explained in the **Reference Manual**. We have selected some of the basic features to give to the user some basic tips to start working with GiD and make the rest of the tutorials.

User interface

For further information about GiD user interface please consult the General aspects->User interface section in the **Reference manual**.



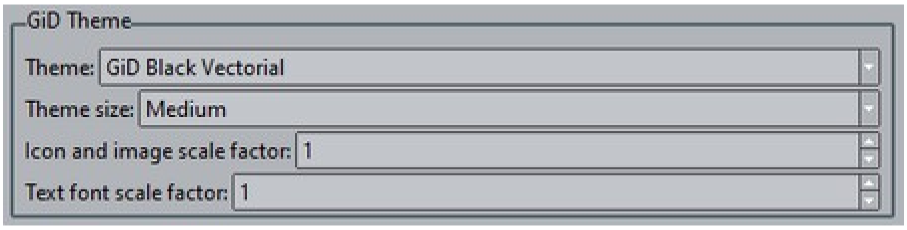
PROFESSIONAL floating licence

Zoom: x 0.64 Nodes: 0, Elements: 0 Render: normal Layers: 1 (-6.273382, 9.654676, 0) Pre

Change theme

User can choose between 'GiD Black Vectorial', 'GiD White Vectorial', 'GiD Classic', or other themes, which change drastically the GUI appearance. User can also choose between some icon and text sizes in each theme.

These options can be changed in **GiD Theme** option inside **Utilities->Preferences->Graphical->Appearance** tab.



GiD Theme

Theme: GiD Black Vectorial

Theme size: Medium

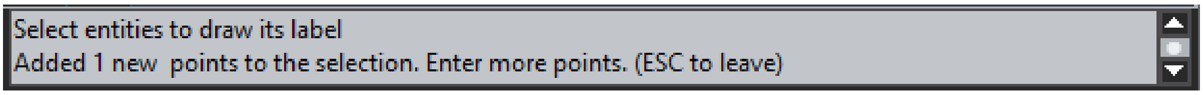
Icon and image scale factor: 1

Text font scale factor: 1

Note: vector-themes have images in SVG format and can be scaled without loss of quality, on the other hand the Classic theme has raster-images (PNG, GIF,...)

Warnline

In some of the operations made in GiD by the user, GiD gives information about what is expected to do by the user. This information is very useful the first times GiD is used as a guideline for the user.
The place where GiD shows this kind of information is the lower part of its main window.



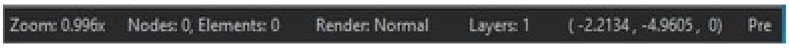
Select entities to draw its label
Added 1 new points to the selection. Enter more points. (ESC to leave)

Command line

Using GiD, sometimes the user is asked to introduce data with the keyboard. The 'Command line' must be used for this purpose. It is placed in the lower part of GiD window.

Status bar

The Status & Information bar located at lower part of GiD's Window, provides basic information at a quick glance.



Zoom: 0.996x Nodes: 0, Elements: 0 Render: Normal Layers: 1 (-2.2134 , -4.9605 , 0) Pre

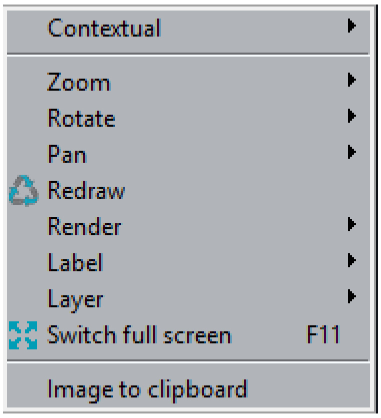
From left to right you can find:

- Zoom factor
- Current number of nodes and elements (Click to access to Status Window)
- Current renter mode (Click to change render)
- Number of layers in Pre, number of sets in Post
- Mouse coordinates (Click to open "Coordinate window" in Pre and "Change result units" in Post)
- Current Mode: Pre or Post

Contextual menu

Clicking the right-mouse button on GiD a popup menu will appear with options related to the clicked object.

When picking the main drawing space, on the top appear **Contextual** that is filled with different commands depending on the current GiD state, e.g. when asking for a point they appear options like "Point in line", to select a point over a line, or "Arc center" to select the coordinates of the center of an arc.



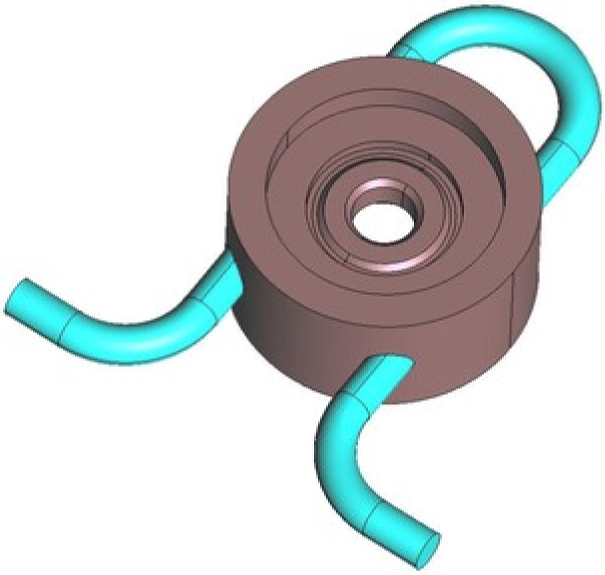
Escape function

An important thing a GiD user should know as a general philosophy of use of the program is the **Escape** key functionality: In almost all the actions performed by the user, to declare the action as done the user should press **Escape** key (or press the center mouse button).

Load a model

In the **Files** menu user can find the typical operations for managing the GiD projects like save a project, open an existing project, import and export files, print or quit the program. Most of this options are also accessible from the icons toolbar. The corresponding icon is shown in the menu, next to the option.

- Click on **Files->Open...** and select the GiD model **gid_model_basic.gid** . GiD also can load a model just with **drag & drop**. The following model should be loaded:

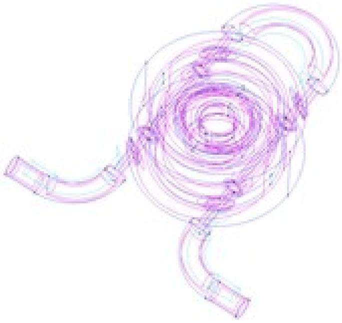
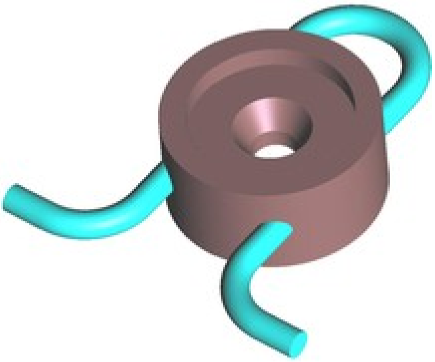
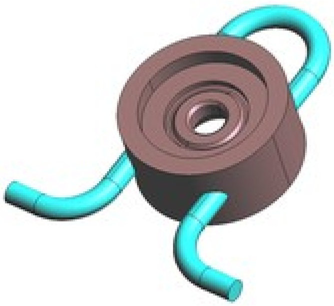


Render modes

In the **View** menu user can find the **Render** options. They are also accessible from the right mouse button and the status bar.

- . Select **View->Render->Normal**
- . Select **View->Render->Flat**
- . Select **View->Render->Smooth**

In **Normal** render mode, user can see the entities drawn in different colors, depending on the kind of entity: volumes in light blue, surfaces in pink, lines in blue, and points in black.
Flat render mode draws each geometrical entity using the color of the layer it belongs to, and **Smooth** mode uses also this criterion, but lines are not drawn to represent the geometry in a smoother way. The following figure shows the visualization of the model changing render modes:

 Render Normal	 Render Smooth	 Render flat

Change views of the model

In the **View** menu user can find the options to change the point of view in which the model is shown. Many of these options are also accessible by the right mouse button menu, or the icons toolbar.

Zoom

To zoom in or out the model user can choose the corresponding options in the **Zoom** section of the **View** menu or the right mouse button menu.
A user friendly way of zooming the model is to use the wheel of the mouse, or clicking the center button of the mouse while the **Shift** key is pressed.
To get a view which includes the whole model the **Frame** option must be selected.

The icons corresponding the zoom operations are the following ones:

Zoom in:

Zoom out:

Zoom frame:

Pan

To move the view of the model user must select the option **Pan**. This option is accessible from the **View** menu, the right mouse button menu, or moving the mouse while the **Shift** key and the right mouse button are pressed.

The corresponding icon for the pan option is the following one:

Rotate

In the 'Rotate' part of the 'View' menu (also present in the right mouse button menu) there are the options to rotate the view of the model.
A user friendly way of rotating is to move the mouse while its left button and the 'Shift' key are pressed.

The corresponding icon for rotating the model is the following one:

Set center of rotation

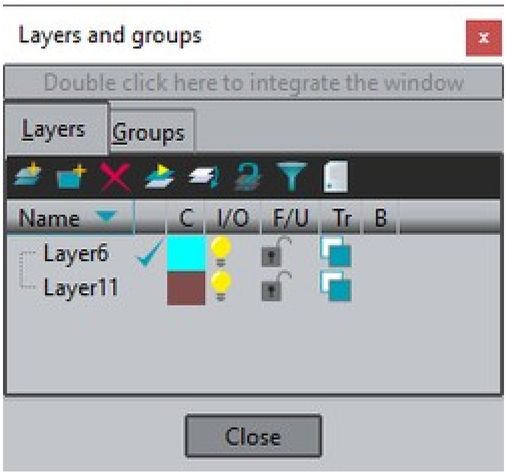
An interesting option for rotating the view of the model is to set the center of rotation. To change it:

- Select **View->Rotate->Center** from top menu or **Rotate->Center** from right button mouse menu. Then, the cursor changes into the selection mode.
- Select an existing point of the model.
- Now rotate the model and check that the center of the rotation is the one selected.

Layers and groups

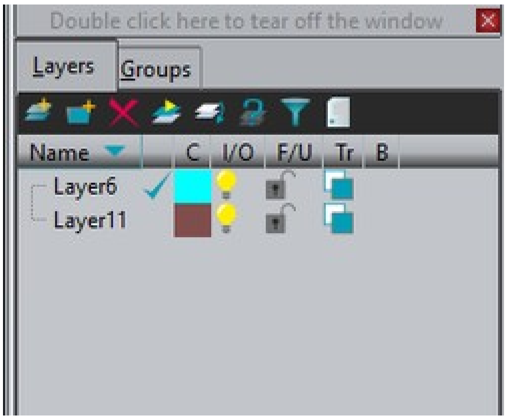
A really useful way for organizing the different parts of the model is using 'Layers'.

- Open the Layers window by selecting the **Utilities->Layer and groups** option or clicking in the upper icons toolbar. The following window should raise up:



As it can be read in the upper part of the window, if user double click on that part, the Layers window is opened as an external window. User can choose to work with the Layers and groups window integrated or not.

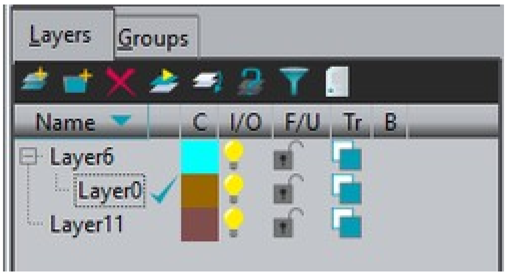
All the actions related with layers and groups can be accessed by clicking the right mouse button onto the Layers and groups window. Most of them can be also used by the corresponding icon in the upper part of the Layers window.
By moving the mouse over the icons of the upper part of the window and staying 2 seconds onto an icon, a help message is shown in order to give the user information about the action associated with the icon.



Create a layer

GiD allows to create a hierarchical structure of Layers, so as a Layer can contain sub-layers. Let's create a Layer into another one as an example:

- Select (using the left button of the mouse) the 'Layer6'.
- Select the **New child** option in the right mouse button menu, or click in the upper part of the Layers and groups window. Automatically, a layer named 'Layer0' should appear, as shown in the following figure:

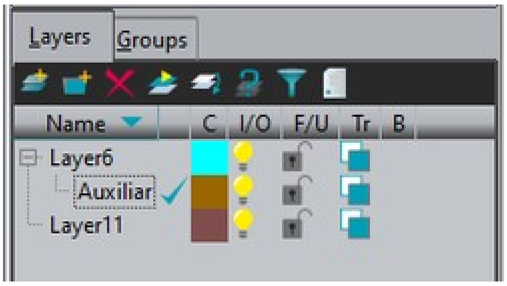


Rename a layer

To rename a Layer user should select the layer in the Layers and groups window and press **F2** key, or select the **Rename** option in the right mouse button menu.

- Select the Layer0
- Rename it to 'Auxiliar'

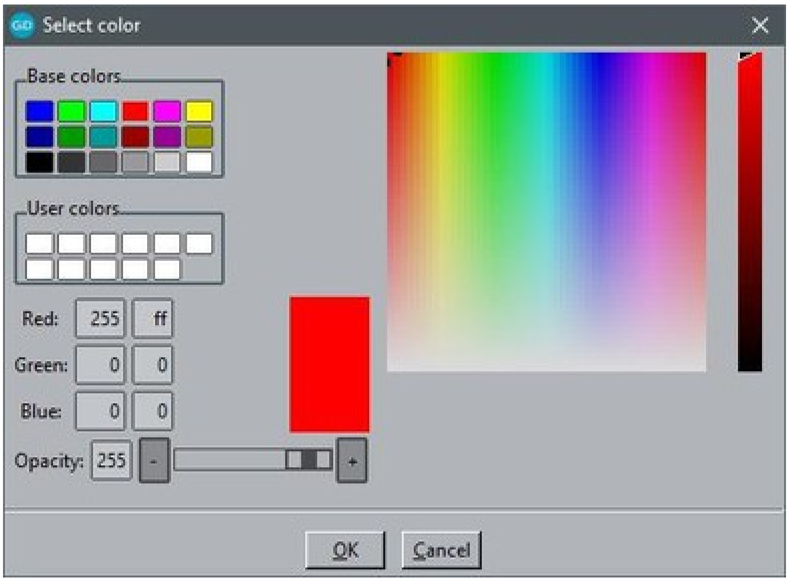
Now the Layers window should look like the following picture:



Change the color of a layer

By clicking on the colored square next to each layer name, the following window pops-up, allowing the user to change the color of the layer:

Set the color of the layer 'Auxiliar' to red (RGB=255,0,0)

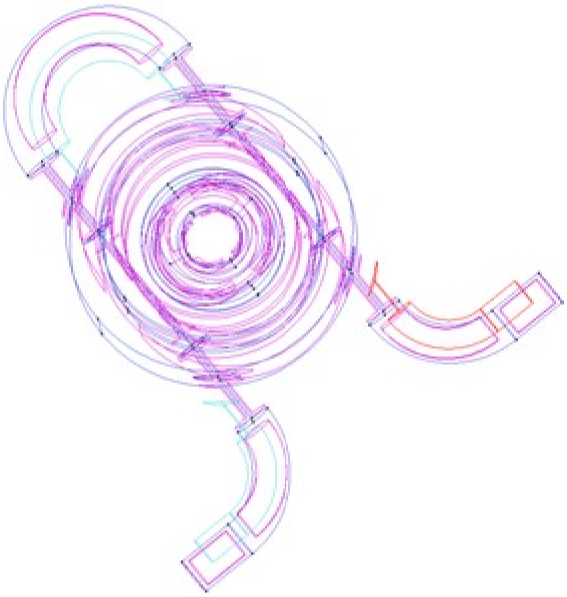


Red: 255 ff
Green: 0 0
Blue: 0 0
Opacity: 255

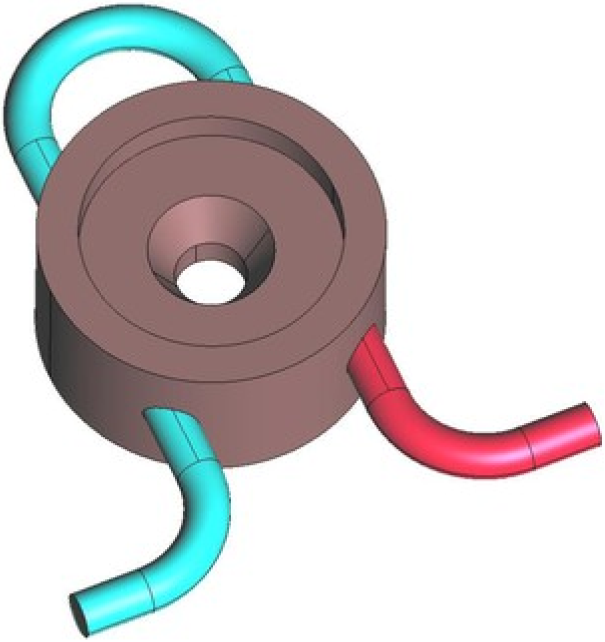
Send entities to a layer

User can send entities to a specific layer. As an example we are going to send to the layer 'Auxiliar' a part of the model:

- Select the layer 'Auxiliar' in the Layers window
- Select the option **Send to** from the right mouse button (or click icon)
- Select **Volumes** and select the volume shown in red in the following figure:



- Then press **Escape** to exit the selection mode.
- Set the render mode to **Flat**. The color of selected volume has changed to the one of the layer 'Auxiliar', as shown in the following figure:



Switch On,Off

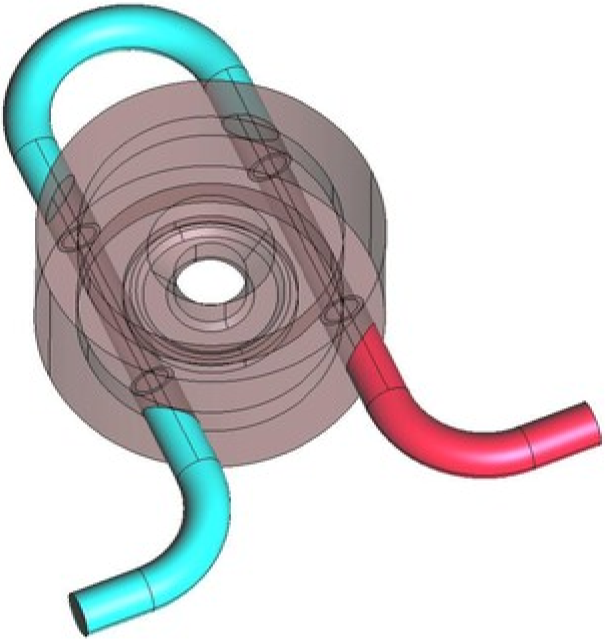
By clicking on the icon which is next to each Layer inside the Layers and groups window, user can switch on and off the corresponding layer. This is very useful in order to visualize just some specific parts of the model.

Freeze a layer

At the right side of the bulb, user can set an icon which is a lock 🔓. If the lock is closed 🔒, the layer is frozen. If a layer is frozen, GiD won't apply anything to the entities of that layer, and its entities will be drawn in light gray color. For instance, if user select some entities to be deleted, if they are into a frozen layer they won't be erased.

Transparency

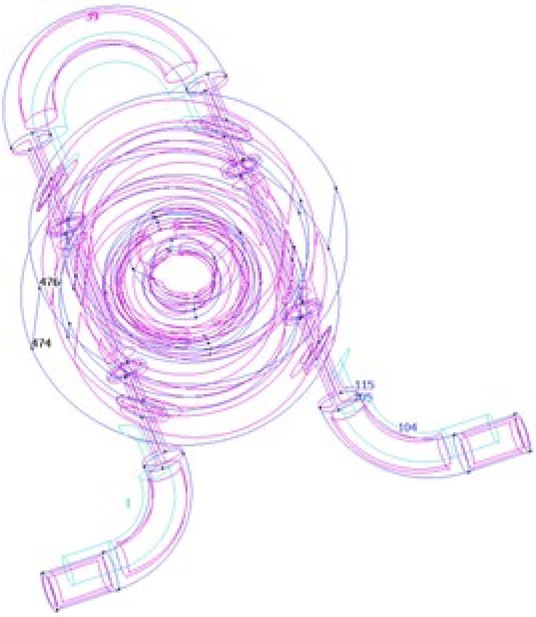
Next to the 'lock' icon of each layer is the transparency icon ❐. By clicking there, the user can set a layer to be transparent or not. The following figure shows the model with the Layer11 set as transparent:



Entities information

Labels

Using the option **Labels** present in the **View** menu (and also in the right mouse button menu), user can see the numeric id of some entities of the model. Either for points, lines, surfaces or volumes user can choose between viewing the labels of all the entities, or just the selected ones.
In the following figure the model can be seen with the number of some entities:



As it can be seen, the colors of the numbers of the entities follows the philosophy of the colors of the entities in

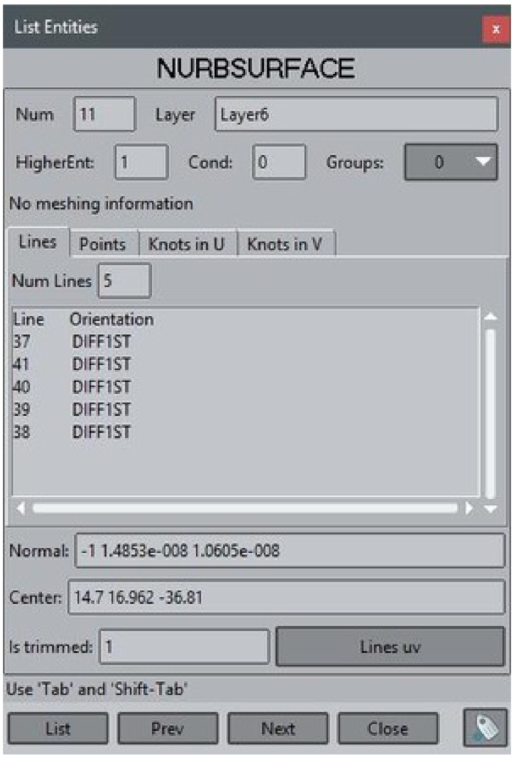
GiD (volume numbers are in light blue, surface numbers are in pink and so on).
In order to get a better visualization set the render mode to normal when showing labels.

List entities

User has also the option of viewing all the characteristics of a specific entity by selecting **List** in the **Utilities** menu (or clicking in the icons toolbar).
For example:

- Select **Utilities->List->Surfaces** in the top menu
- Select some surfaces of the model
- Press **Escape**

An example of the information got using this option is the following figure:



List Entities

NURBSURFACE

Num 11 Layer Layer6

HigherEnt: 1 Cond: 0 Groups: 0

No meshing information

Num Lines 5

Line	Orientation
37	DIFF1ST
41	DIFF1ST
40	DIFF1ST
39	DIFF1ST
38	DIFF1ST

Normal: -1 1.4853e-008 1.0605e-008

Center: 14.7 16.962 -36.81

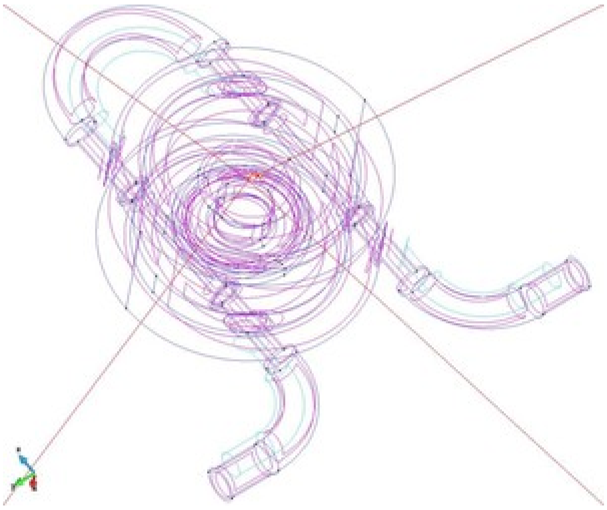
Is trimmed: 1

Use 'Tab' and 'Shift-Tab'

Signal

In complex geometrical models sometimes it is hard to localize a specific entity. Using the **Signal** option in the **Utilities** menu user can know graphically where the entity is, as GiD shows with a red lines cross its position. As an example we will signal the line number 290:

- Select **Utilities->Signal->Lines**
- Write in the Command bar the number 290 and click <Return>. The result is shown in the next figure:

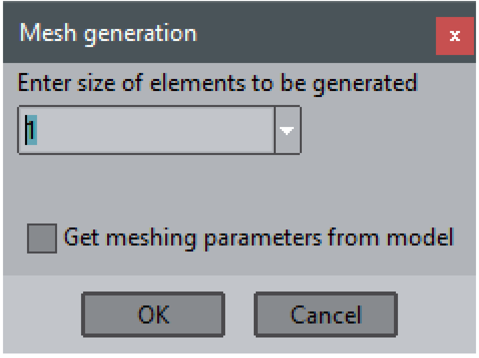


The red lines are centered always onto the specific entity independently on the rotations or view movements.

Geometry and Mesh modes

In the preprocessing part of GiD there are two basic modes the user can work with: geometry and mesh. Just in order to see how the mode can be changed we are going to generate a mesh with all the default parameters and general edge size=1.0.

- Select **Mesh->Generate mesh...** The following window should appear.

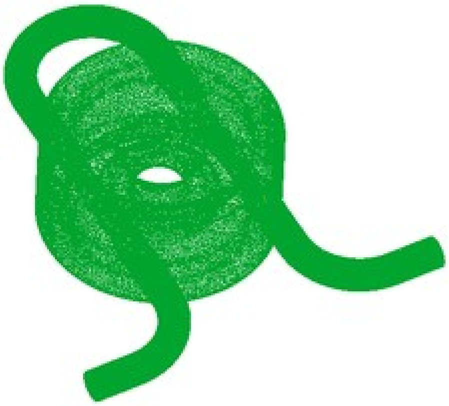

Mesh generation

Enter size of elements to be generated

1

□ Get meshing parameters from model

- Click OK and wait for the mesh generation. Once the mesh is generated, a window pops up and shows the user the result from the mesh generation.
- Click on 'View mesh' option, and the following visualization of the model should appear:



The color of the mesh in render normal could be set in Preferences Graphical - Colours,

e.g. can set the elements color to gray or other color.

Now we are in 'mesh' mode. Changing the render mode user can see that the color of the mesh entities also follows the Layer colors.

Selecting **View->Mode->Geometry** user can change to the geometry mode again. The icon in the toolbar switch between both modes

Pre and Post

GiD basically works in two modes: preprocessing and postprocessing.

To change between both modes please select **Files->Postprocess** or **Files->Preprocess** (or clicking in the upper toolbar).
We will use a different model to work in postprocess mode.

- Open the **box3D.gid** project
- Select **Files->Postprocess**

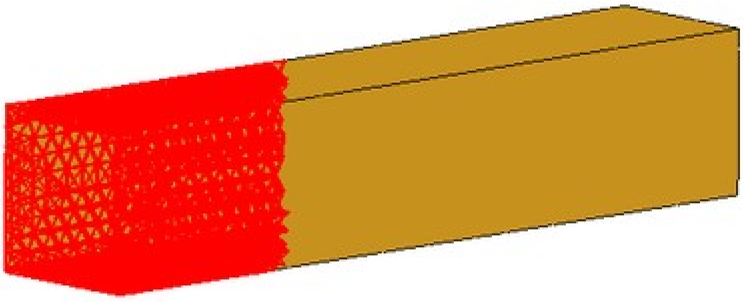
Select and display style

Through the **Select & Display Style** window several options can be specified for volumes, surfaces and cuts. Among these options volumes, surfaces and cuts can be switched on and off, their colour properties can be changed, and their transparency too.
Other interesting options which can be changed are the style of the set and the width of the elements' edges. From this window, volumes, surfaces or cuts can be deleted or their names can be modified.

- Select **Window->View style...** using the menu bar or clicking on

Our model only has 1 layer. We will create a new layer with some elements (in fact in postprocess instead of 'layers' are 'sets', but are similar concepts)

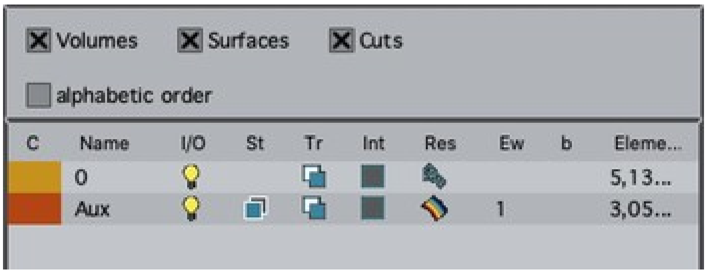
- Press button **Send to->New set**.
- Select some elements.



- Press **Escape**.
- A window appears asking for a name. Enter 'Aux'.
- Press **Accept**.

A new layer is created with the selected elements. Now we will change the color of the new layer.

- Click on the colored square next to the layer name. A new window is opened. Select a new color.
- Press **Apply** and then **Close**.



☒ Volumes ☒ Surfaces ☒ Cuts

□ alphabetic order

C	Name	I/O	St	Tr	Int	Res	Ew	b	Eleme...
🟧	0	💡		▣	■	▨			5,13...
🟥	Aux	💡	▣	▣	■	▨	1		3,05...

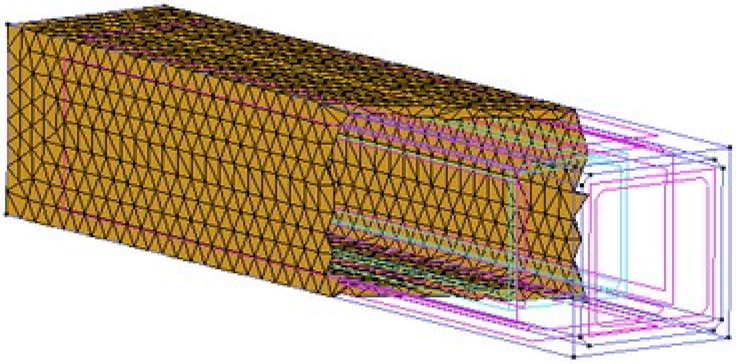
Let's play with some visualization options

- Select **Body Lines** in the "Style" option, at the bottom of the window. You can also do it clicking on in the St column or the same icon in the main window, but then it only affects this set.
- Click on the 💡 icon of 'Aux' layer in order to switch it off.

It's possible to draw preprocessing information, for example the geometry.

- In **Draw Model** option select **Geometry**

Now our model should look like this:

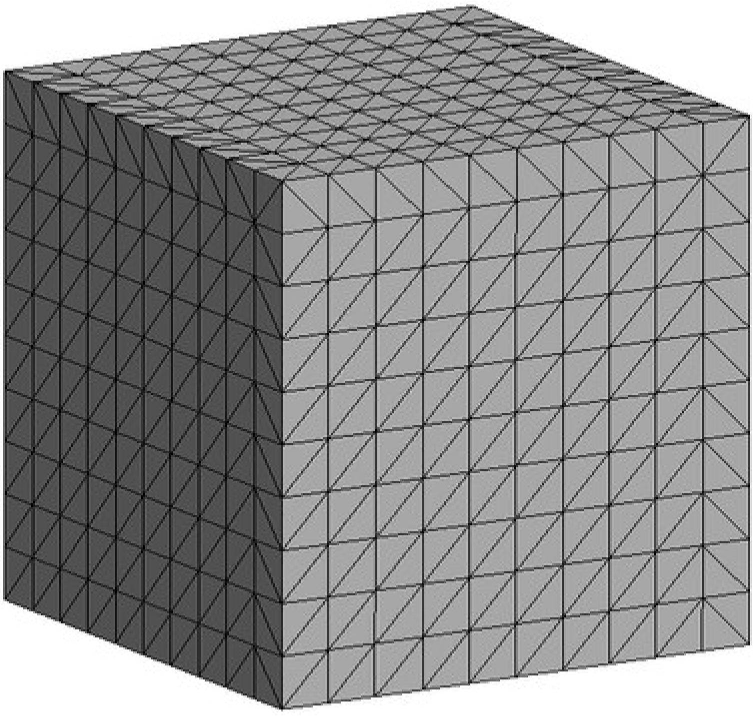


NOTE: The View style window can be integrated inside GiD interface, just double click on the upper bar of the window. To tear it off again, double click the upper bar again.

NOTE: Mesh styles can also be changed clicking on the icon ☒ , placed in the left icon bar. This style affects all sets of the model.

INITIATION TO PREPROCESSING

With this example, the user is introduced to the basic tools for the creation of geometric entities and mesh generation.



First steps

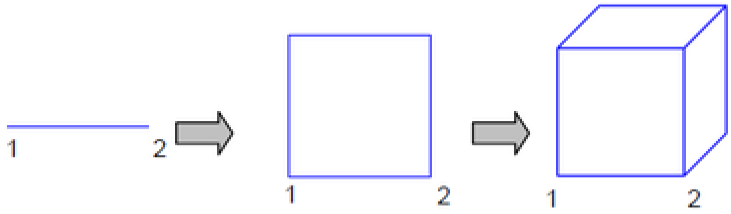
Before presenting all the possibilities that **GiD** offers, we will present a simple example that will introduce and familiarize the user with the **GiD** program.

The example will develop a finite element problem in one of its principal phases, the preprocess, and will include the consequent data and parameter description of the problem. This example introduces creation, manipulation and meshing of the geometrical entities used in **GiD**.

First, we will create a line and the mesh corresponding to the line. Next, we will save the project and it will be described in the **GiD** database form. Starting from this line, we will create a square surface, which will be meshed to obtain a surface mesh. Finally, we will use this surface to create a cubic volume, from which a volume mesh can then be generated.

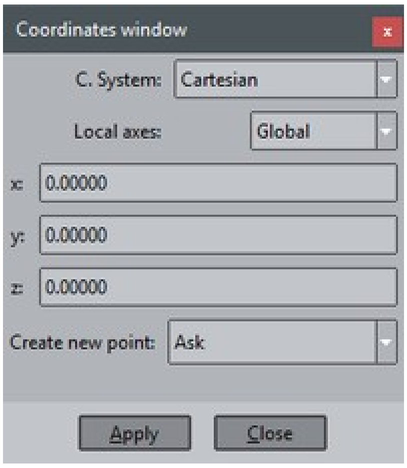
Creation and meshing of a line

We will begin the example creating a line by defining its origin and end points, points 1 and 2 in the following figure, whose coordinates are (0,0,0) and (10,0,0) respectively.
It is important to note that in creating and working with geometric entities, **GiD** follows the following hierarchical order: point, line, surface, and volume.



To begin working with the program, open **GiD**, and a new **GiD** project is created automatically.

From this new database, we will first generate points 1 and 2.
Next, we will create points 1 and 2. To do this, we will use an **Auxiliary Window** that will allow us to simply describe the points by entering coordinates. It is accessed by the following sequence: **Utilities->Tools->Coordinates window**
Then, from the Top Menu, select **Geometry->Create->Point**
In the coordinate window opened previously, enter the coordinates of point 1 in the "x", "y", "z" entries and click **Apply** or press **Enter** on the keyboard.


Coordinates window

C. System: Cartesian

Local axes: Global

x: 0.00000

y: 0.00000

z: 0.00000

Create new point: Ask

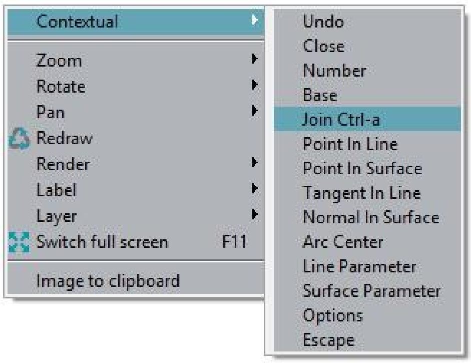
And create point 2 in the same way, introducing its coordinates in the **Coordinates window**.

The last step in the creation of the points, as well as any other command, is to press **Escape**, either via the **Escape** button on the keyboard or by pressing the central mouse button. Select **Close** to close the **Coordinates window**.

In order to view everything that has been created to this point, center the image on the screen by choosing in the **Mouse Menu: Zoom->Frame**.
Now, we will create the line that joins the two points. Choose from the **Top Menu**: **Geometry->Create->Straight line.** Option in the **Toolbar** shown below can also be used.

Next, the origin point of the line must be defined. In the **Mouse Menu**, opened by clicking the right mouse button, select **Contextual->Join Ctrl-a.**

NOTE: It is important to note that the Contextual submenu in the Mouse Menu will always offer the options of the command that is currently being used. In this case, the corresponding submenu for line creation, has the following options:



NOTE: With option Join, a point already created can be selected on the screen. The command No Join is used to create a new point that has the coordinates of the point that is selected on the screen. We can see that the cursor changes form for the Join and No Join commands.

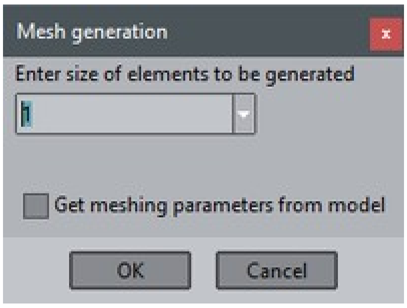
⊡ Cursor during use of **Join** command

✛ Cursor during use of No **Join** command

Now, choose on the screen the first point, and then the second, which define the line. Finally, press **Escape** to indicate that the creation of the line is completed. Press **Escape** again to end the line creation function, if you don't press Escape you can continue creating lines.

Once the geometry has been created, we can proceed to the line meshing. In this example, this operation will be presented in the simplest and most automatic way that **GiD** permits. To do this, from the **Top Menu** select:
Mesh->Generate mesh (or <Ctrl>-g keys)
And an **Auxiliary Window** appears, in which the size of the elements should be defined by the user.

NOTE: The size of an element with two nodes is the length of the element. For, surfaces or volumes, the size is the mean length of the edge of the element.
In this example, the size of the element is defined in concordance with the length of the line, chosen for this case as size 1. Click **OK**.

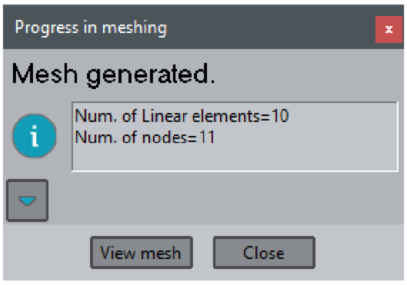

Mesh generation

Enter size of elements to be generated

1

□ Get meshing parameters from model

Once the mesh has been generated a window with the mesh information appears. Click **View mesh**.



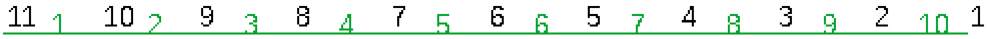
Progress in meshing

Mesh generated.

Num. of Linear elements=10
Num. of nodes=11

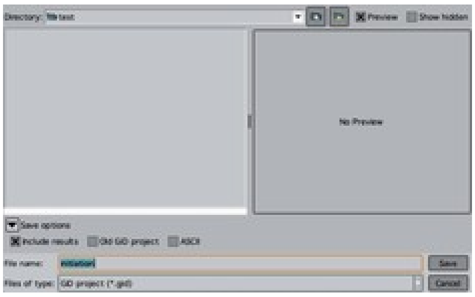
Automatically **GiD** generates a mesh for the line. Pressing 'View mesh' the finite element mesh is presented on the screen.

The mesh is formed by ten linear elements of two nodes. To see the numbering of the nodes and mesh elements, select from the **Mouse Menu: Label->All on,** and the numbering for the 10 elements and 11 nodes will be shown, as below.



Once the mesh has been generated, the project should be saved. To save the example select from the **Top Menu: Files->Save.**

The program automatically saves the file if it already has a name. If it is the first time the file has been saved, the user is asked to assign a name. For this, an **Auxiliary Window** will appear which permits the user to browse the computer disk drive and select the location in which to save the file. Once the desired directory has been selected, the name for the current project can be entered in the space titled **File Name**. Save it as **initiation.gid**.



Directory: test ☒ Preview ☐ Show hidden

Save options
☒ Include results ☐ Old GiD project ☐ ASCII

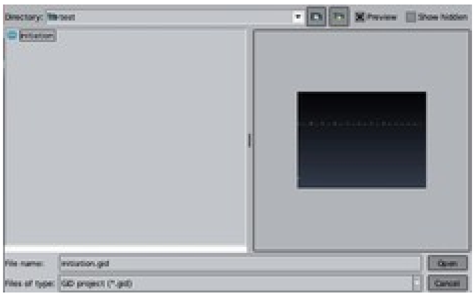
File name: initiation

Files of type: GiD project (*.gid)

NOTE: Next, the manner in which **GiD** saves the information of a project will be explained. **GiD** creates a directory with a name chosen by the user, and whose file extension is **.gid**. **GiD** creates a set of files in this directory where all the information generated in the present example is saved. All the files have the same name of the directory to which they belong, but with different extensions. These files should have the name that **GiD** designates and should not be changed manually.
Each time the user selects option **save** the database will be rewritten with the new information or changes made to the project, always maintaining the same name.

To exit **GiD**, simply choose **Files->Quit.**

To access the project that we have just created, simply start **GiD** again and select from the **Top Menu: Files->Open.** An **Auxiliary Window** will appear which allows the user to access and open the directory **initiation.gid** .



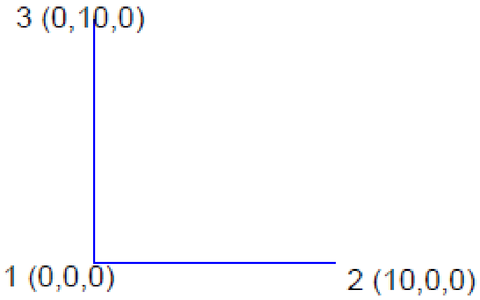
Directory: test ☒ Preview ☐ Show hidden

File name: initiation.gid

Files of type: GiD project (*.gid)

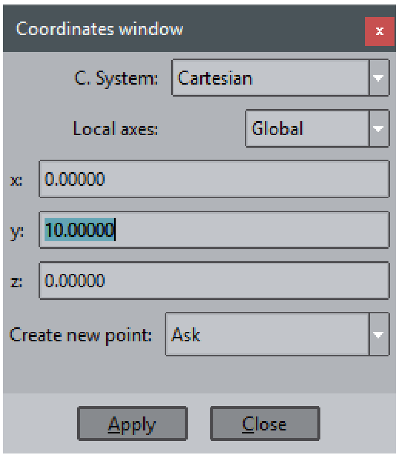
Creation and meshing of a surface

We will now continue with the creation and meshing of a surface.
First, we will create a second line between points 1 and 3.



We will now generate the second line. We will now use again the **Coordinates Window** to enter the points. (**Utilities->Tools->Coordinates Window**)
Select the line creation tool in the toolbar.

Enter point (0,10,0) in the **Coordinates Window** and click **Apply**.


Coordinates window

C. System: Cartesian

Local axes: Global

x: 0.00000

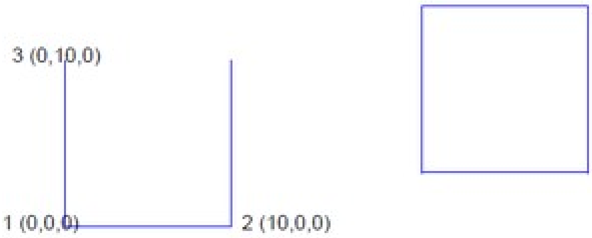
y: 10.00000

z: 0.00000

Create new point: Ask

With option **Contextual->Join Ctrl-a** (mouse menu) click over point 1. A line should be created between (0,10,0) and (0,0,0). Press **Escape** twice.
With this, a right angle of the square has been defined.
Center the image in the screen with **View->Zoom->Frame**.

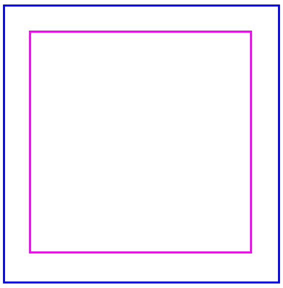
Finish the square by creating point (10,10,0) and the lines that join this point with points 2 and 3.



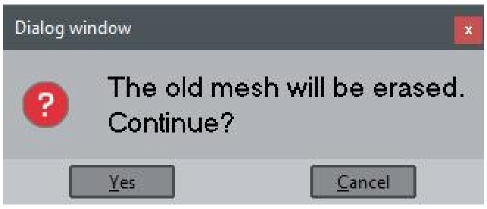
Now, we will create the surface that these four lines define. To do this, access the create surface command by choosing: **Geometry->Create->NURBS surface->By contour**. This option is also available in the toolbar:

GiD then asks the user to define the 4 lines that describe the contour of the surface. Select the lines using the cursor on the screen, either by choosing them one by one or selecting them all with a window. Next, press **Escape** twice.

As can be seen below, the new surface is created and appears as a smaller, magenta-coloured square drawninside the original four lines.



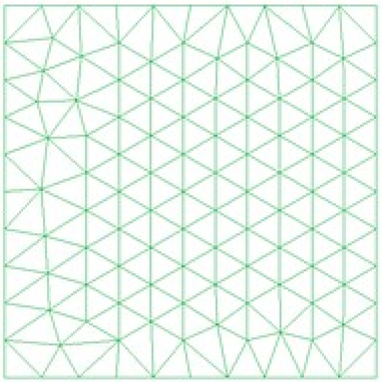
Once the surface has been created, the mesh can be created in the same way as was done for the line. From the **Top Menu** select: **Mesh->Generate mesh.**
A window appears asking if the previous mesh should be eliminated. Click **Yes**.



Dialog window

The old mesh will be erased. Continue?

Another window appears which asks for the maximum size of the element, in this example defined as 1.

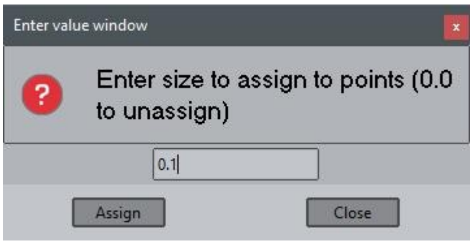


We can see that the lines containing elements of two nodes have not been meshed. Rather the mesh generated over the surface consists of planes of three-nodded, triangular elements.

NOTE: GiD meshes by default the entity of highest order with which it is working.

GiD allows the user to concentrate elements in specified geometry zones. Next, a brief example will be presented in which the elements are concentrated in the top right corner of the square.

This operation is realized by assigning a smaller element size to the point in this zone than for the rest of the mesh. Select the following sequence: **Mesh->Unstructured->Assign sizes on points.** The following dialog box appears, in which the user can define the size:

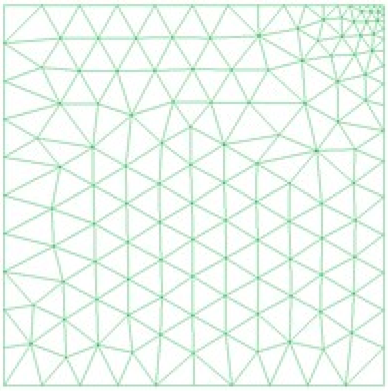


Enter value window

Enter size to assign to points (0.0 to unassign)

0.1

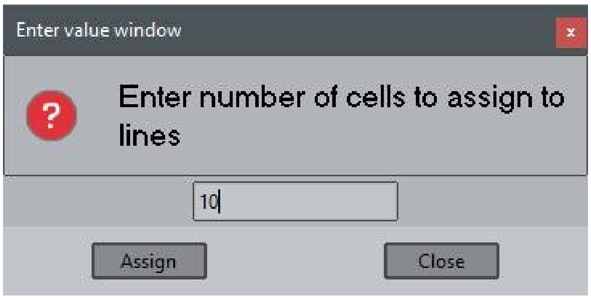
Enter **0.1** and click **Assign.**
Select the upper-right corner point and press **Escape.** The same window comes up again, click **Close.**
We must now regenerate the mesh, erasing the mesh generated earlier, and we obtain the following:



As can be seen in the figure above, the elements are concentrated around the chosen point. Various possibilities exist for controlling the evolution of the element size, which will be presented later in the manual.

To generate a surface mesh in which the elements are presented uniformly, the user can select the option for a structured mesh. This guarantees that the same number of elements appears around a node and that the element size is as uniform as possible. To generate this type of mesh, choose: **Mesh->Structured->Surfaces->Assign number of divisions to surface lines.**

Using this command, the user should first select the **4-sided NURBS surface** that will be defined by the mesh and press **ESC.**
Then a window appears where the number of subdivisions for the surface limit lines should be entered.

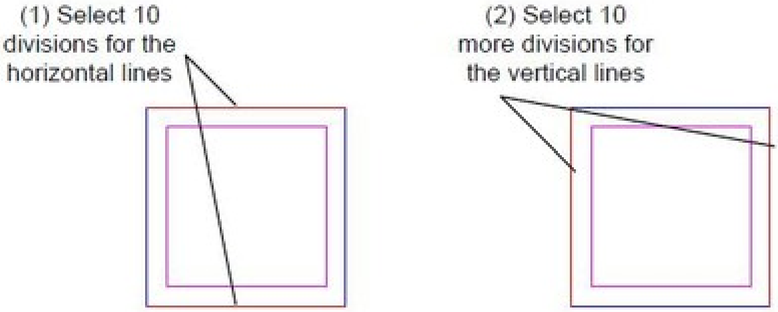


Enter value window

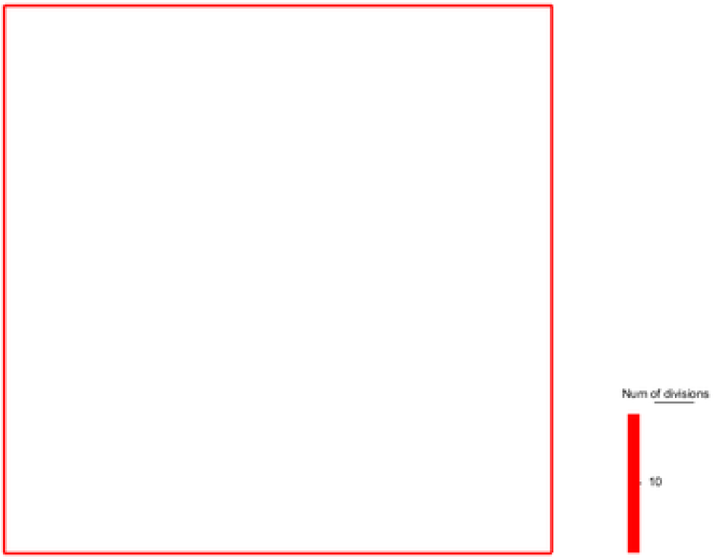
Enter number of cells to assign to lines

10

Enter **10** and click **Assign** and select one of the **horizontal** lines, the parallel line is also selected. Press **ESC**.
The same window appears again, click **Assign** and select one of the **vertical** lines, the parallel one is also selected. Press **ESC**.
Click **Close** when the window appears again.

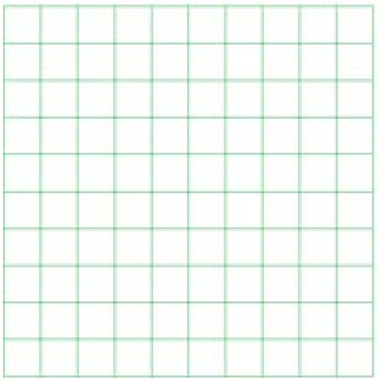

(1) Select 10 divisions for the horizontal lines
(2) Select 10 more divisions for the vertical lines

The number of divisions can be checked selecting **Mesh->Draw->Num of divisions**. To exit this visualization mode press **ESC.**



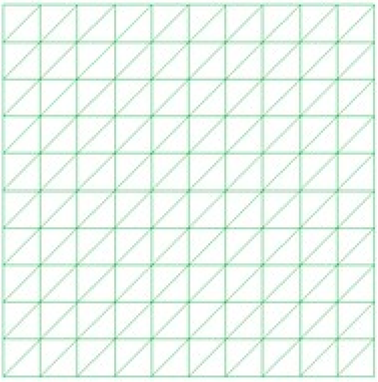
NOTE: GiD only generates structured meshes for surfaces of the type **4-sided surface** or **NURBS surface**.

When this has been done, the mesh is generated in the same way as the unstructured mesh, by choosing **Mesh->Generate mesh.** Erase the old mesh and assign a general element size of 1, though in this case it is not necessary.



NOTE: Another way to get the same result is using the option **Mesh->Structured->Surfaces->Assign size to surface lines**. With this option we set the element size. If we want to get 10 elements per line and the line measures 10 units, we should set 1 as size.
If we don't know how much measures a line we can use the option **Utilities->Distance** and select the 2 points defining the line.

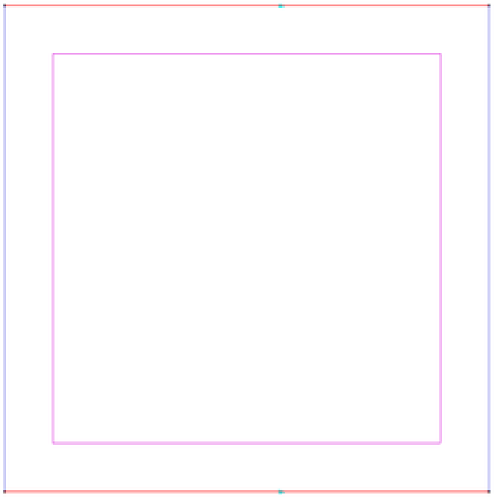
We can see here that the default element type used by **GiD** to create a structured mesh is a square element of four nodes rather than a three-nodded, triangular element. To obtain triangular elements, the user can specifically define this type of element, by choosing **Mesh->Element type->Triangle,** and selecting the surface to mesh as a triangular element. Regenerate the mesh, and the following figure is obtained:

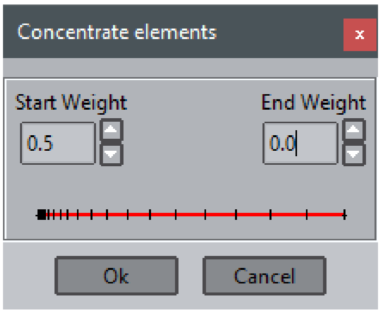


GiD also allows the user to concentrate elements in structured meshes. This can be done by selecting **Mesh->Structured->Lines->Concentrate elements.**

First, we must select the lines that need to be assigned an element concentration weight. The value of this weight can be either positive or negative, depending on whether the user wants to concentrate elements at the beginning or end of the lines. Next, a vector appears which defines the start and end of the line and which helps the user assign the weight correctly.

We want to concentrate the elements in the left zone of the square.
Select both **horizontal** lines and press **ESC**. A window appears to enter the weights values.
Both lines should have the same direction so enter a weight of 0.5 to the beginning of the line and click **Ok**.
Press **ESC** again to leave the function.





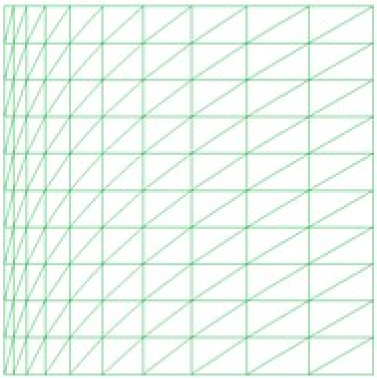
Concentrate elements

Start Weight: 0.5

End Weight: 0.0

If lines have different direction, to obtain the same result, we should assign the weight for one line to the beginning and to the end for the other line.

From these operations, we obtain the following mesh:



Creation and meshing of a volume

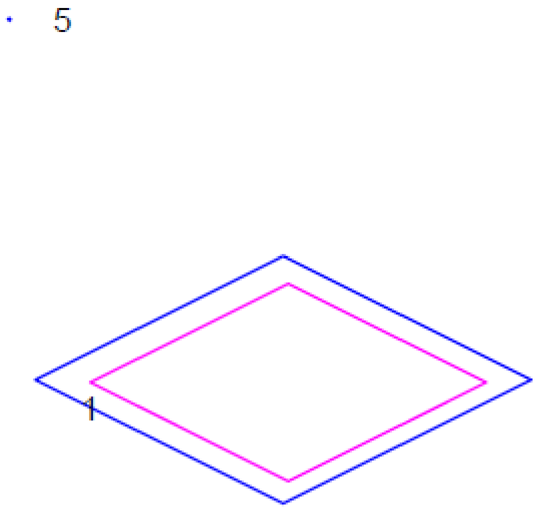
We will now present a study of entities of volume. To illustrate this, a cube and a volume mesh will be generated.

Without leaving the project, save the work done up to now by choosing **Files->Save,** and return to the geometry last created by choosing **View->ModeGeometry.**

In order to create a volume from the existing geometry, firstly we must create a point that will define the height of the cube. This will be point 5 with coordinates (0,0,10), superimposed on point 1. To view the new point, we must rotate the figure by selecting from the **Mouse Menu: RotateTrackball.**

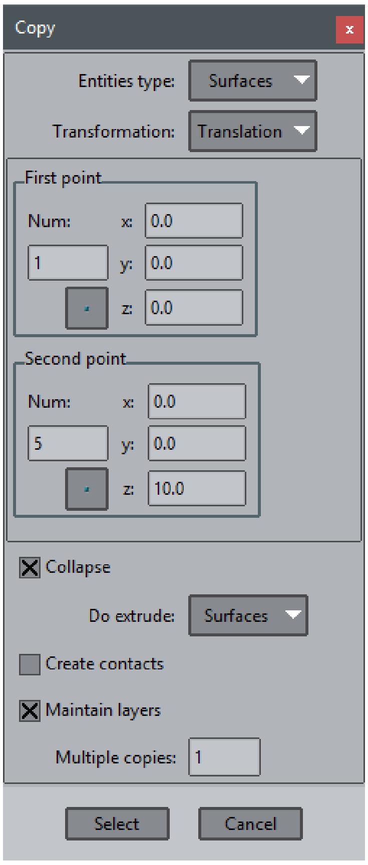
This option is also available in the toolbar:

Rotate the figure until the following position is achieved and press **ESC**:



Next, we will create the upper face of the cube by copying from point 1 to point 5 the surface created previously. To do this, select the copy command, **UtilitiesCopy.**

In the **Copy** window, we define the translation vector with the first and second points, in this case (0,0,0) and (0,0,10). Option **Do extrude surfaces** must be selected; this option allows us to create the lateral surfaces of the cube. Fill in the rest of variables as shown in the following image.



Copy

Entities type: Surfaces

Transformation: Translation

First point

Num: 1

x: 0.0

y: 0.0

z: 0.0

Second point

Num: 5

x: 0.0

y: 0.0

z: 10.0

☒ Collapse

Do extrude: Surfaces

□ Create contacts

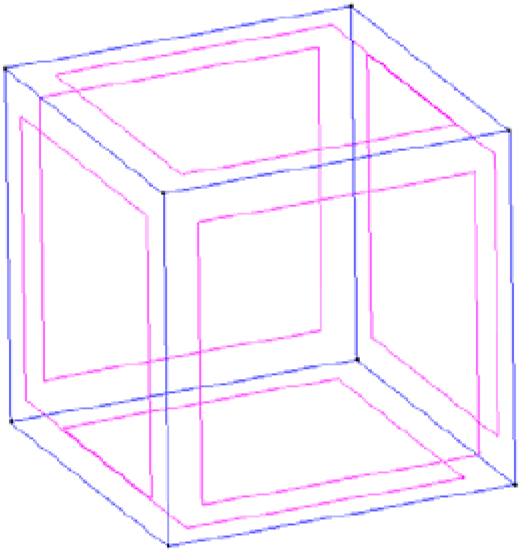
☒ Maintain layers

Multiple copies: 1

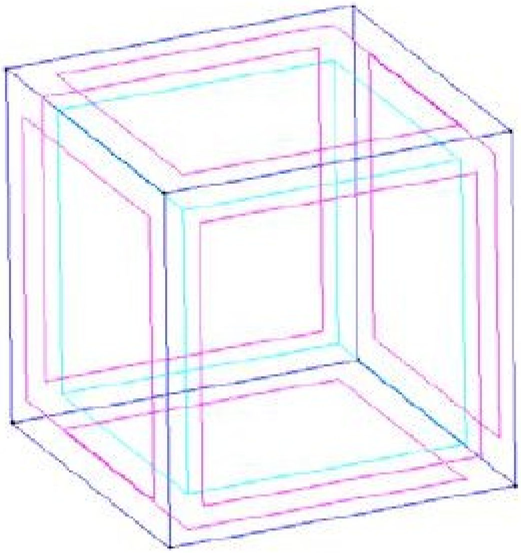
NOTE: In the **Copy** and **Move** windows, the button may be used to select existing points with the mouse, or alternatively enter its number in the entry field.

NOTE: If we look at the **Copy Window,** we can see an option called **Collapse**. By activating this option, point 6 will be merged with point 5 when the entities are copied. By labeling the entities we could verify that only one point has been created.

If the user does not choose option **Collapse**, when the entities are copied (in this case from point 1 to point 5) **GiD** would create a new point (point 6) with the same coordinates as point 5.
To finish the copy command click **Select**, select the surface and then press **ESC**. We obtain the following surfaces:

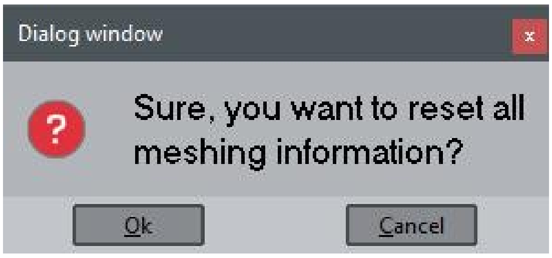


Now, we can generate the volume delimited by these surfaces. To create the volume, simply select the command **Geometry->Create->Volume->By contour** . This option is also available in the toolbar:



Select all the surfaces and press **ESC twice**. **GiD** automatically generates the volume of the cube. The volume viewed on the screen is represented by a cube with an interior color of sky blue.

Before proceeding with the mesh generation of the volume, we should eliminate the information of the structured mesh created previously for the surface. Do this by selecting **Mesh->Reset mesh data,** and the following dialog box will appear on the screen:

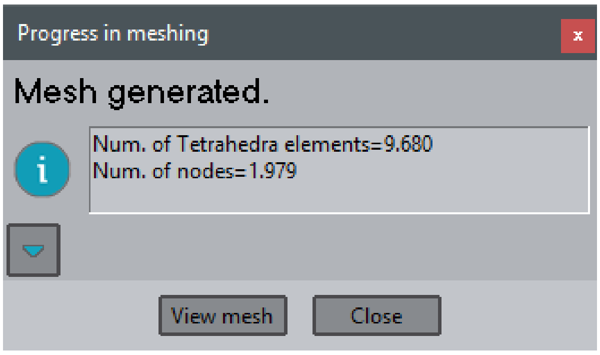


Dialog window

Sure, you want to reset all meshing information?

In which the user is asked to confirm the erasure of the mesh information.

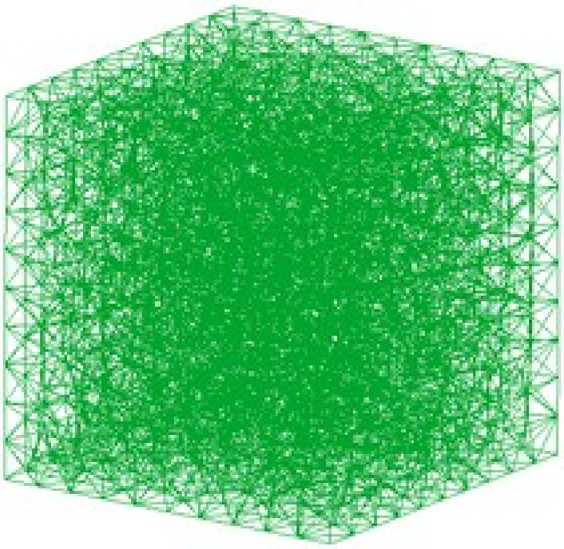
NOTE: Another valid option would be to assign a size of 0 to all entities. This would eliminate all the previous size information as well as the information for the mesh, and the default options would become active. Next, generate the mesh of the volume by choosing **Mesh->Generate mesh.** Another **Auxiliary Window** appears into which the size of the volumetric element must be entered. In this example, the value is 1.



Progress in meshing

Mesh generated.

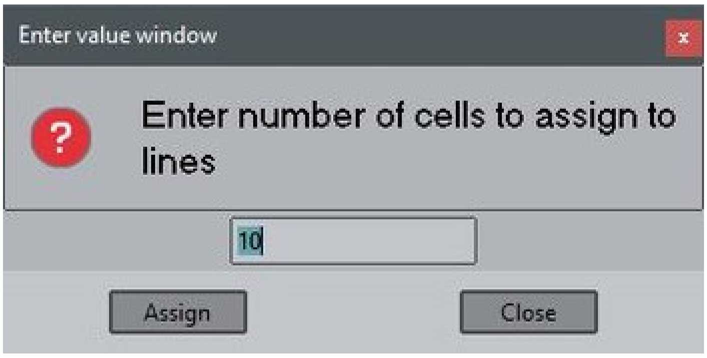
```
Num. of Tetrahedra elements=9.680
Num. of nodes=1.979
```



The mesh generated above is composed of tetrahedral elements offour nodes, but **GiD** also permits the use of hexahedral, eight-nodded structured elements.
We will generate a structured mesh of the volume of the cube. This is done by selecting **Mesh->Structured->Volumes->Assign number of divisions to volume lines.**

Now select the volume to mesh and press **ESC**.
Then a window appears where the number of subdivisions for the volume limit lines should be entered.

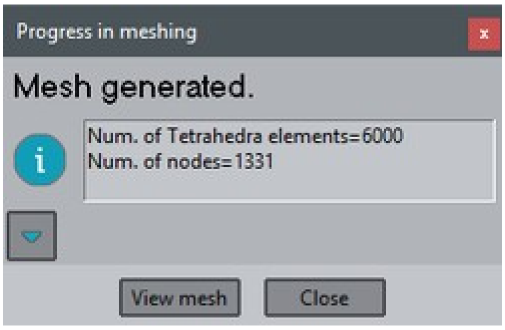
Enter **10** and click **Assign** and select one of the lines in **X** axis, the parallel lines are also selected. Press **ESC**.
The same window appears again, click **Assign** and select one of the lines in **Y** axis, the parallel ones are also selected. Press **ESC**.
Again click **Assign** and select one of the lines in **Z** axis, the parallel ones are also selected. Press **ESC**.
Click **Close** when the window appears again.
Then, create again the mesh.



Enter value window

Enter number of cells to assign to lines

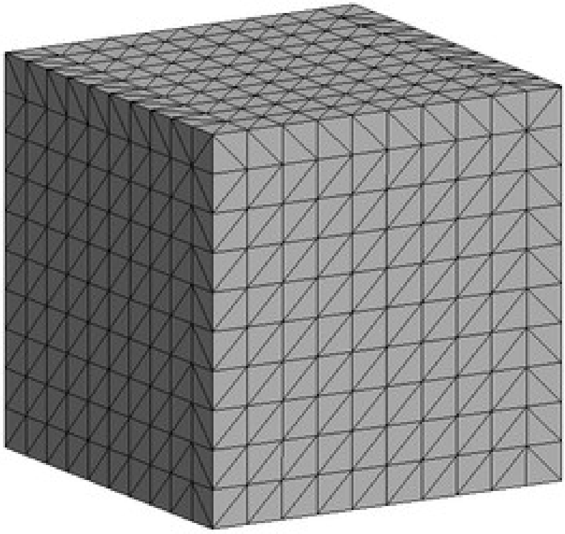
10



Progress in meshing

Mesh generated.

Num. of Tetrahedra elements=6000
Num. of nodes=1331



NOTE: GiD only allows the generation of structured meshes of 6-sided volumes.

With this example, the user has been introduced to the basic tools for the creation of geometric entities and mesh generation.

IMPLEMENTING A MECHANICAL PART

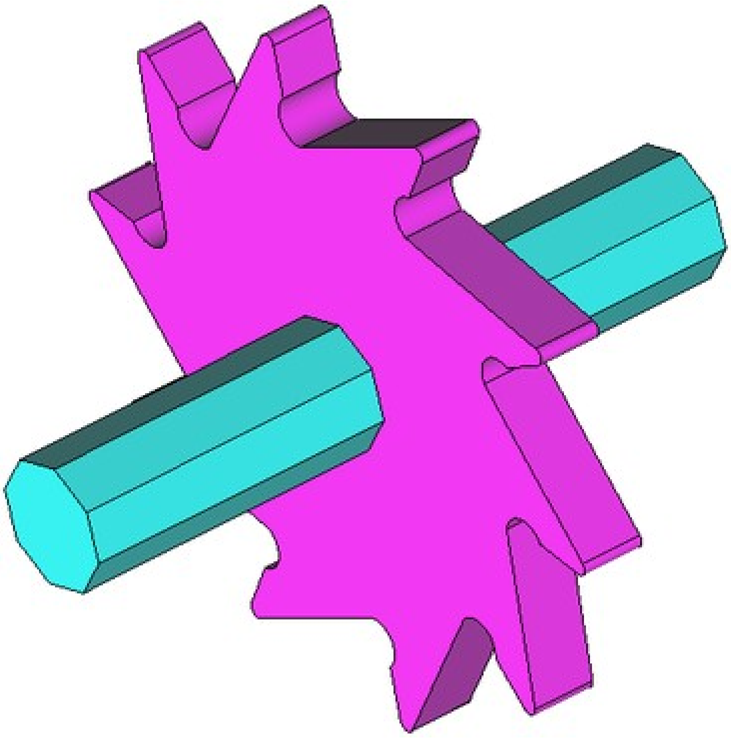
2D TOOLS, BASIC 3D TOOLS AND MESHING

IMPLEMENTING A MECHANICAL PART

The objective of this case study is implementing a mechanical part in order to study it through meshing analysis. The development of the model consists of the following steps:

- Creating a profile of the part
- Generating a volume defined by the profile
- Generating the mesh for the part

At the end of this case study, you should be able to handle the 2D tools available in GiD as well as the options for generating meshes and visualizing the prototype.



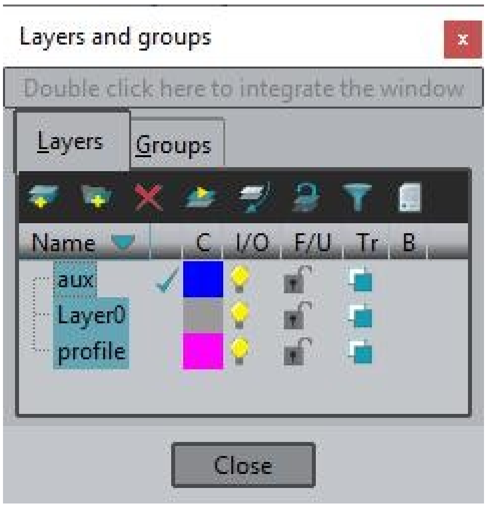
Working by layers

Defining the layers

A geometric representation is composed of four types of entities, namely points, lines, surfaces, and volumes. A layer is a grouping of entities. Defining layers in computer-aided design allows us to work collectively with all the entities in one layer.
The creation of a profile of the mechanical part in our case study will be carried out with the help of auxiliary lines. Two layers will be defined in order to prevent these lines from appearing in the final drawing. The lines that define the profile will be assigned to one of the layers, called the "profile" layer, while the auxiliary lines will be assigned to the other layer, called the "aux" layer. When the design of the part has been completed, the entities in the "aux" layer will be erased.

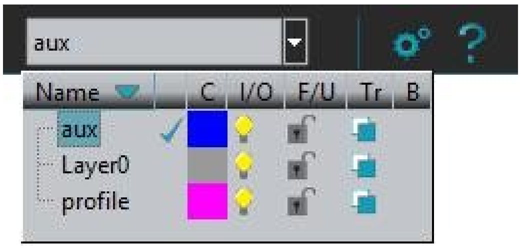
Creating two new layers

- Open the layer management window. This is found in **Utilities->Layers and groups**...
- Create two new layers called "aux" and "profile."
- Choose "aux" as the activated layer. From now on, all the entities created will belong to this layer.



Name		C	I/O	F/U	Tr	B
aux	✓	🟦	💡	🔓	■	
Layer0		■	💡	🔓	■	
profile		🟪	💡	🔓	■	

NOTE: You can also access to the layers information through the standard bar. The layer in use is selected in the combobox.



Name		C	I/O	F/U	Tr	B
aux	✓	🟦	💡	🔓	■	
Layer0		■	💡	🔓	■	
profile		🟪	💡	🔓	■	

Creating a profile

In our case, the profile consists of various teeth. Begin by drawing one of these teeth, which will be copied later to obtain the entire profile.

Creating a size-55 auxiliary line

- Choose the **Line** option, by going to **Geometry->Create->Straight line** or by going to the GiD Toolbar[1].
- Enter the coordinates of the beginning and end points of the auxiliary line[2]. For this example, the coordinates are (0,0) and (55,0), respectively. Besides creating a straight line, this operation implies creating the end points of the line.
- Press **ESC** [3]to indicate that the process of creating the line is finished. Press **ESC** again to end the line creation function.
- If the entire line does not appear on the screen, use the **Zoom Frame** option, which is located in the GiD toolbar and in **Zoom** in the mouse menu.

NOTE: The **Undo** option, located in **Utilities->Undo**, enables you to undo the last operation. **Utilities->Undo Multiple** is similar but open a window in which the operations to be undone can be selected.

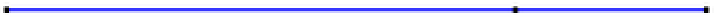
[1] The GiD Toolbar is a window containing the icons for the most frequently executed operations. For information on a particular tool, click on the corresponding icon with the right mouse button.

[2] The coordinates of a point may be entered on the command line, not enclosed in parentheses, either with a space or a comma between them. If the Z coordinate is not entered, it is considered 0 by default. After entering the numbers, press **Return**. Another option for entering a point is using the **Coordinates window**, found in **Utilities->Tools->Coordinates window**.

[3] Pressing the **ESC** key is equivalent to pressing the center mouse button.

Dividing the auxiliary line near coordinates (40, 0)

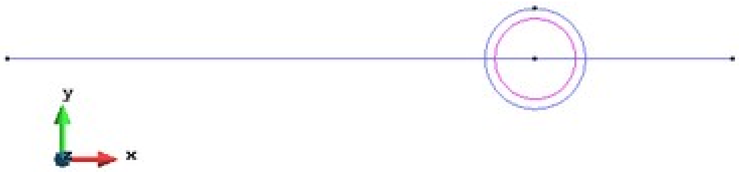
- Choose **Geometry->Edit->Divide->Lines->Near point.** This option will divide the line at the point on the line closest to the coordinates entered.
- Select from **Mouse Menu: Contextual->No Join Ctrl-a.**
- Enter the coordinates of the point that will divide the line. In this example, the coordinates are (40, 0). On dividing the line, a new point (entity) has been created.
- Select the line that is to be divided by clicking on it.
- Press **ESC** to indicate that the process of dividing the line is finished. Press **ESC** again to finish the dividing function.



Creating a 3.8-radius circle around point (40, 0)

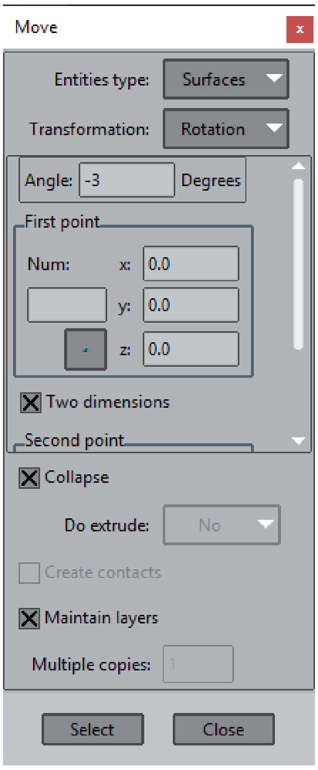
- Choose the option **Geometry->Create->Object->Circle**.
- The center of the circle (40, 0) is a point that already exists. To select it, go to **Contextual->Join Ctrl-a** in the mouse menu (right-click). The pointer will become a square, which means that you may click an existing point.
- The **Enter Normal** window appears. Set the normal as Positive Z and press **OK**.
- Enter the radius of the circle. The radius is 3.8 (In GiD the decimals are entered with a point, not a comma). Two circumferences are visualized; the inner circumference in pink represents the surface of the circle in 'normal' render mode.

Press **ESC** to indicate that the process of creating the circle is finished.



Rotating the circle around a point

- Use the **Move** window, which is located in **Utilities->Move**.
- Within the **Move** menu and from among the **Transformation** possibilities, select **Rotation**. The type of entity to receive the rotation is a surface, so from the **Entities type** menu, choose **Surfaces.**
- Enter -3 in the **Angle** box and check the **Two dimensions** option. (Provided we define positive 2D rotation in the mathematical sense, which is counter-clockwise, -3 degrees equates to a clockwise rotation of 3 degrees).
- Enter the point (0, 0, 0) under **First Point**. This is the point that defines the center of rotation.
- Click **Select** to select the surface that is to rotate, which in this case is that of the circle.
- Press **ESC** (or **Finish** in the **Move window**) to indicate that the selection of surfaces to rotate has been made, thus executing the rotation.



Move

Entities type: Surfaces

Transformation: Rotation

Angle: -3 Degrees

First point

Num: x: 0.0

y: 0.0

z: 0.0

☒ Two dimensions

Second point

☒ Collapse

Do extrude: No

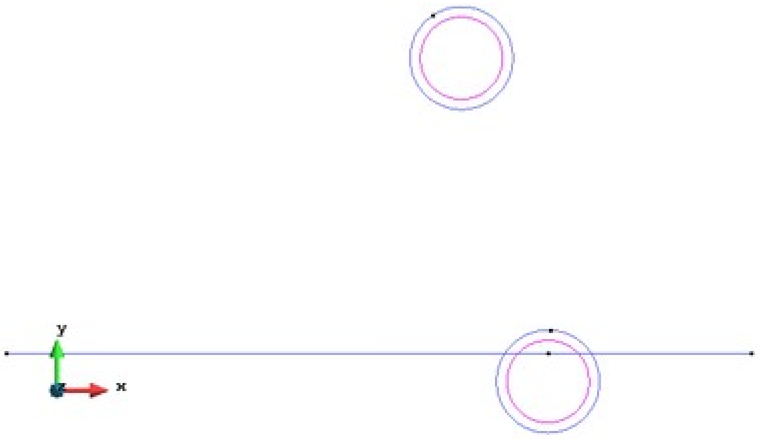
□ Create contacts

☒ Maintain layers

Multiple copies: 1

Rotating the circle 36 degrees around a point and copying it

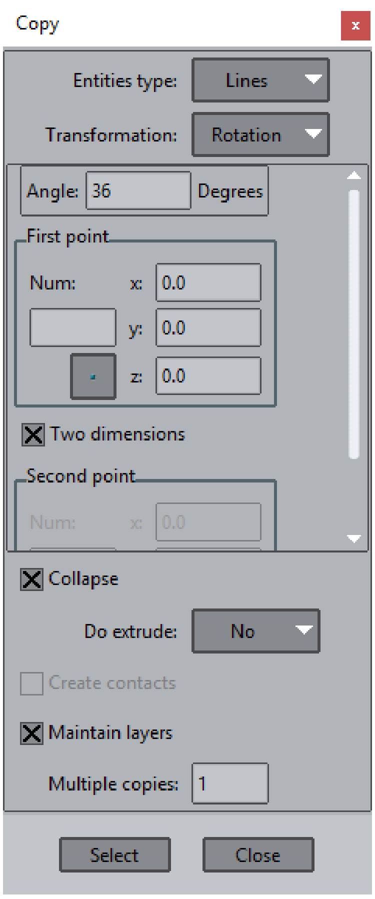
- Use the **Copy** window, located in **Utilities->Copy**.
- Repeat the rotation process explained in section Rotating the circle around a point, but this time with an angle of 36 degrees.



NOTE: The **Move** and **Copy** operations differ only in that **Copy** creates new entities while **Move** displaces entities.

Rotating and copying the auxiliary lines

- Use the **Copy** window, located in **Utilities->Copy**.



Copy

Entities type: Lines

Transformation: Rotation

Angle: 36 Degrees

First point

Num: x: 0.0

y: 0.0

z: 0.0

☒ Two dimensions

Second point

Num: x: 0.0

☒ Collapse

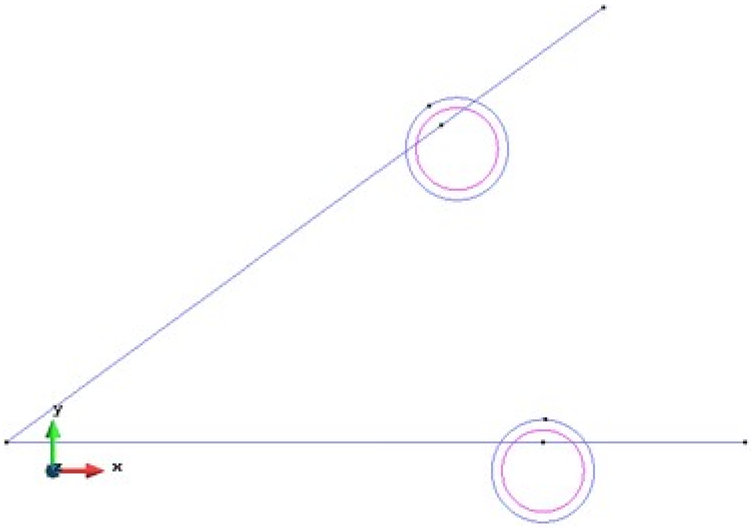
Do extrude: No

□ Create contacts

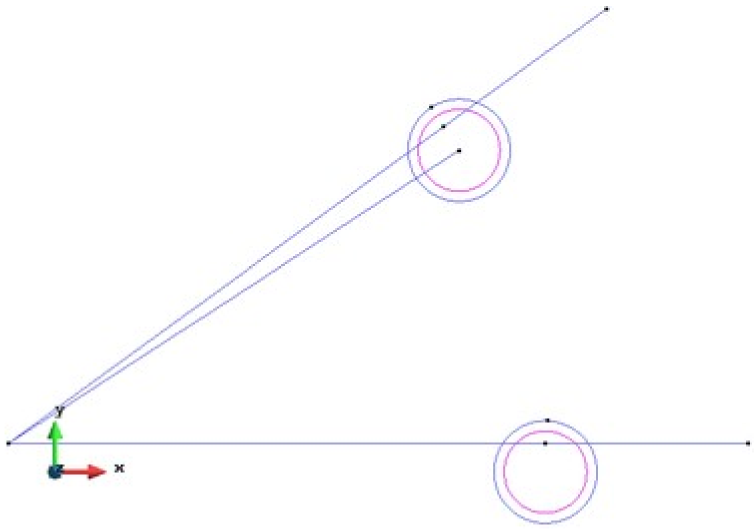
☒ Maintain layers

Multiple copies: 1

- Repeat the rotating and copying process from section for the two auxiliary lines. Select the option **Lines** from the **Entities type** menu and enter an angle of 36 degrees.
- Select the lines to copy and rotate. Do this by clicking **Select** in the **Copy** window.
- Press **ESC** to indicate that the process of selecting lines is finished, thus executing the task.



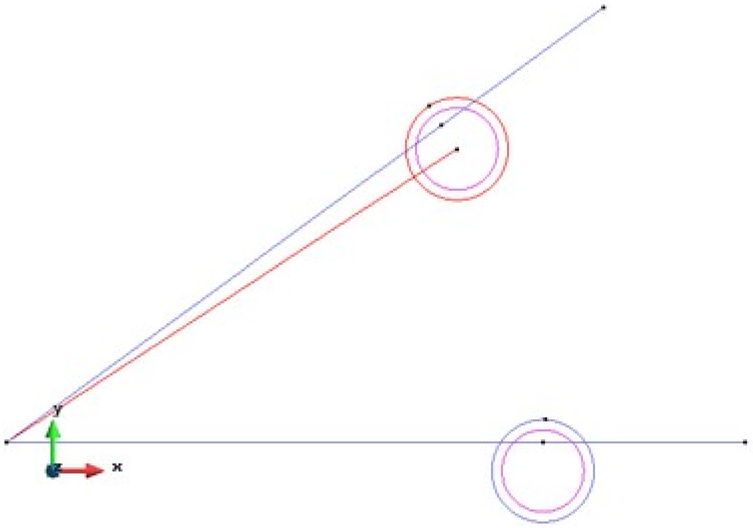
- Rotate the line segment that goes from the origin to point (40, 0) by 33 degrees and copy it.



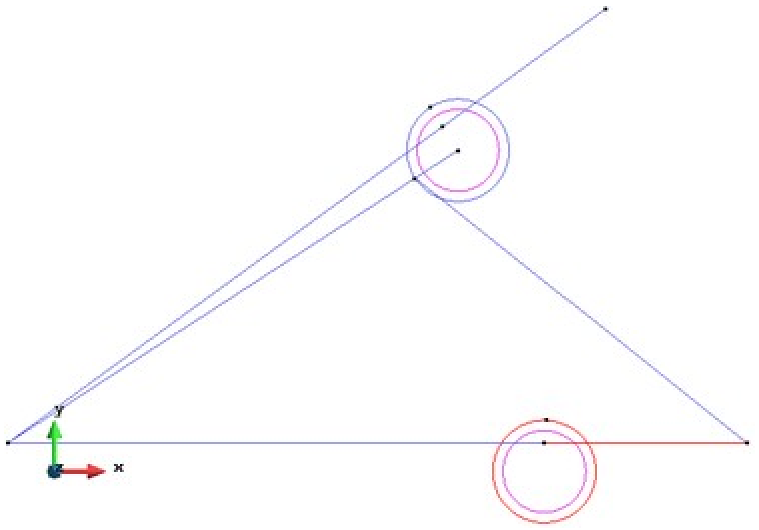
NOTE: In the **Copy** window, the button ■ may be used to select existing points with the mouse, or alternatively enter its number in the entry field.

Intersecting lines

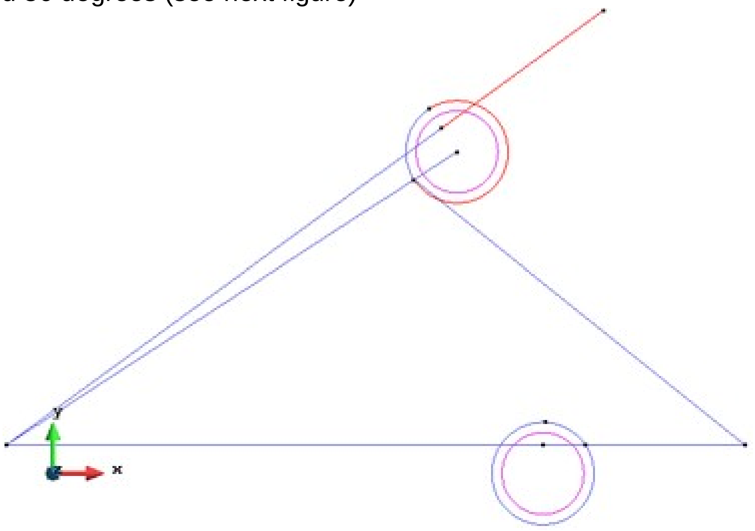
- Choose the option **Geometry->Edit-> Intersection->Lines.**
- Select the upper circumference
- Select the line resulting from the 33-degree rotation (see next figure)



- Press **ESC** to conclude the intersection of lines.
- Press **ESC** to finish the intersection function.
- Create a line (**Geometry->Create->Straight line**) between the existent point (55, 0) and the point generated by the intersection.
- Choose the option **Geometry->Edit-> Intersection->Lines** in order to make another intersection between the lower circle and the line segment between point (40, 0) and point (55, 0) (see next figure)

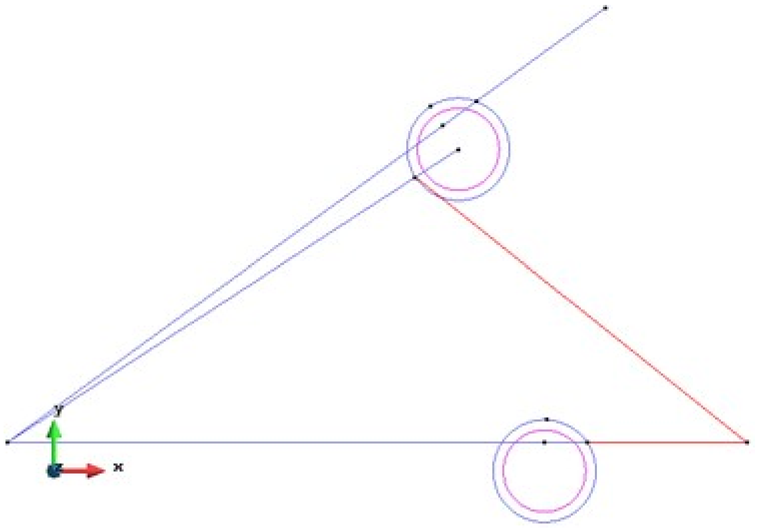


- Then continue selecting to make an intersection between the upper circle and the farthest segment of the line that was rotated 36 degrees (see next figure)



Creating an arc tangential to two lines

- Choose **Geometry->Create->Arc->Fillet curves**.
- Enter a radius of 1.35 in the command line.
- Now select the two line segments shown in next figure. Then press **ESC** to indicate that the process of creating the arcs is finished.



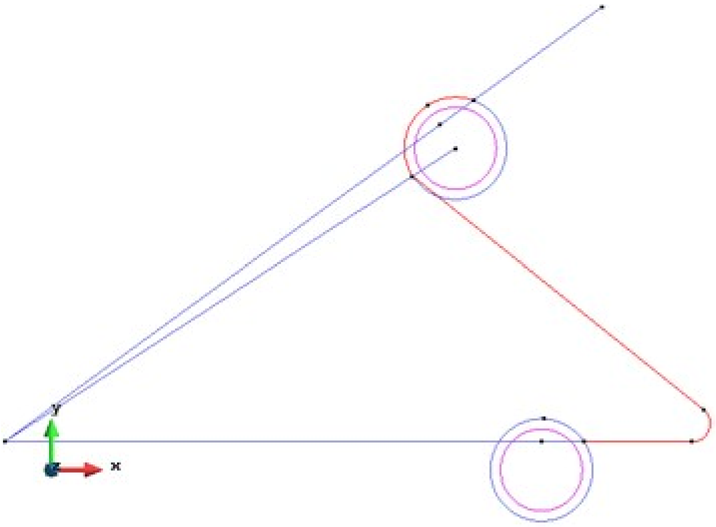
Changing the definitive lines to the 'profile' layer

The auxiliary lines will be eliminated and the "profile" layer will contain only the definitive lines.

- Select the "profile" layer in the **Layers** window.
- We will move the lines defining the profile to the "profile" layer:
    -
        - Click on the icon **(Send To)** and select **Lines**

Be sure that 'Also lower entities' is checked, to send to this layer also the points of the lines.

- Select only the lines that form the profile (see next figure).
- To conclude the selection process, press the **ESC** key or click **Finish** in the **Layers** window.



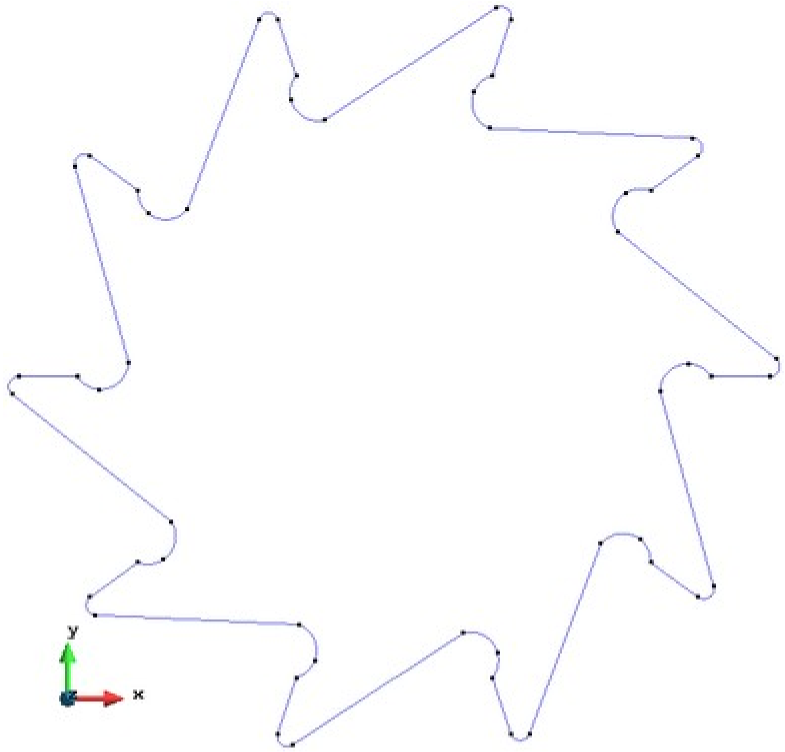
Deleting the 'aux' layer

- Select the "profile" layer and set it off. Click on the light bulb icon 💡
- Check that 'Also lower entities' is not checked in the delete menu (to avoid delete the lines of the surfaces)
- Choose **Geometry->Delete->All Types** (or use the GiD Toolbar).
- Select all the lines and surfaces that appear on the screen.
- Press **ESC** to conclude the selection of elements to delete.
- Select the "aux" layer and delete it lick in the ✕ icon
- Select the "profile" layer and set it on.

NOTE: To cancel the deletion of elements after they have been selected, open the mouse menu, go to **Contextual** and choose **Clear Selection**.
NOTE: Elements forming part of higher level entities may not be deleted. For example, a point that defines a line may not be deleted.
NOTE: A layer containing entities will ask for confirmation to be deleted.

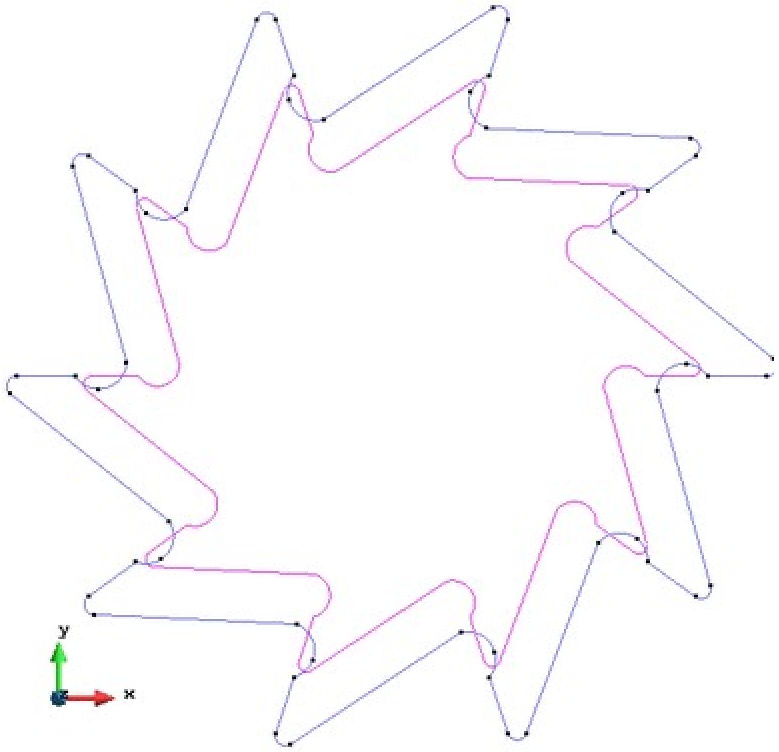
Rotating and obtaining the final profile

- Make sure that the activated layer is the "profile" layer. (Use the option **To use** or simply double-click the layer).
- In the **Copy** window, select the line rotation (**Lines, Rotation**).
- Enter an angle of 36 degrees. Make sure that the center is point (0, 0, 0) and that you are working in two dimensions.
- In the **Multiple Copies** box enter 9. This way, 9 copies will be made, thus obtaining the 10 teeth that form the profile of the model (9 copies and the original).
- Ensure that **Collapse** is set, to avoid duplicated points on the same location.
- Click **Select** and select the lines defining the profile. Press the **ESC** key or click **Finish** in the **Copy** window in order to conclude the operation. The result is shown in next figure.



Creating a surface

- Create a NURBS surface. To do this, select the option **Geometry->Create->NURBS surface->By contour.** This option can also be found in the GiD Toolbar.
- Select the lines that define the profile of the mechanical part and press **ESC** to create the surface.
- Press **ESC** again to exit the function.



NOTE: To create a surface there must be a set of lines that define a closed contour.

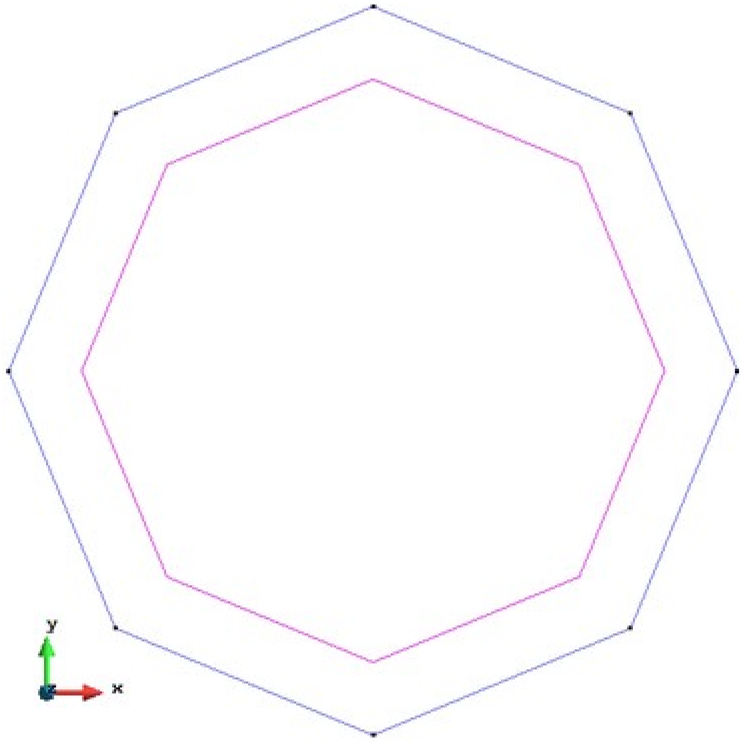
Creating a hole in the mechanical part

In the previous sections we drew the profile of the part and we created the surface. In this section we will make a hole, an octagon with a radius of 10 units, in the surface of the part. First we will draw the octagon.

- Select from the menu **Geometry->Create->Object->Polygon** to create a regular polygon.
- Enter 8 as the number of sides of the polygon.
- Enter (0,0,0) as the center of the polygon. (use Ctrl-a keys to swap to select new point mode if required)
- Select Positive Z as the normal of the polygon, this mean a normal direction (0,0,1)
- Enter 10 as the radius of the polygon and press **ENTER**. Press **ESC** to finish the action.

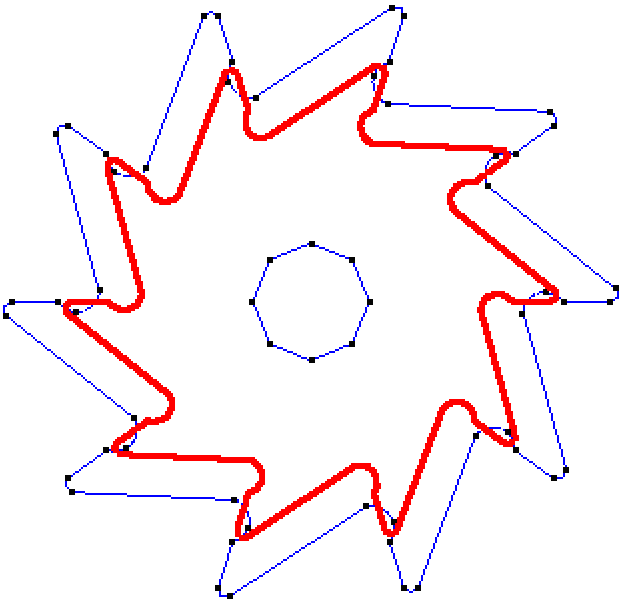
We get the result as shown in following figure. As we only need the boundary we should remove the associated surface. Select the option **Geometry->Delete->Surfaces** and then select the surface of the octagon. Press **ESC** to finish.

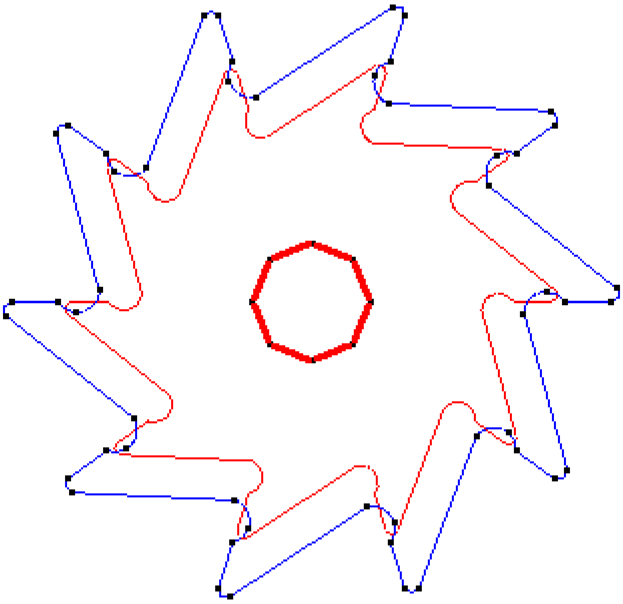
Note: check that 'Also lower entities' is not checked in the delete menu, otherwise also the lines and points of the surface will be deleted.



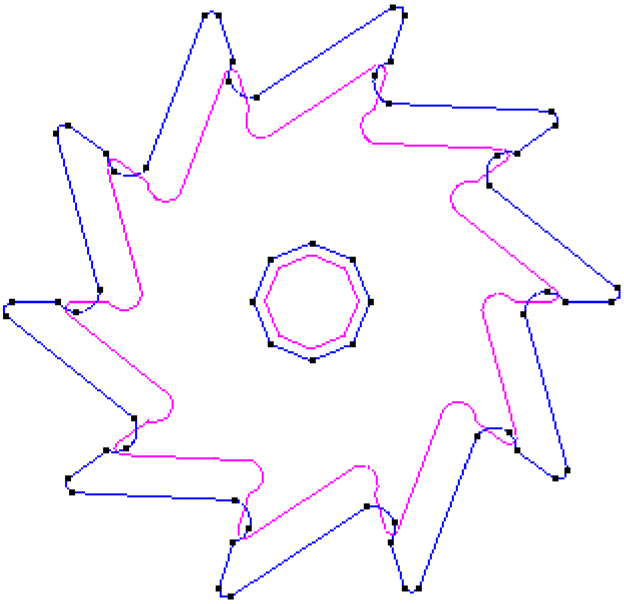
Creating a hole in the surface of the mechanical part

- Choose the option **Geometry->Edit->Hole surface->With lines**.
- Select the surface in which to make the hole (Left figure).
- Select the lines that define the hole (Right figure) and press **ESC**.





- Again, press **ESC** to exit this function.



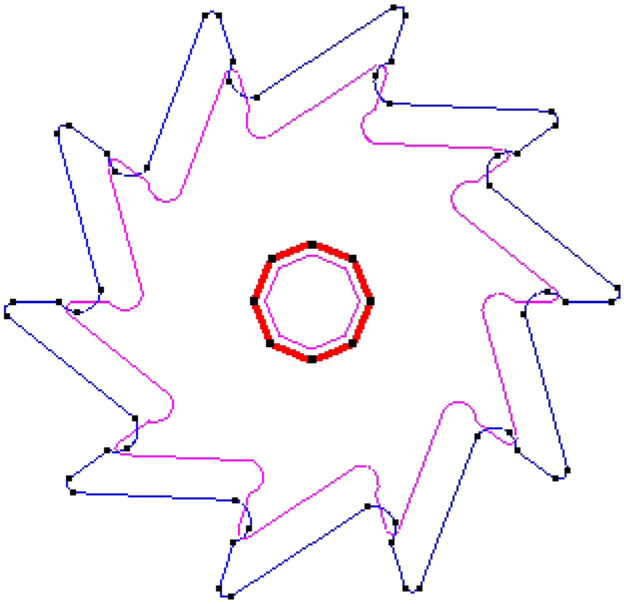
The model part with the hole in it

Creating volumes from surfaces

The mechanical part to be constructed is composed of two volumes: the volume of the wheel (defined by the profile), and the volume of the axle, which is a prism with an octagonal base that fits into the hole in the wheel. Creating this prism will be the first step of this stage. It will be created in a new layer that we will name "prism".

Creating the 'prism' layer and translating the octagon to this layer

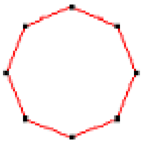
- In the **Layers** window, create a new layer named "prism".
- Select the "prism" layer and double-click it to choose as the activated layer.
- Right-click on "prism" layer and select **Send To->Lines**. Select the lines that define the octagon. Press **ESC** to conclude the selection.



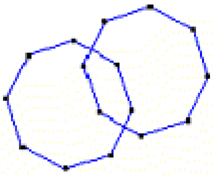
- Select the "profile" layer and set it **Off**.

Creating the volume of the prism

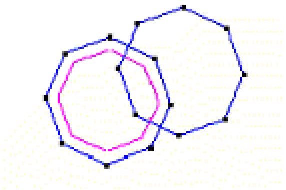
- First copy the octagon a distance of 50 units relative to the surface of the wheel, which is where the base of the prism will be located. In the **Copy** window, choose **Translation** and **Lines**. Since we want to translate 50 units, enter two points that define the vector of this translation, for example (0, 0, 0) and (0, 0, 50). (Make sure that the Multiple Copies value is 1).
- Choose **Select** and select the lines of the octagon. Press **ESC** to conclude the selection.



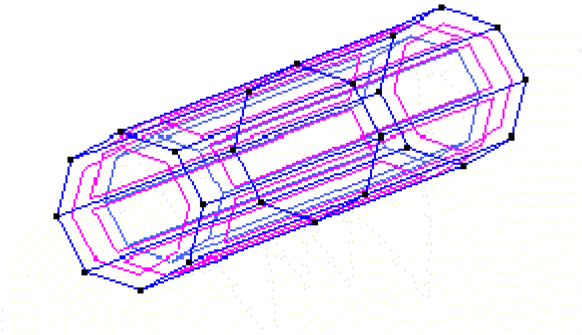
- Since the Z axis is parallel to the user's line of vision, the perspective must be changed to visualize the result. To do this, use the **Rotate Trackball** tool, which is located in the GiD Toolbox and in the mouse menu. (or press <Shift> key and drag the right-mouse button to rotate the view)



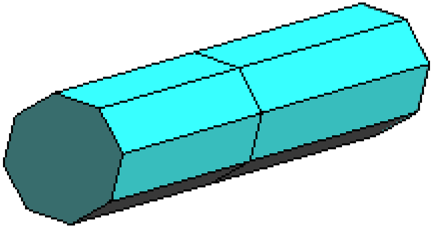
- Choose **Geometry->Create->NURBS surface->By contour**. Select the lines that form the displaced octagon and press **ESC** to conclude the selection. Again, press **ESC** to exit the function of creating the surfaces.



- In the **Copy** window, choose **Translation** and **Surfaces**. Make a translation of 110 units. Enter two points that define a vector for this translation, for example (0, 0, 0) and (0, 0, -110).
- To create the volume defined by the translation, select **Do Extrude Volumes** in the **Copy** window.
- Click **Select** and select the surface of the octagon. Press **ESC**. The result is shown in next figure.



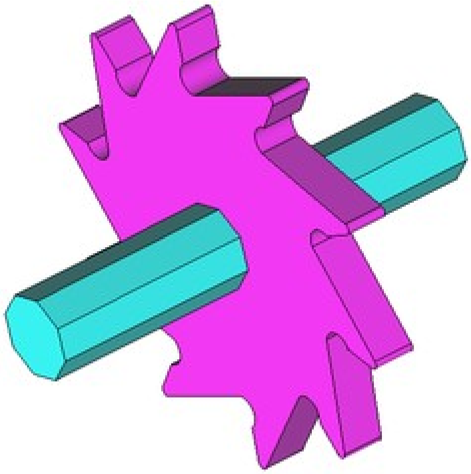
- Choose the option **Render->Flat** from the mouse menu to visualize a more realistic version of the model. Then return to the normal visualization using **Render->Normal**.



NOTE: The **Color** option in the **Layers** window lets you define the color and the opacity of the selected layer. This color is then used in the rendering of elements in that layer.

Creating the volume of the wheel

- Visualize the "profile" layer and activate it. The volume of the wheel will be created in this layer. Set off the "prism" layer in order to make the selection of the entities easier.
- In the **Copy** window, choose **Translation** and **Surfaces**. A translation of 10 units will be made. To do this, enter two points that define a vector for this translation, for example (0, 0, 0) and (0, 0, -10).
- Choose the option **Do Extrude Volumes** from the **Copy** window. The volume that is defined by the translation will be created.
- Make sure that the **Maintain Layers** option is not checked, hence the new entities created will be placed in the layer to use; otherwise, the new entities are copied to the same layers as their originals
- Click **Select** and select the surface of the wheel. Press **ESC**.
- Select the two layers and click them **On** so that they are visible.
- Choose **Render->Flat** from the mouse menu to visualize a more realistic version of the model.



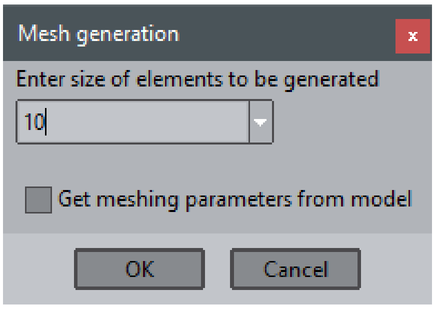
Generating the mesh

Now that the part has been drawn and the volumes created, the mesh may be generated. First we will generate a simple mesh by default.

Depending on the form of the entity to be meshed, GiD performs an automatic correction of the element size. This correction option, which by default is activated, may be modified in the **Meshing->General** branch of the **Preferences** window, under the option **Automatic correct sizes**. Automatic correction is sometimes not sufficient. In such cases, it must be indicated where a more precise mesh is needed. Thus, in this example, we will increase the concentration of elements along the profile of the wheel by following two methods: 1) assigning element sizes around points, and 2) assigning element sizes around lines.

Generating a coarse mesh

- Choose **Mesh->Generate mesh**.
- A window comes up in which to enter the desired edge element size of the mesh to be generated. Set a big size of 10 units to have a coarse mesh and click **OK**.



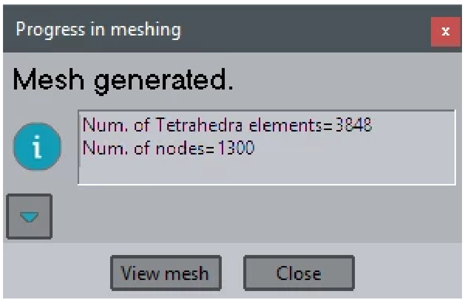
Mesh generation

Enter size of elements to be generated

10

□ Get meshing parameters from model

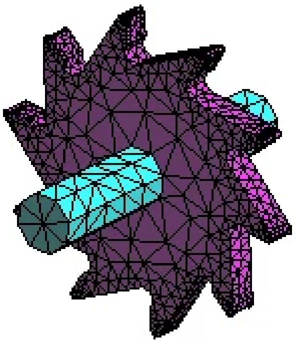
- A window appears showing how the meshing is progressing. Once the process is finished it show information about the mesh that has been generated. Click **View mesh** to visualize the resulting mesh. (Note that the number of nodes and elements may vary slightly depending on the version of GiD used).



Progress in meshing

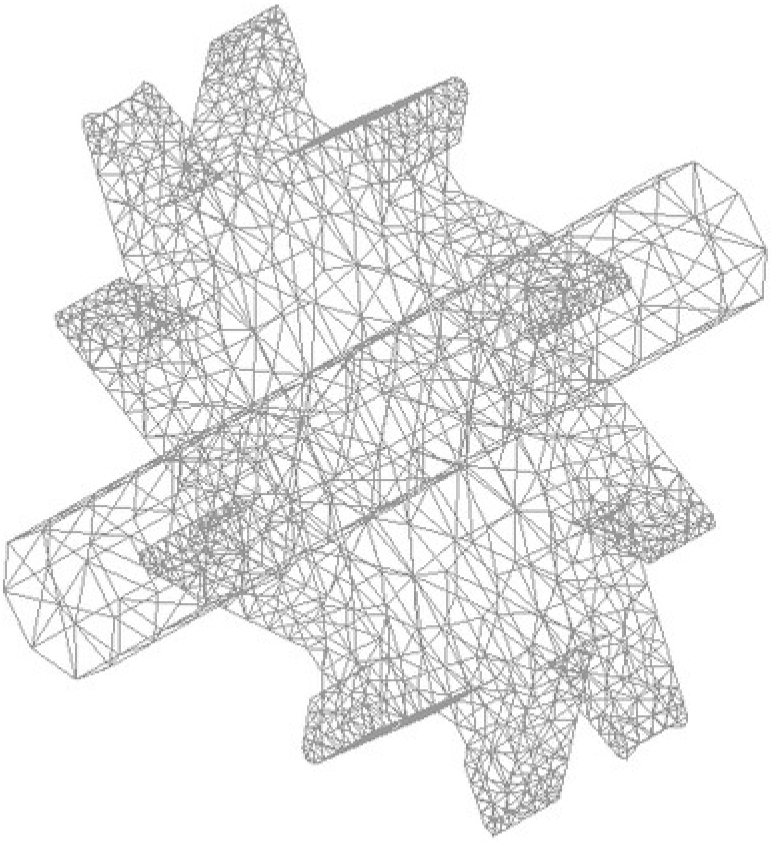
Mesh generated.

Num. of Tetrahedra elements=3848
Num. of nodes=1300

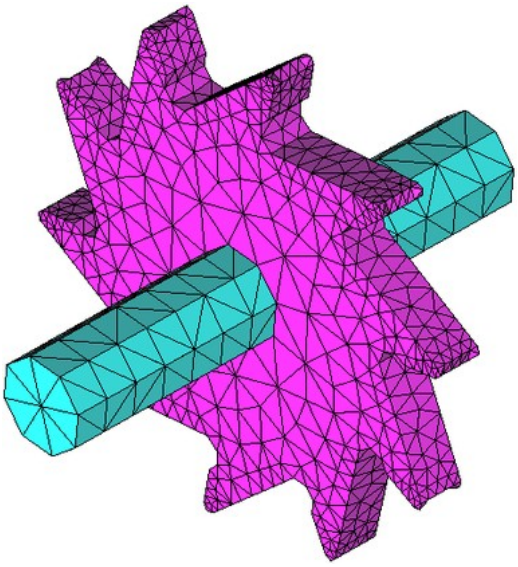


- Use the **Mesh->View mesh boundary** option to see only the contour of the volumes meshed without the interiors. This visualization mode may be combined with the various rendering methods.

- A window appears asking if we want to maintain this visualization mode. Click **No**. To exit the mesh boundary visualization mode press **ESC**.



- Visualize the mesh generated with the various rendering options in the **Render** menu, located in the mouse menu.

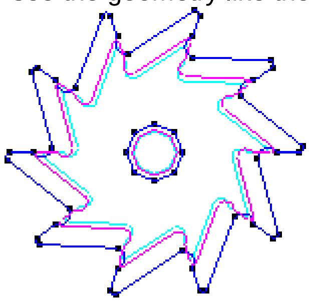


- Choose **View->Mode->Geometry** to return to the visualization of the geometric model.

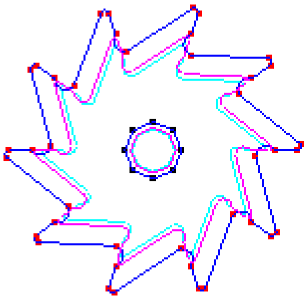
NOTE: To visualize the geometry of the model use **View->Mode->Geometry**. To visualize the mesh use **View->Mode->Mesh**.

Generating the mesh with assignment of size around points

- Choose **View->Rotate->Plane XY (Original)**. This way we will have a side view, and **View->Mode->Geometry** and **Render->Normal** to see the geometry like the image

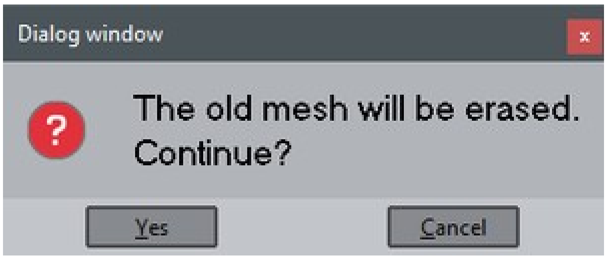


- Choose **Mesh->Unstructured->Assign sizes on points**. A window appears in which to enter the element size around the point to be selected. Enter 0.7 and click **Assign**.
- Select only the points on the wheel profile (see next figure). One way of doing this is to select the entire part and then deselect the points that form the prism hole. Press **ESC** to conclude the selection process.
- The window appears again, click **Close** to finish.



- Choose **Mesh->Generate mesh**.
- A window opens asking if the previous mesh should be eliminated. Click **Yes**. Another window appears in

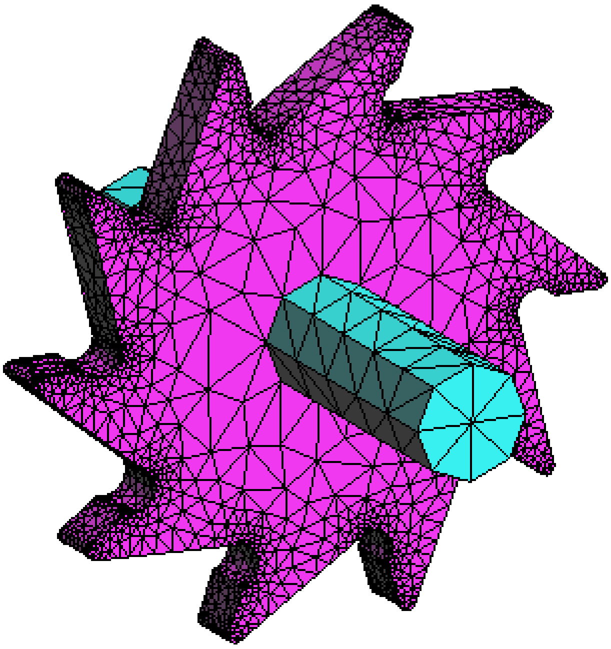
which the desired element size should be entered. Leave the previous value of 10 unaltered.



Dialog window

The old mesh will be erased. Continue?

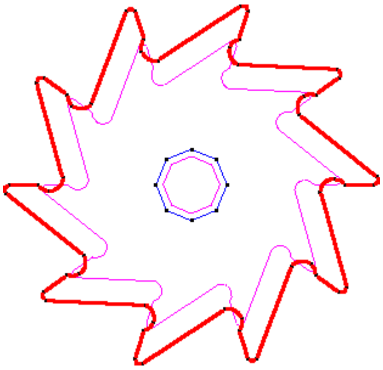
- A third window shows the meshing process. Once it has finished, click **View mesh** to visualize the resulting mesh.



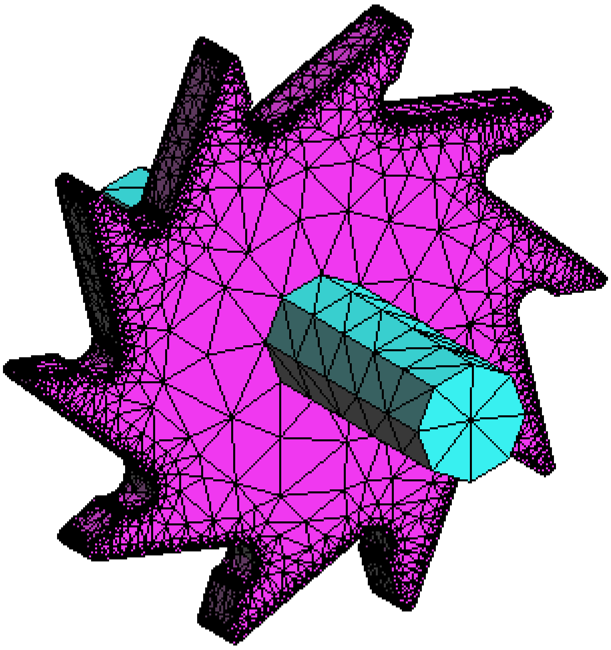
- A greater concentration of elements has been achieved around the points selected.
- Choose **View->Mode->Geometry** to return to this visualization.

Generating the mesh with assignment of size around lines

- Open the **Preferences** window, which is found in **Utilities**, and select the **Meshing->General** branch. In this window there is an option called **Size transitions** which defines the size gradient of the elements. A high transition number means a fast grown of small sizes. Select a transition size of **0.8** to have a fast transition and then obtain few elements. Click **Apply**.
- Choose **Mesh->Reset mesh data** to delete the previously assigned sizes.
- Choose **Mesh->Unstructured->Assign sizes on lines**. A window appears in which to enter the element size around the lines to be selected. Enter size 0.7 and click **Assign**.
- Select only the lines of the wheel profile (see next figure) in the same way as in previous section and press **ESC**.
- The window appears again, click **Close** to finish.

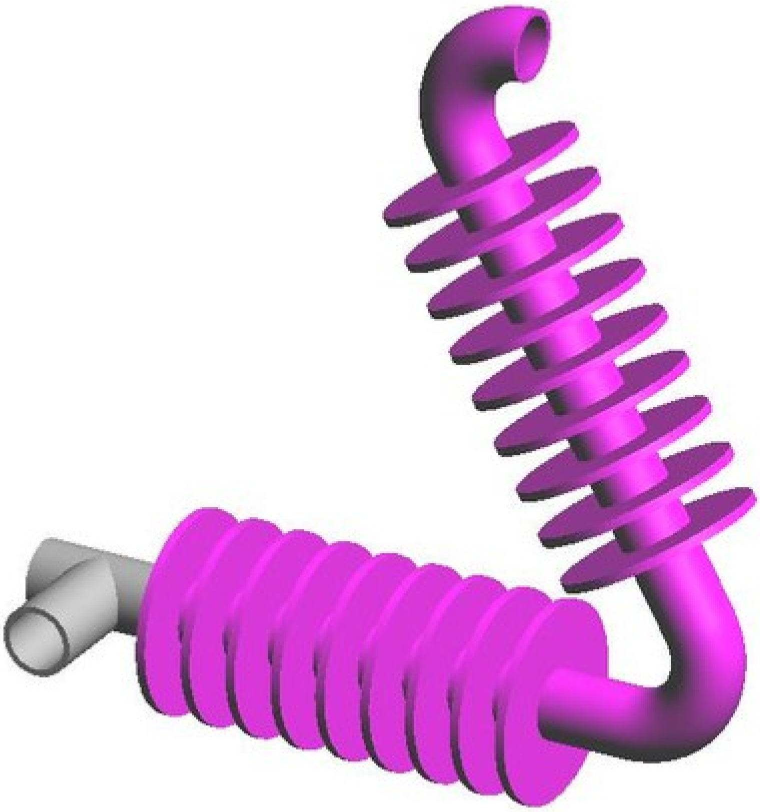


- Choose **Mesh->Generate mesh**. A window appears asking if the previous mesh should be eliminated. Click **Yes**.
- Another window opens in which the maximum element size should be entered. Leave the last value unaltered and click **View mesh**.
- A greater concentration of elements has been achieved around the selected lines. In contrast to the case in previous section, this mesh is more accurate since lines define the profile much better than points do.



IMPLEMENTING A COOLING PIPE

ADVANCED 2D & 3D TECHNIQUES AND MESHING
IMPLEMENTING A COOLING PIPE



This case study shows the modeling of a more complex piece and concludes with a detailed explanation of the corresponding meshing process. The piece is a cooling pipe composed of two sections forming a 60-degree angle.
The modeling process consists of four steps:

- Modeling the main pipes
- Modeling the elbow between the two main pipes, using a different file
- Importing the elbow to the main file
- Generating the mesh for the resulting piece

At the end of this case study, you should be able to use the CAD tools available in GiD as well as the options for generating meshes and visualizing the result.

Working by layers

Various auxiliary lines will be needed in order to draw the part. Since these auxiliary lines must not appear in the final drawing, they will be in a different layer from the one used for the finished model.

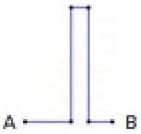
Create the layers called "part1", "union" and delete the layer "Layer0"
Choose "part1" as the activated 'layer to use' with a <Double-click>. From now on, all the entities created will belong to this layer.

Creating a component part

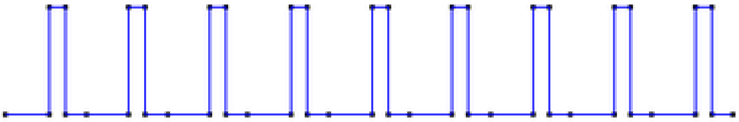
In this section the entire model, except the T junction, will be created. The model to be created is composed of two pipes forming a 60-degree angle. To start with, the first pipe will be created. This pipe will then be rotated to create the second pipe.

Creating the profile

- Choose the **Line** option, located in **Geometry->Create->Straight line.**
- Enter the following new points in the command line: (0, 11), (8, 11), (8, 31), (11, 31), (11, 11) and (15, 11). Press **ESC** twice to indicate that the process of creating lines is finished.



- From the **Copy** window, choose **Lines** and **Translation**. A translation defined by points A (0, 11) as first point and B (15, 11) as second point will be made. In the **Multiple copies** option, enter 8 (the number of copies to be added to the original). Be sure than the 'Collapse' option is set, and then Select the lines that have just been drawn and press **ESC**.

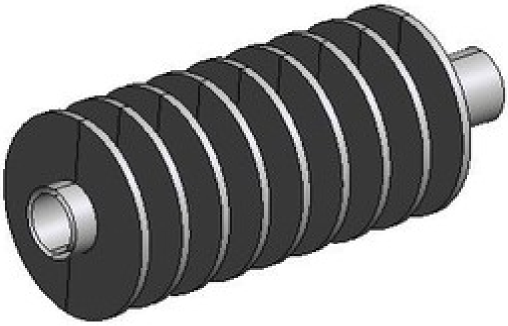


- Create a line using **Geometry->Create->Straight line.** Use the contextual option **Join** or press **<Ctrl-a>** and select the last point on the profile (at the right part of the profile). Now choose the option **No join Ctrl-a** and enter new point (160, 11) in the command line. Press **ESC** twice to finish the process of creating lines.
- Again, choose the **Line** option and enter the new points (0, 9) and (160, 9). Press **ESC** twice to conclude the process of creating lines.

Creating the surfaces by revolution

Rotation of the profile will be carried out in two rotations of 180 degrees each.

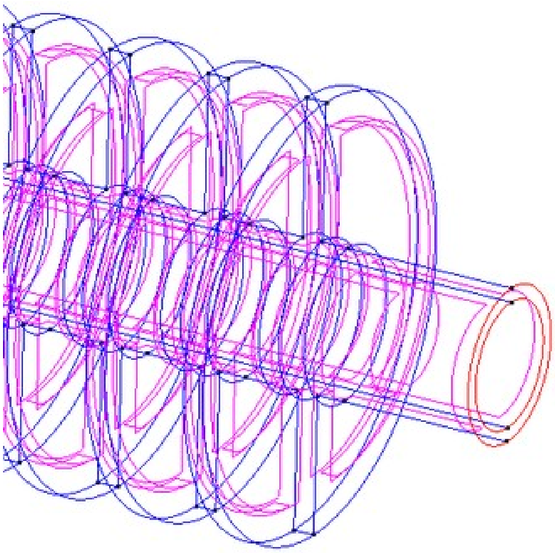
- From the **Copy** window, select **Lines** and **Rotation**. Enter an angle of 180 degrees. The axis of rotation is that defined by the line that goes from point (0, 0) to point (200, 0). Enter these two points as the **First point** and **Second point** (Two dimensions must be unchecked). From the **Do extrude** menu, select **Surfaces**.Be sure to enter 2 in **Multiple copies**. and select all lines and press **ESC** when the selection is finished.
- Rotate the view from the mouse menu **Rotate->Trackball** and choose **Render->Flat** to visualize a more realistic version of the model.



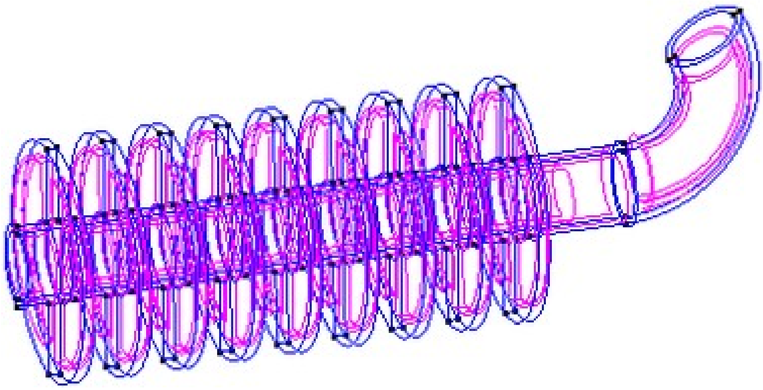
- Return to the normal visualization with Render->Normal. This option is more comfortable to work with. To return to the side view (elevation), choose in the mouse menu Rotate->Plane XY (Original).

Creating the union of the main pipes

- Choose the **Zoom->In** option from the mouse menu. Magnify the right end of the model and rotate the view to facilitate the selection.
- Set "union" as current layer to use, with a <Double-click>
- From the **Copy** window, select **Lines** and **Rotation**. Enter an angle of 120 degrees and select the rotation center (160, 25) as First point. Since the rotation may be done in 2D, choose the option **Two dimensions**. From the **Do extrude** menu, select **Surfaces** and be sure that **Multiple copies** is 1 and **Maintain layers** is unset, then the new entities will be created in the layer to use instead of the layer of the source curves.



- Click **Select** and select the four lines that define the right end of the pipe (see figure above). Press **ESC** when the selection is finished.

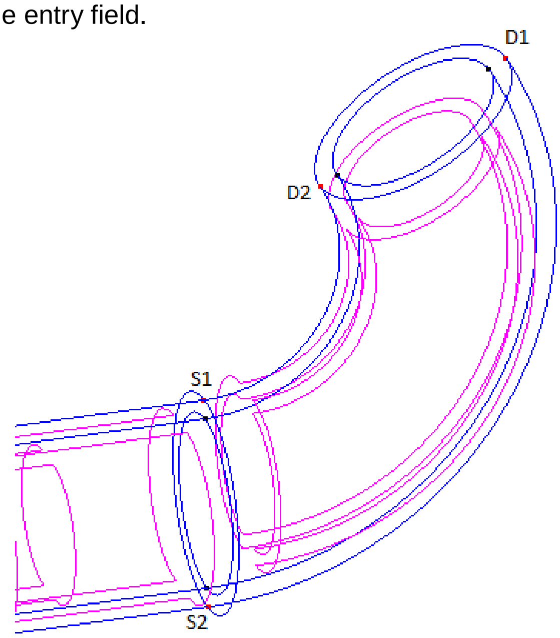


Copying the main pipe

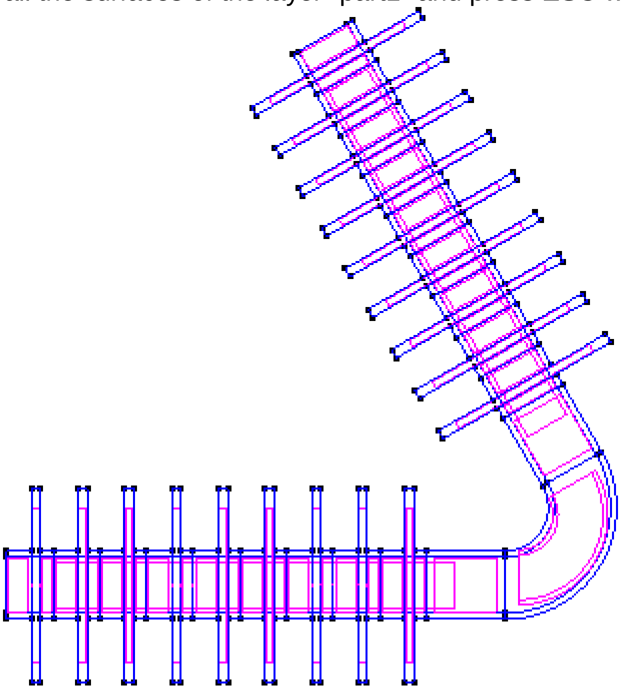
Align uses a rigid body movement defined by three source points and its destination points.

- From the **Copy** window, select **Surfaces** and **Align**. Choose the **Two Dimensions** option. The source points S1, S2 and its destination points D1, D2 are highlighted in the image. Ensure the **Do Extrude** menu is set to **No**. and set **Maintain layers**.

NOTE: In the **Copy** window, the button □ may be used to select existing points with the mouse, or alternatively enter its number in the entry field.

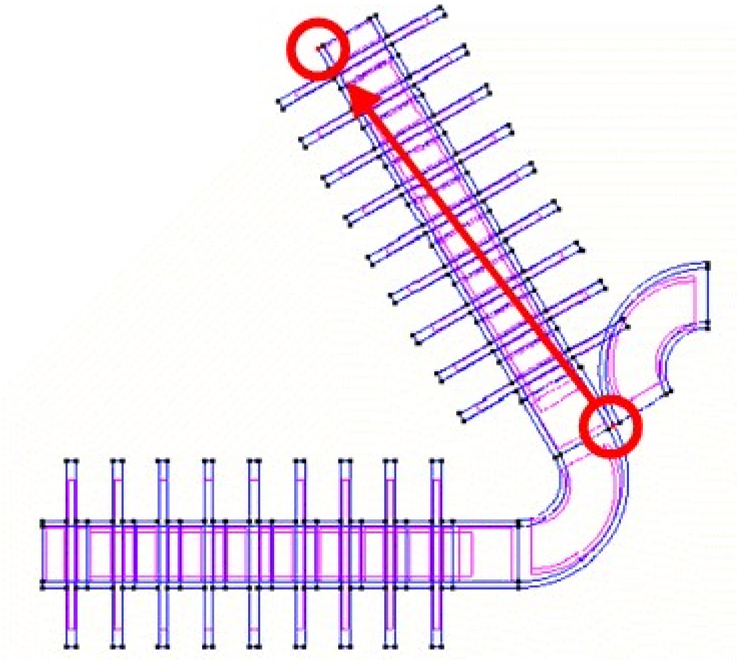


- Click **Select** and select all the surfaces of the layer "part1" and press **ESC** when the selection is finished.

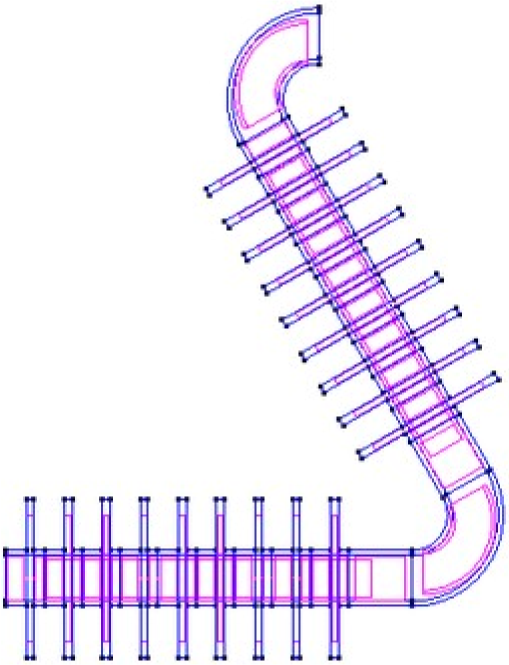


Creating the end of the pipe

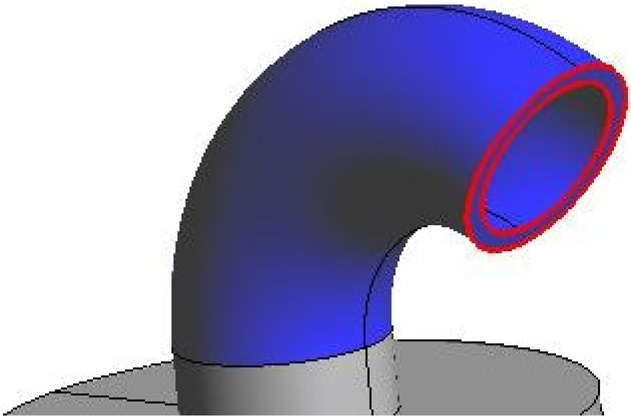
- From the **Copy** window, select **Surfaces** and **Rotation**. Enter an angle of 180 degrees. Since the rotation may be done in 2D, choose the option **Two Dimensions**. The center of rotation is the upper right point of the pipe elbow. Make sure the **Do Extrude** menu is set to **No**.
- Click **Select** and select the surfaces that join the two pipe sections and press **ESC**.
- Select **Utilities->Move** window, select **Surfaces** and **Translation**. The points defining the translation vector are circled in next figure.



- Click **Select** and select the surfaces created in point 1. Press **ESC.** The result should be as is shown.



- To create a ring surface, choose **Geometry->Create->NURBS Surface->By contour** and select the four lines that define the opening of the pipe (see next figure). Press **ESC** twice.



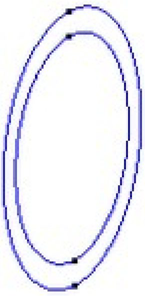
Opening at the end of the pipe
From the **Files** menu, choose **Save** in order to save the file. Enter a name for the file and click **Save**.

Creating the T junction

Now, an intersection composed of two pipe sections will be created in a separate file. Then this file will be imported to the original model to create the entire piece.

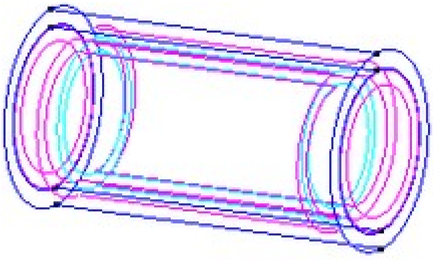
Creating one of the pipe sections

- Choose **Files->New**, thus starting work in a new file.
- Rename the layer 'Layer0' to 'pipe1' and create two new child layers "inner" and "outer". Set 'pipe1//inner' as layer to use with a <Double-click>
- Choose **Geometry->Create->Point** and enter new point (0, 9)
- set 'pipe1//outer' as layer to use and create the new point (0, 11). Press **ESC** to conclude the creation of points.
- From the **Copy** window, select **Points** and **Rotation**. Enter an angle of 180 degrees and from the **Do extrude** menu, select **Lines**. The axis of rotation is the x axis. Enter two points defining the axis, one in **First Point** and the other in **Second Point**, for example, (0, 0, 0) and (1, 0, 0) , check **Maintain layers**, and set **Multiple copies** to 2.
- Click **Select** and select the two points just created. Press **ESC**.



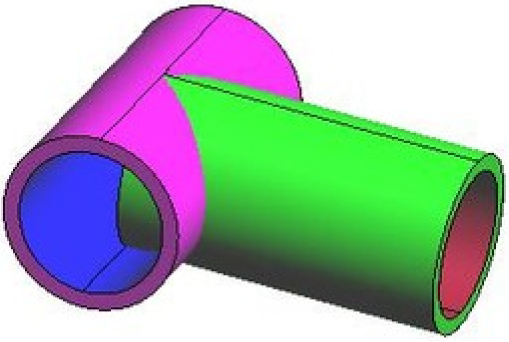
- Create a surface: choose **Geometry->Create->NURBS Surface->By contour** and select the four lines. Press **ESC** twice.
- From the **Copy** window, choose **Surfaces** and **Translation**. In **First Point** and **Second Point**, enter the points defining the translation vector. Since the pipe section must measure 40 length units, the vector is defined by points (0, 0, 0) and (-40, 0, 0).

- From the **Do extrude** menu, choose the **Volumes** option, and set **Multiple copies** to 1.
- Click **Select** to select the surface. Press **ESC** to conclude the selection process.



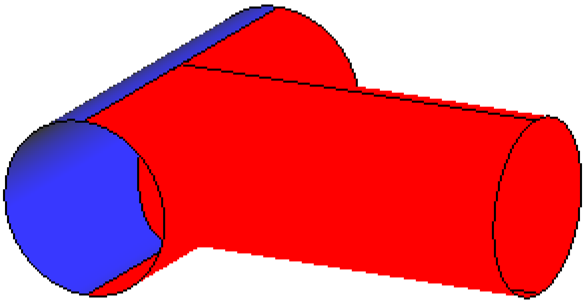
Creating the other pipe section

- Create a new layer named "pipe2" with two child layers "inner" and "outer". Set "pipe2//inner" as the 'layer to use', and set off the layers "pipe1"
- Choose **Geometry->Create->Point** and enter points (-20, 9) and (-20, 11). Press **ESC** to conclude the creation of points.
- Change the layer of the second point to "pipe2//outer"
- From the **Copy** window, select **Points** and **Rotation**. Enter an angle of 180 degrees and from the **Do extrude** menu, select **Lines**. Since the rotation can be done on the **xy** plane, choose **Two Dimensions**. The center of rotation is the coordinates (-20, 0, 0), and set **Multiple copies** to 2
- Click **Select** and select the two points just created.
- Create a surface: choose **Geometry->Create->NURBS Surface->By contour** and select the four lines. Press **ESC** twice.
- From the **Copy** window, select **Surfaces** and **Translation**. In **First Point** and **Second Point** enter the points defining the translation vector. Since this pipe section must also measure 40 length units, the vector is defined by points (0, 0, 0) and (0, 0, 40).
- From the **Do extrude** menu, select the **Volumes** option and set **Multiple copies** to 1.
- Click **Select** to select the surface and press **ESC** to conclude the selection.



Creating the lines of intersection

- Set off the layers "outer" to facilitate the selection
- Choose **Geometry->Edit->Intersection->Surfaces**.
- Select the three inner surfaces of the pipes that are intersecting.



- Repeat the process with the three outer surfaces of the pipes that are intersecting.

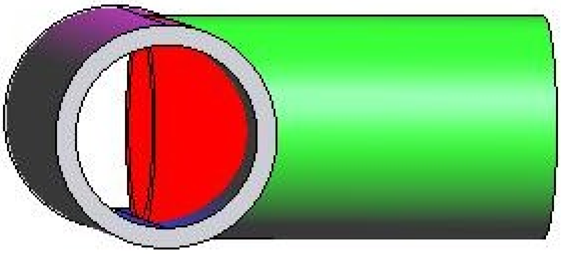
Now the intersection lines are created and some surfaces are splitted by these lines.

Deleting surfaces and close a volume

- Choose **Geometry->Delete->Volumes** and select the two volumes, to be able to delete some of its unwanted surfaces.

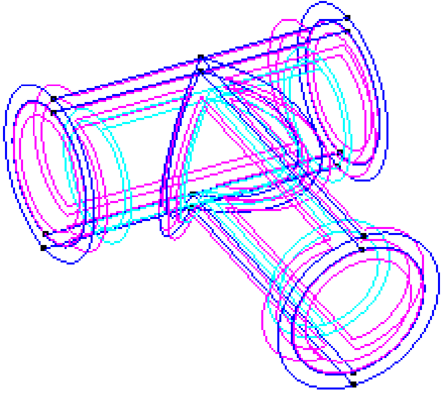
Note: verify that 'Also lower entities' of the delete menu is not set, otherwise its surfaces, lines and points will be deleted.

- Choose **Geometry->Delete->Surfaces** and select the small surfaces inside the first pipe. Select **Lower Entities** in the contextual menu, to delete its dependencies also. Press **ESC** to conclude the process of selection.



Delete the rest of unwanted surfaces to left only the skin of a volume.

- Use **Geometry->Create->Volume->By contour** and select all surfaces



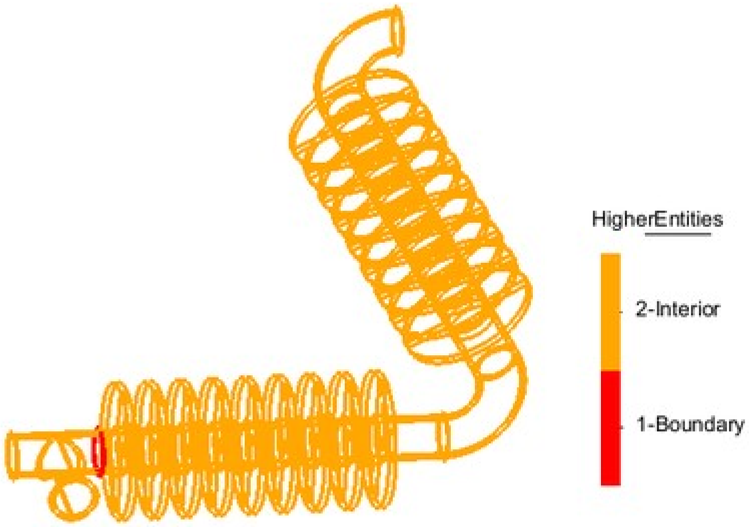
- From the **Files** menu, select **Save** to save the file. Enter a name for the file and click **Save**.

Importing the T junction to the main file

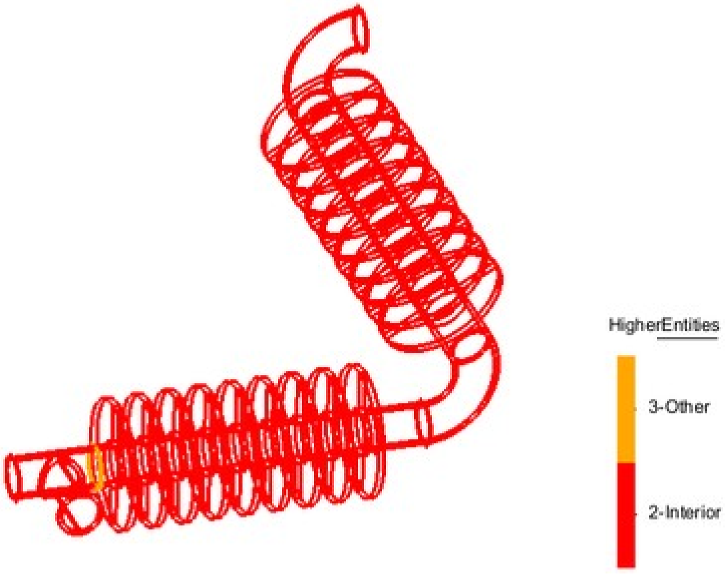
The two parts of the model have been drawn. Now they must be joined so that the final volume may be created and a mesh of the volume may be generated.

Importing a GiD file

- Choose **Open** from the **Files** menu. Select the file where the first part, created in section Creating a component part, was saved. Click **Open**.
- Choose **Files->Import->Insert GiD model** from the menu. Select the file where the second part, created in section Creating the T junction, was saved. Click **Open**.
- The T junction appears. Bear in mind that the lines which define the end of the first pipe (background) of the T junction, and which have been imported, were already present in the first file. Notice that the lines are duplicated. This overlapping will be remedied by collapsing the lines.
- Check duplicated lines with the tool **View->Higher entities->Lines**



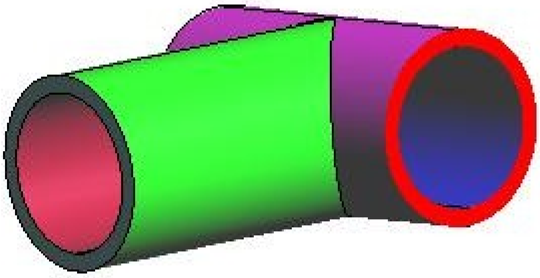
- Choose the option **Geometry->Edit->Collapse->Lines**. Select the overlapping lines and press **ESC**. It doesn't matter if do you select some other lines more, because they won't be collapsed with the small automatic tolerance.



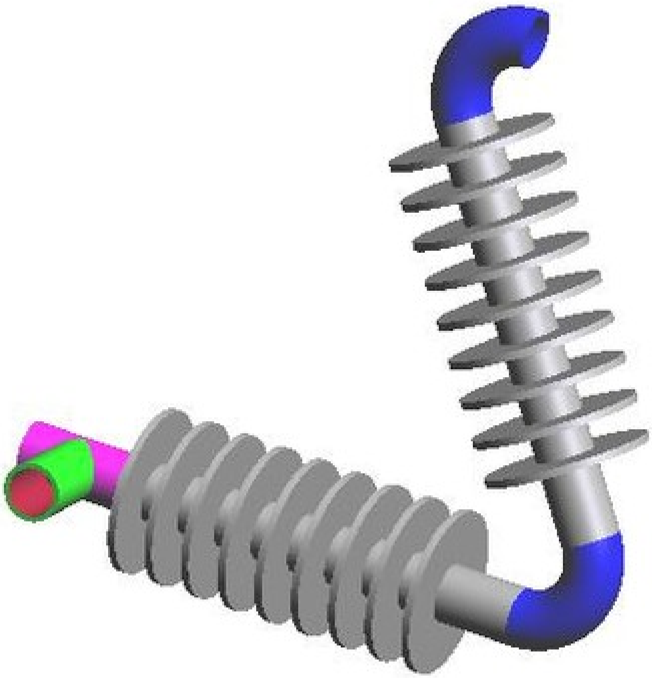
Creating the final volume

Now we have a volume of the T junction, and we want to create another volume with the rest of the piece, connected to the first volume. Two volumes are connected if they share some surface. we will reuse the ring surface of the first volume for the new volume.

- Set off the layers "part1" and "union"
- Send the shared surface to the "part1" layer, to facilitate the selection of the new volume boundary: select "part1", use **Send to->Surfaces** and select the surface of the image and press **Escape**



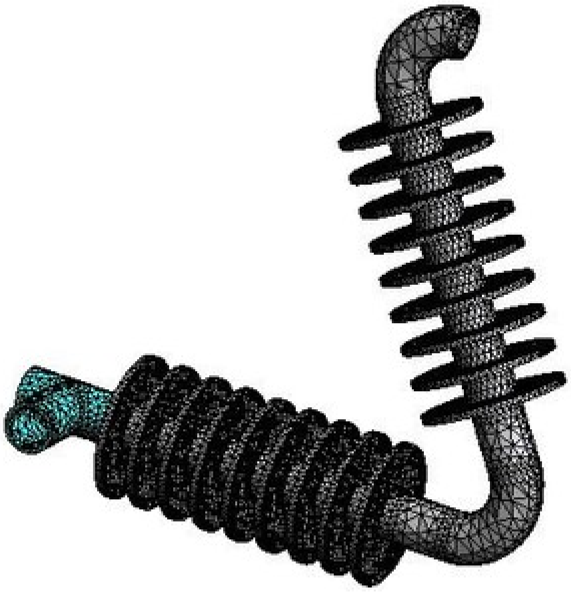
- Set off the layers "pipe1" and "pipe2" and set on "part1" and "union", and send "part1" as layer to use, to create the new volume in this layer.
- Choose **Geometry->Create->Volume->By contour** and select all the visible surfaces to define the volume. Press **ESC** to conclude the selection process.
- Choose **Render->Smooth** to visualize a more realistic version of the full model with all layers on.



Generating a mesh

Now that the model is finished, it is ready to be meshed.
Generate a coarse mesh is a good test to check that the model is correct and is valid to be used in a numerical simulation. We will use default meshing settings.

- Choose **Mesh->Generate mesh**.
- A window opens in which to enter the edge size of the mesh to be generated. Set a value of 5 units and click **OK**.
- When the meshing process is finished, a window appears with information about the mesh, press **View mesh** in order to be showed.



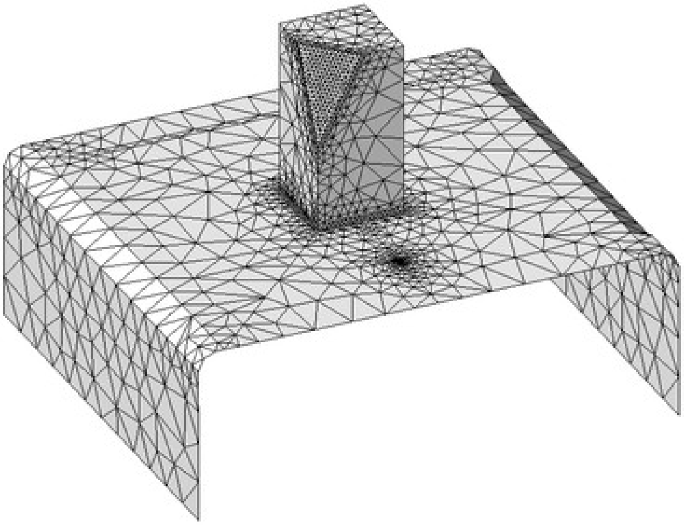
ASSIGNING MESH SIZES

ASSIGNING SIZES TO THE ELEMENTS OF A MESH

The objective of this example is to generate a body-fitted mesh of a mechanical piece using the various options in GiD for assigning sizes to elements, and the different surface meshers available. In this example a mesh is generated for each of the following methods for assigning sizes:

- Assigning sizes on points
- Assigning sizes on lines
- Assigning sizes on surfaces
- Assigning sizes with Chordal Error

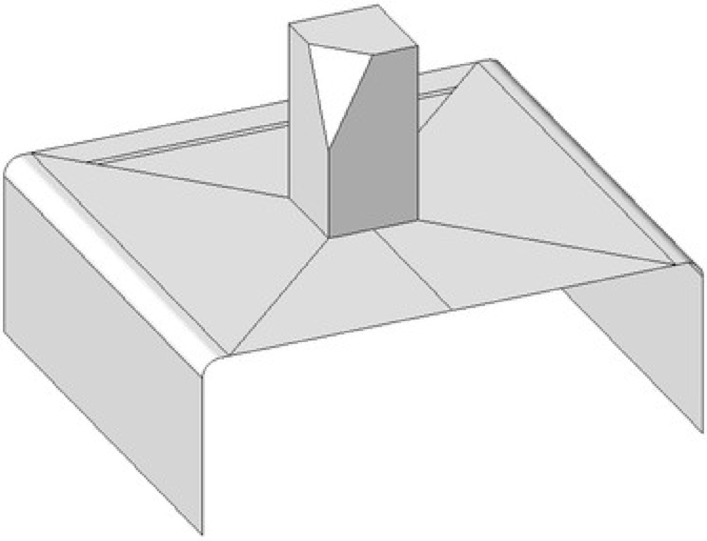
The use of different meshing options, such as skip entities when meshing, is also explored in this example.



Reading the initial project

In order to carry out this example, start by opening the project "ToMesh4.gid". This project contains a geometry that will be meshed using four different methods, each one resulting in a different density of elements in certain zones.

- Open the project "**ToMesh4.gid**".
- The geometry appears on the screen. It is a set of surfaces.
- Change the render mode (from the mouse menu, or from the status bar in the lower part of GiD window) and rotate the model in order to get a better perception of the geometry of the model.
- Finally, return to the normal visualization, selecting **Render->Normal**. This mode is more user-friendly.



The geometrical model of ToMesh4.gid project (view in render flat mode)

Element-size assignment methods

GiD automatically corrects element sizes according to the shape of the entity to be meshed and its surrounding entities. This default option may be changed by going to the **Utilities** menu, selecting **Preferences**, and then **Automatic correct sizes**[2] inside **Meshing - General** branch[3] .

Sometimes, however, this type of correction is not sufficient and it is necessary to indicate where on the mesh greater accuracy is needed. In these cases, GiD offers various options and methods to assign different mesh sizes.

To be sure that all the preferences are the ones used in this tutorial and get the same mesh as result, set the default values for all the preferences:

- To set as default all the meshing preferences, click on **Meshing** branch of the **Preferences** window and click on **Default values on selection**. This option is in the contextual menu (clicking on the left mouse button). Be sure the option '**Also selection children**' is checked, so the default values will be set to all the meshing options.

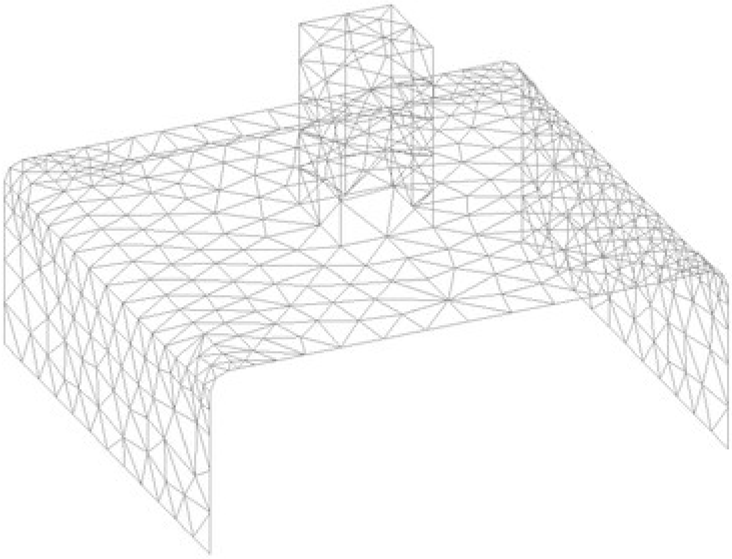
[2] The different options are: **None**, no size correction is made; **Normal**, a size correction is made according to the sizes of geometrical entities and the compatibility between meshing sizes of neighbouring entities; **Hard**, theNormal correction is made and, furthermore, an automatic chordal error criteria is applied to assign sizes to surfaces which are the contours of some volume in order to improve volume meshes.

[3] Similar options to **Automatic correct sizes** but to execute them manually one time can be found **on Mesh->Unstructured->Correct Sizes...**

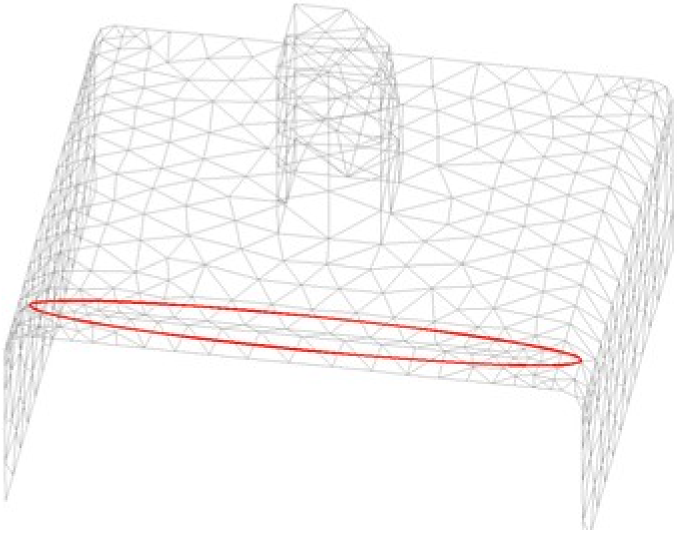
Assign general mesh size with default options

To generate the mesh using the default options:

- Select **Mesh->Generate mesh**. (or press <Ctrl>-g keys)
- A window appears asking the desired element edge size. This is the general mesh size that will be used for meshing the whole model. Leave this default size unaltered and click **OK**.
- A meshing process window opens. Then another window appears with information about the mesh generated. Click **View mesh** to visualize the mesh.



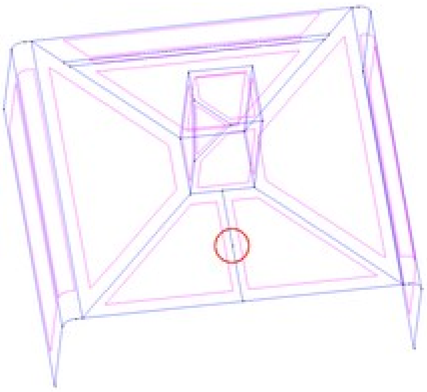
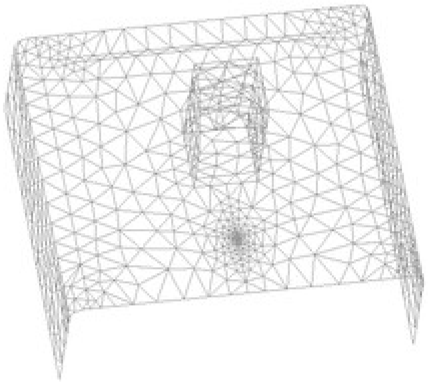
Note that in the zone highlighted in next figure, the elements are smaller than in the rest of the model. This is because of the shape of the surface placed there. When all meshing preferences are set by default, as for this example, the RFAST surface mesher is used. In this way, geometrical entities are meshed hierarchically: first of all lines are meshed, then the surfaces, and finally the volumes. The line elements size depends on the shape of surfaces (as can be seen in this example). Later on we will see an example of how to SKIP ENTITIES in the meshing process, leading to a different element sizes distribution.



Assign size to points

- Select **Mesh->Unstructured->Assign size on points**. A window appears in which to enter the element size around the points to be chosen. Enter **0.1** and click **Assign**.
- Select the point indicated in the figure. Press **<Esc>** [4] to indicate that the selection of points is finished, and **Close** the window.
- Select **Mesh->Generate mesh**.
- A window opens asking whether the previous mesh should be eliminated. Click **Yes**.
- GiD then asks you to enter the general maximum element size. Leave the default value unaltered and click **OK**.

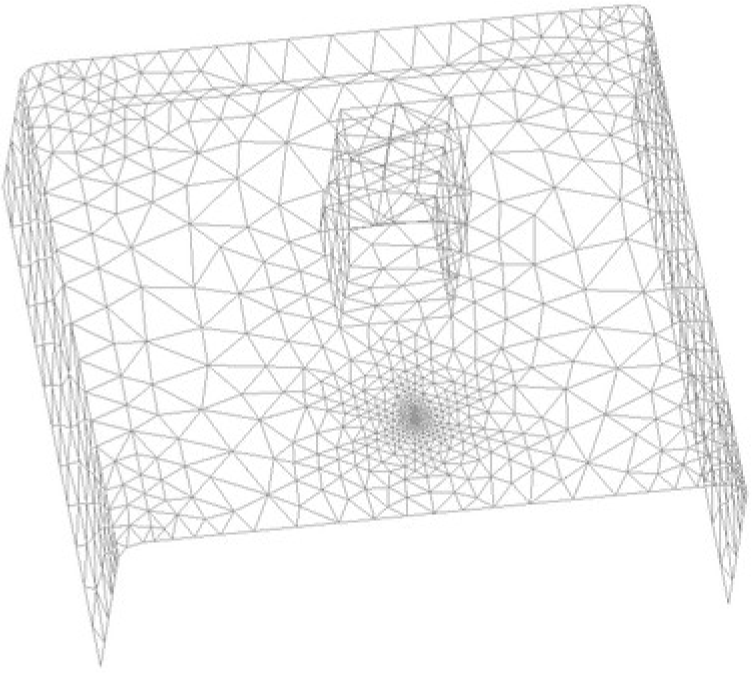
- Click **View mesh** in the pop up window to see the result.

 Geometry of the model. The point around which the mesh will be concentrated.	 The mesh with a concentration of elements around the point.

- A concentration of elements appears around the chosen point, given the selected size (0.1) of these elements.

One can control the way the size of the elements changes from a finer to a coarser region:

- Go to **Utilities** and open **Preferences** window. In the **Meshing - General** branch there is the option **Size transitions**. This option defines the transition gradient of element sizes (size gradient), whose values are between 0 and 1. The greater the size gradient is, the sharper change of sizes in space. To test this, enter the value **0.4** and click **Accept**.
- Again, generate the mesh (**Mesh->Generate mesh**) with the same general mesh size.



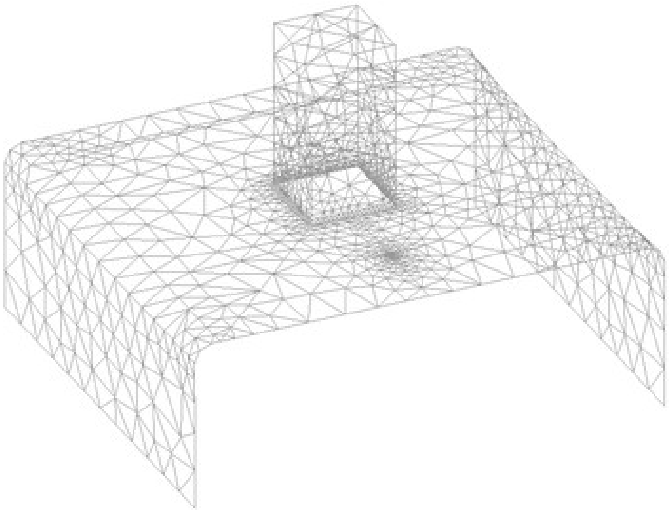
- The size gradient (0.4) results in a higher density around the point (see figure).
- Now go back and enter 0.6 in **Size transitions**. This will result in a mesh more suitable for our objectives. Click **Accept**.

[4] Instead of pressing the **<Esc>** key, the center mouse button or the mouse wheel can also be used.

Assign size to lines

We are going to set a specific mesh size to some of the lines of the model:

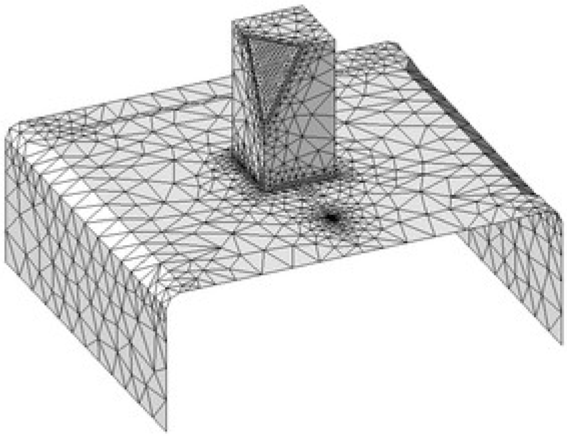
- To see entity numbers select the **Label** option (see Labels). Select all the lines in order to see their numbers.
- Select **Mesh->Unstructured->Assign sizes on lines**. In the window that appears, enter the size of the elements around the lines that will be chosen. Enter **0.5** and click **Assign**.
- Select the lines defining the base of the prism (i.e. lines 1, 2, 3, 4 and 40).
- Press <**Esc>** to indicate that the selection of lines is finished, and **Close** the window.
- Then generate the mesh again with the same general mesh size as before.
- This results in a high concentration of elements around the chosen lines, given that the selected element size (0.5) is smaller than that of the rest of the elements in the model.



Assign size to surfaces

Now we are going to set a specific mesh size to the triangular surface resulting from the section of one of the vertexes of the prism (surface number 1).

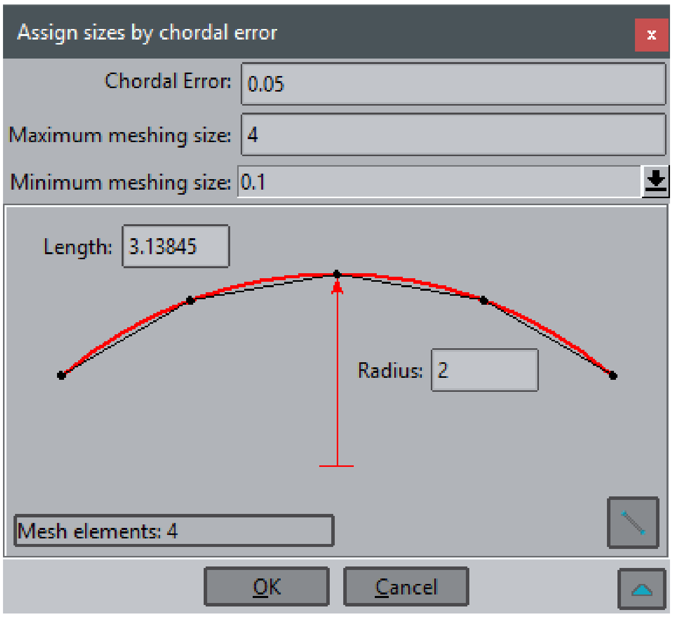
- To detect the surface one the Label option can be chosen, but also the Signal one (see Signal).
- Select **Mesh->Unstructured->Assign sizes on surfaces**. In the window that appears, enter the size of the elements to be assigned on the surfaces that will be chosen. Enter **0.5** and click **Assign**.
- Select surface number 1, press **<Esc>** and **Close** the window.
- Generate again the mesh with the same general mesh size.
- This results in a high concentration of elements on the chosen surface due to the value selected (0.5).



Assignment following chordal error criterion

In this section, an automatic size assignment is set taking into account a given maximum allowed chordal error. Note that using this options GiD will use the mean curvature of each entity in order to assign a mesh size to it. This size will be the same for the whole entity, so if there is a surface with zones with very different curvature, the mesh should not be adapted locally inside the entity.

- Select **Mesh->Unstructured->Sizes by chordal error…**
- The following window appears



Chordal Error: 0.05

Maximum meshing size: 4

Minimum meshing size: 0.1

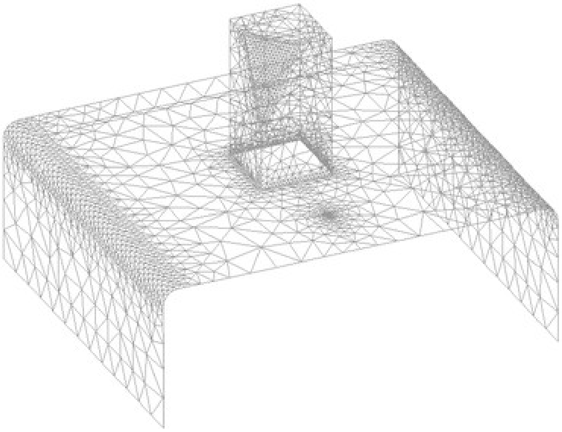
Length: 3.13845

Radius: 2

Mesh elements: 4

- Enter **0.05** as chordal error. This error is the maximum distance between the element generated and the real object (geometry).
- Enter **4** as maximum meshing size.
- Enter **0.1** as minimum meshing size.
- If the right bottom button is clicked some extra information appears. We can select a curved line and see how the mesh will look like in this line. Click on the button "line" icon and select the line 34 and change the chordal error value in order to see how the mesh would change.
- Press **OK**.
- To see the sizes that GiD has assigned automatically, you can select the **Draw->Sizes->All types** option in the **Mesh** menu.
- Generate the mesh with the same general mesh size.
- The resulting mesh has a high concentration of elements in curved areas. Now our approximation is significantly improved.

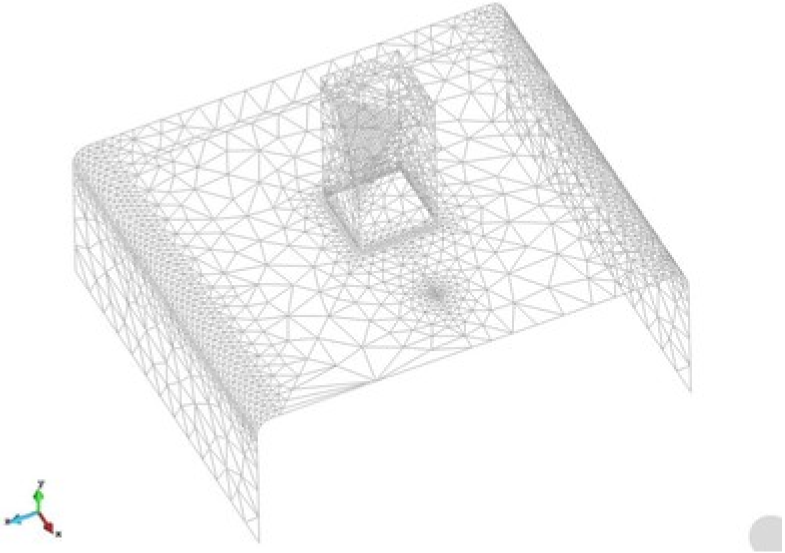
Note than the minimum or maximum mesh size is not a limit of the possible sizes of the elements to be generated, it is only a limit of the 'assigned wished sizes' (a line shorter than this minimum size will generate a short element)



Assign structured size

In this section, structured size is assigned to a line. When assigning structured sizes, the exact number of divisions is set, instead of the size of the elements.

- Select **Mesh->Structured->Lines->Assign number of divisions.**
- Enter **2 number of divisions** in the appearing window, and assign it to the line number 41.
- Press **<Esc>** to indicate that the selection of lines is finished, and **Close** the window.
- You can check graphically the number of structured divisions assigned to the entities using the **Mesh->Draw->Num of divisions**.
- Then generate the mesh again with the same general mesh size as before.
- This resulting mesh has only 2 linear elements in line 41, as it can be seen in the figure.



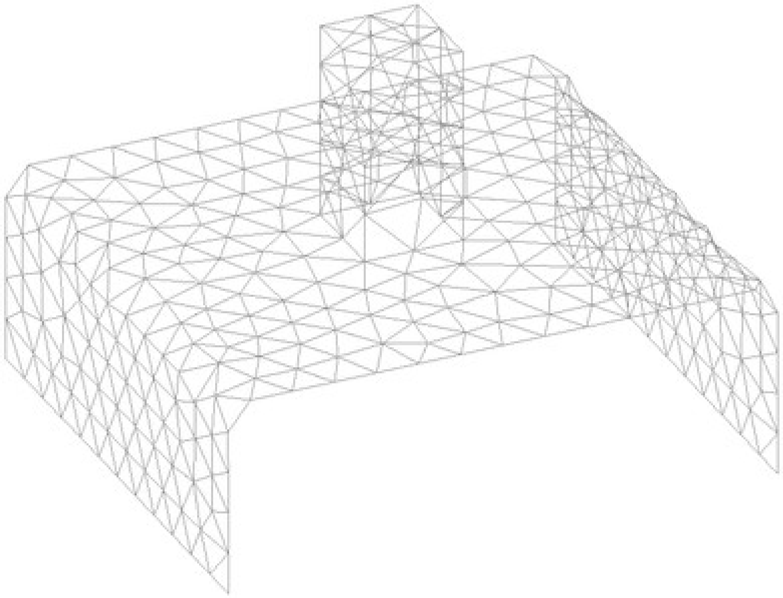
Skip entities

GiD allows to skip entities when meshing, so as patches of surface meshes can be meshed together skipping their inner lines (points connecting lines can be skipped as well, looking at the lines meshes). There are two ways of skipping entities: let GiD decide which are the entities tangent enough to be meshed together, or set manually the entities to be skipped or not.

By selecting **Mesh->Draw->Skip entities**, the entities that are going to be skipped and the ones that will have mesh with the current meshing preferences are displayed in different colors. In this chapter we will see how to skip entities and force some of them not to be skipped.

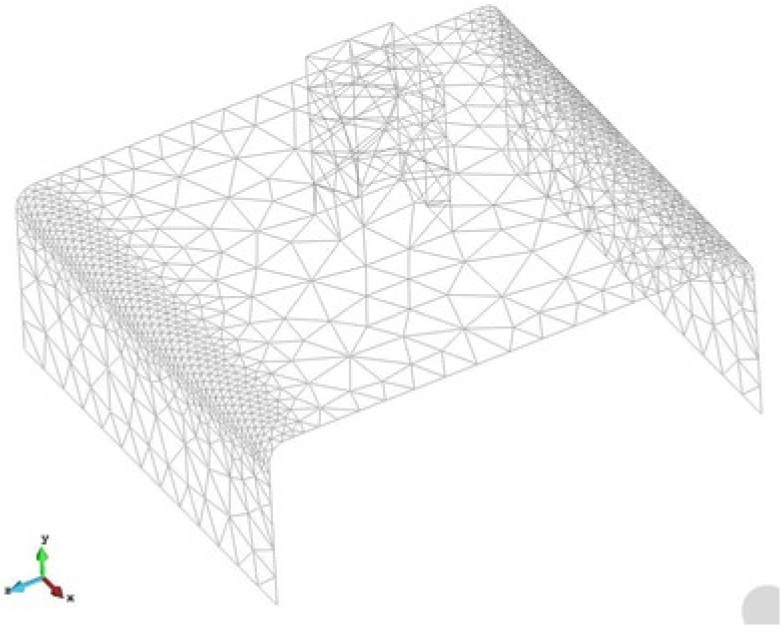
Skip entities automatically

- Select **Mesh->Reset mesh data** to reset all mesh sizes introduced previously.
- A window appears advising that all the mesh information is going to be erased. Press **Ok**.
- Set the **Default Values** in the **Meshing** branches of the **Preferences** window.
- Set the **Skip entities automatically** option (Preferences-Meshing-Structuration type) and click **Apply**. It can be appreciated the limit tangency angle by default is 10 degrees, and the layers the entities belongs to are not considered. This parameter can be set by the user.
- Generate the mesh with the default general mesh size.
- In the generated mesh, the contact lines between surfaces that are tangent enough do not have nodes; contact points between lines tangent enough are also skipped when meshing.



Note that the size of the elements is more uniform than the ones of previous meshes.
Using the option of skipping entities it is also possible to assign sizes to different entities. As an example, select **Mesh->Unstructured->Sizes by chordal error…**.

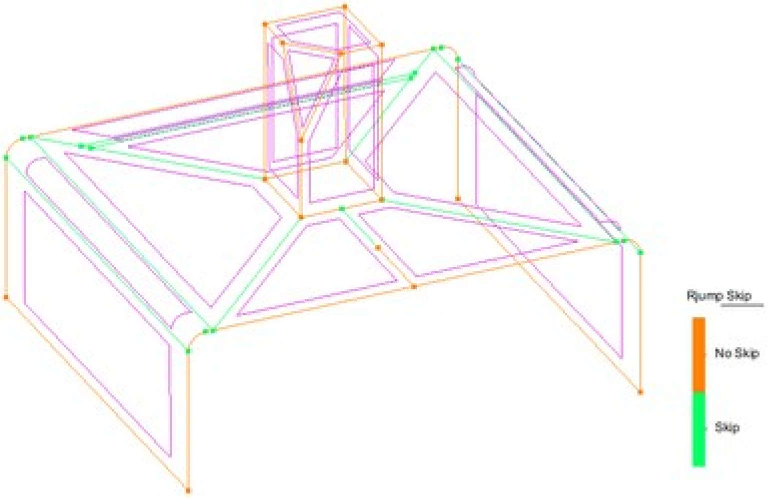
- In the window enter **0.05** as chordal error.
- Enter **10** as maximum meshing size and **0.1** as minimum meshing size.
- Press **OK**.
- Again, generate the mesh with the same default general mesh size.
- This results in a high concentration of elements in curved areas, without the nodes in the lines and points that mesher skips. Now our approximation is significantly improved.



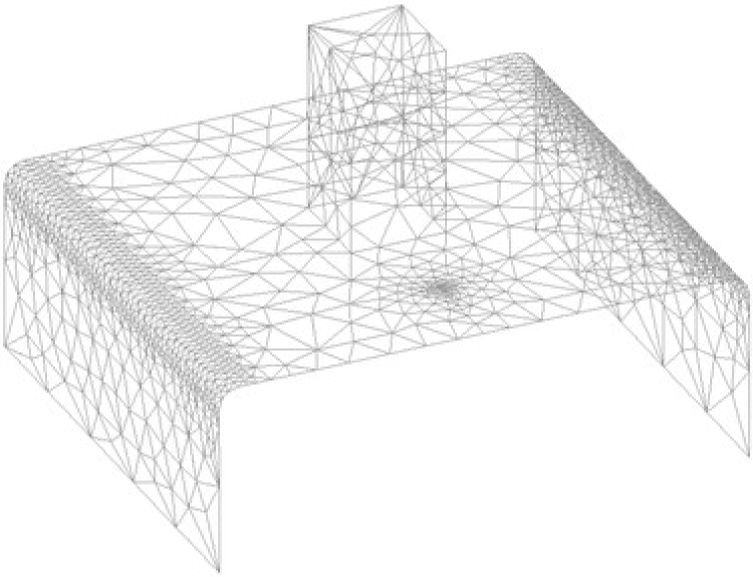
Force to mesh some entity

If there is a line or a point that the automatically skip options would skip, but it is required for you to be meshed, you can specify the entity to not be skipped. As an example, we will force to mesh the line number 43, in order to concentrate elements around point number 29, as it was done in section Assign size to points example.

- Select **Mesh->Mesh criteria->No skip->Lines**, and select line number 43. Press **ESC**.
- Select **Mesh->Draw->Skip entities** to display the entities that will and will not be skipped in different colors. As is shown in next figure, the line 43 will now not be skipped; the rest of the lines are unaffected, and GiD will either skip or mesh them according to the automatic criteria set in the preferences window.



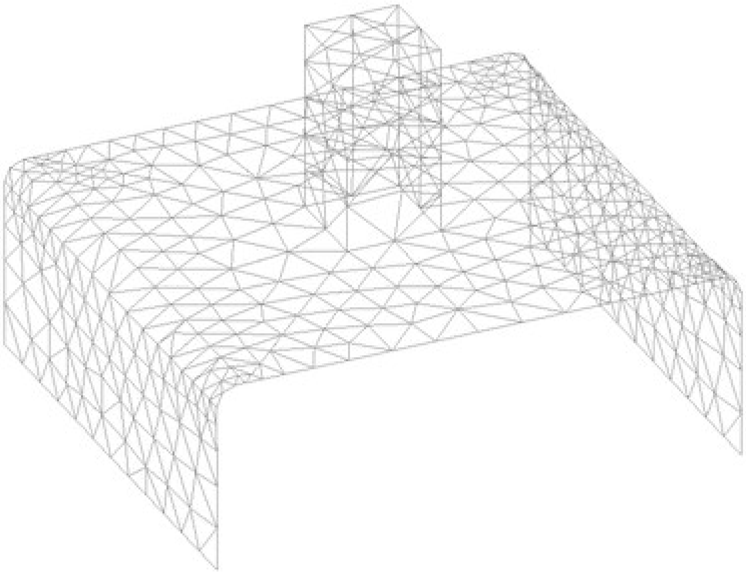
- Select **Mesh->Unstructured->Assign sizes on points**. A window appears in which to enter the element size around the points to be chosen. Enter 0.1 and click **Assign**.
- Select the point number 29. Press **<Esc>** to indicate that the selection of points is finished and **Close** the window.
- Generate the mesh again with the default general mesh size.
- The resulting mesh is depicted. A high concentration of elements around point number 29 can be appreciated. Note that there are nodes on line number 43 because we have forced not to skip this line.



In this last example we have forced the mesher not to skip an entity, but it may be interesting in some models to allow the mesher only to skip a few entities, meshing almost all of them.
Unsetting the **Skip entities automatically** option in the preferences window, all the lines and points will be meshed except the ones set explicitly to be skipped using the **Mesh->Mesh criteria->Skip** option. In this case, no entity will be skipped automatically according to tangency with neighboring entities.
The next example shows how to work like this.

- Select **Mesh->Reset mesh data** to reset all mesh sizes introduced previously. (A window opens advising that all the mesh information is going to be erased. Press **Ok**.).

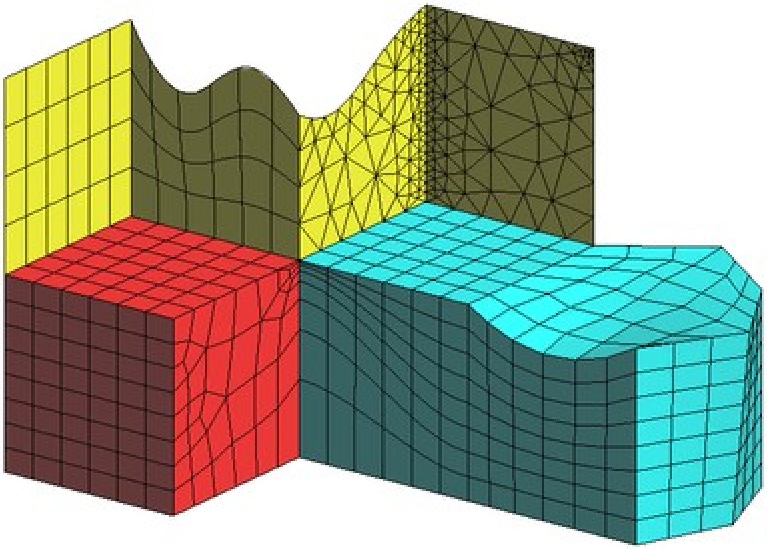
- Select **Mesh->Mesh criteria->Skip->Lines**, and select lines 48 and 53. Press **<Esc>**.
- Generate the mesh with the default general mesh size.
- The result is a mesh similar to the first test, but the smaller elements near the lines 48 and 53 do not appear because these lines are now skipped when meshing.



METHODS FOR MESH GENERATION

The objective of this example is to mesh a model using the various options available in GiD for body-fitted meshes controlling the element type in structured, semi-structured and unstructured meshes. It also presents how to concentrate elements and control the distribution of mesh sizes, as well as the generation of embedded and cartesian meshes.
The methods covered are:

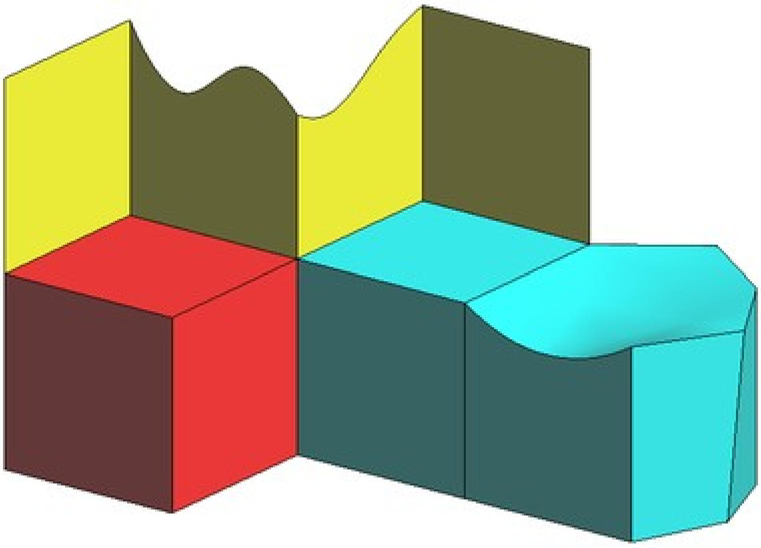
- Generating a mesh using tetrahedral
- Generating a volume mesh using spheres
- Generate a mesh using circles
- Generating a volume mesh using points
- Generating a mesh using quadrilaterals
- Generating a structured mesh on surfaces and volumes
- Generating a semi-structured volume mesh
- Generating a mesh using quadratic elements
- Generating a mesh with minimum number of elements
- Generating cartesian meshes
- Generating embedded meshes



Reading the initial project

In order to carry out this example, start from the project "ToMesh3.gid". This project contains a geometry that will be meshed using different types of elements and meshers.
For the embedded mesh, the project "gid_embedded_example.gid" is used.

- In the **Files** menu, select **Read**. Select the project "ToMesh3.gid" and click **Open**.
- The geometry appears on the screen. It is a set of surfaces and three volumes. Select **Render->Flat** from the mouse menu or from the **View** menu. In Figure 1 the geometrical model is shown.
- Rotate and make several changes in the perspective so as to get a good idea of the geometry involved.



Finally, return to the normal visualization **Render->Normal**. This mode is more user-friendly.

Types of body-fitted mesh

Using GiD the mesh may be generated in different ways, depending on the needs of each project. The two basic types of body-fitted meshes are the structured[2] and the unstructured mesh. For volumes only there is one additional type, the semi-structured[3] mesh.
For all these types of mesh a variety of elements may be used (line ones, triangles, quadrilaterals, circles, tetrahedra, hexahedra, prisms, spheres or points). In this tutorial you will become familiarized with the mesh-generating combinations available in GiD.

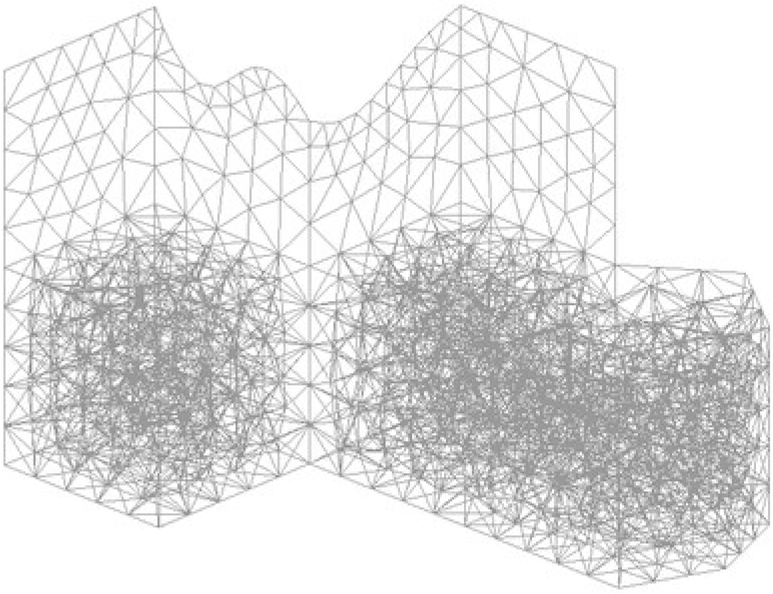
[2] A structured mesh is one in which each internal node is connected to a constant number of elements.

[3] A semi-structured volume mesh is one in which you can distinguish a fixed structure in one direction, i.e. there is a fixed number of divisions. However, within each division the mesh may not be structured. This kind of mesh is only possible for topologically prismatic volumes.

Generating the mesh by default

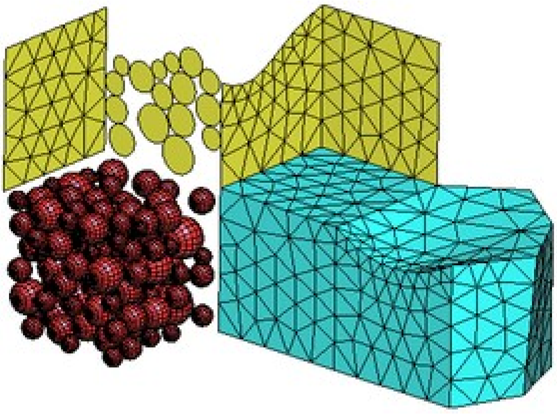
In order to get the same results we will reset the mesh options.

- Open the preferences window selecting **Utilities->Preferences.**
- Select the **Meshing** branch, click on **Default** values and then **Apply**.
- Now select **Mesh->Reset mesh data** in order to ensure the geometry has no meshing property attached.
- Select **Mesh->Generate mesh**.
- A window comes up in which to enter the desired 2element size for the mesh to be generated. As default value could change from one version of GiD to another, insert **2** to get the same results as shown in images **OK**.
- A meshing process window comes up. Then another window appears with information about the mesh generated. Click **View mesh** to visualize the mesh.
- The result is the mesh in Figure. There are various surfaces and volumes. By default, mesh generation in GiD obtains unstructured meshes of triangles on surfaces and tetrahedra on volumes.



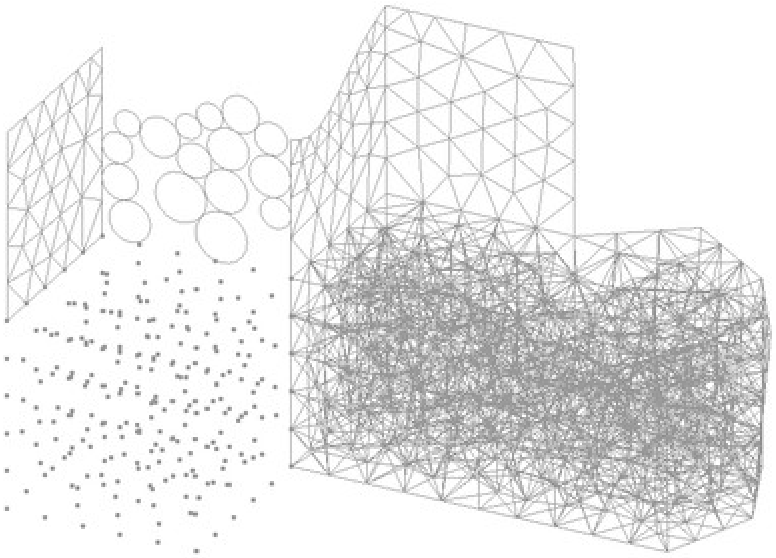
Generating the mesh using circles and spheres

- Select **Mesh->Element type->Sphere**. Select volume number 1 and press **ESC.** To see entity numbers select **Label** from the mouse menu or from the **View** menu. If you wish the geometrical entity labels to be displayed, the view mode needs to be changed to Geometry using **View->Mode->Geometry** (this option may also be found in the GiD Toolbar). Select **Render->Normal** in the mouse menu to see the labels.
- Select **Mesh->Element type->Circle**. Select surface number 24 and press **ESC**.
- Select **Mesh->Generate mesh**.
- A window comes up asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which to enter the desired element size. Leave the default value unaltered and click **OK**. The result is a mesh as illustrated.



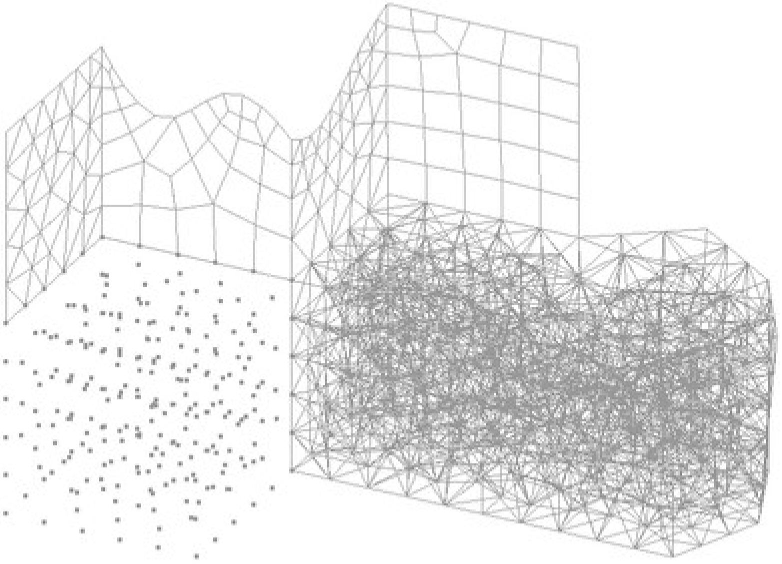
Generating the mesh using points

- Select **Mesh->Element type->Points**. Select volume number 1 and press **ESC.**
- Select **Mesh->Generate mesh**.
- A window comes up asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which to enter the desired element size. Leave the default value unaltered and click **OK**. The result is a mesh as illustrated, where volume number 1 is meshed using 'point elements' with only 1-node. (Note that in render flat or smooth visualization mode the nodes may not appear in the screen depending on visualization properties. Set render normal mode to see them).



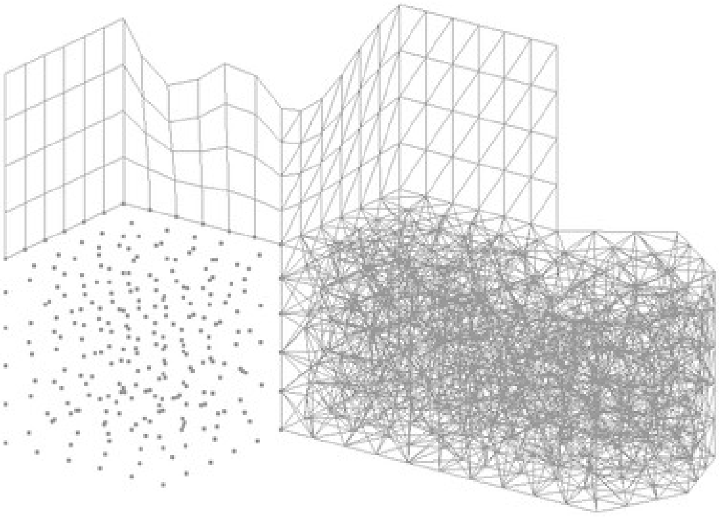
Generating the mesh using quadrilaterals

- Select **Mesh->Element type->Quadrilateral**. Select surfaces number 24 and 12 and press **ESC**.
- Select **Mesh->Generate mesh**.
- A window comes up asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which the desired element size can be entered. Leave the default value unaltered and click **OK**. The result will be the mesh illustrated, where it can be appreciated the quadrilateral mesh of both surfaces.

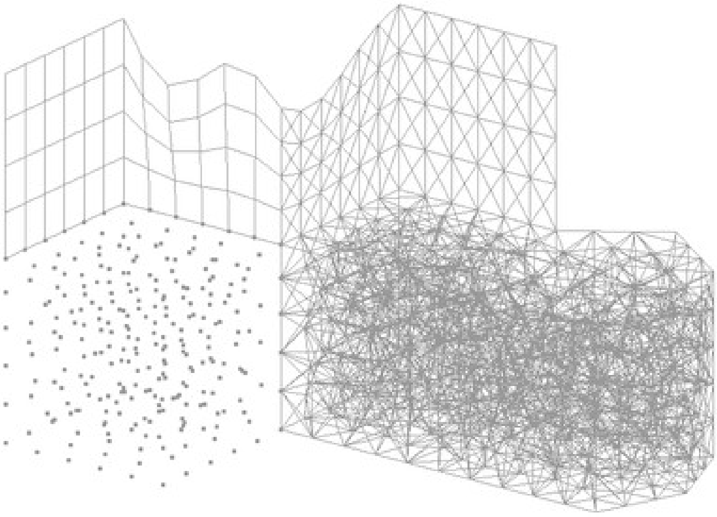


Generating a structured mesh (surfaces)

- To mesh surfaces with a structured mesh, select the option **Mesh->Structured->Surfaces->Assign number of divisions to surface lines**.
- Select all top surfaces 9, 24, 26 and 12 and press **ESC**.
- A window appears asking the number of divisions that the lines to be selected will have. Enter4.
- Click **Assign** and select one vertical line[5] (parallel to the **Y** axis). Press **ESC**.
- Another window appears in which to enter the number of divisions on the lines. Enter 6.
- Click **Assign** and select the 4 bottom lines[5]. Press **ESC**.
- Another window appears in which to enter the number of divisions on the lines. In this case, all the boundary lines have already been defined. Therefore, click **Close**.
- Select **Mesh->Element type->Triangle**. Select surfaces 26 and 12. Press **ESC**.
- Select **Mesh->Generate mesh**.
- A window comes up asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which to enter the desired element size. Leave the default value unaltered and click **OK**. The result is the mesh shown.



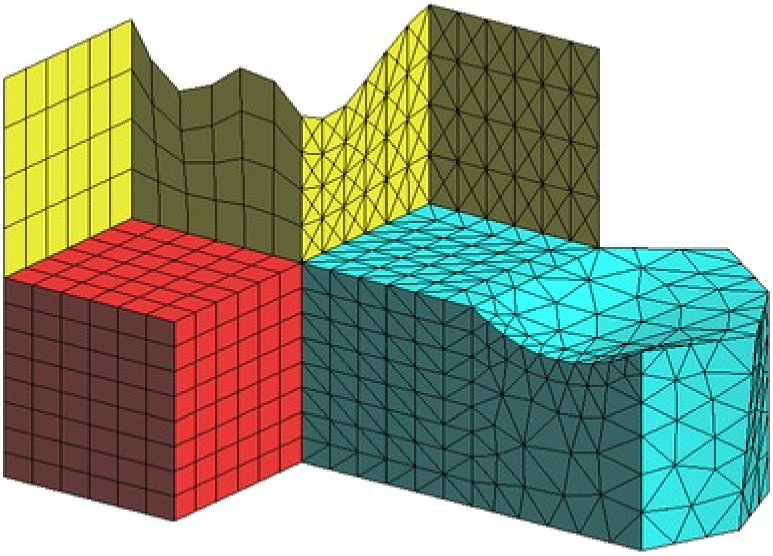
- As seen in Figures, GiD can obtain surface structured meshes made of quadrilaterals or triangles. There are two kinds of structured mesh that use triangles: the one shown in Figure 7 is obtained when the **Utilities->Preferences->Meshing->Structuration type->Symmetrical structured->triangles** option is set. If this option is not set, the mesh presented in the previous figure is produced (with symmetrical structured extra center nodes are created).



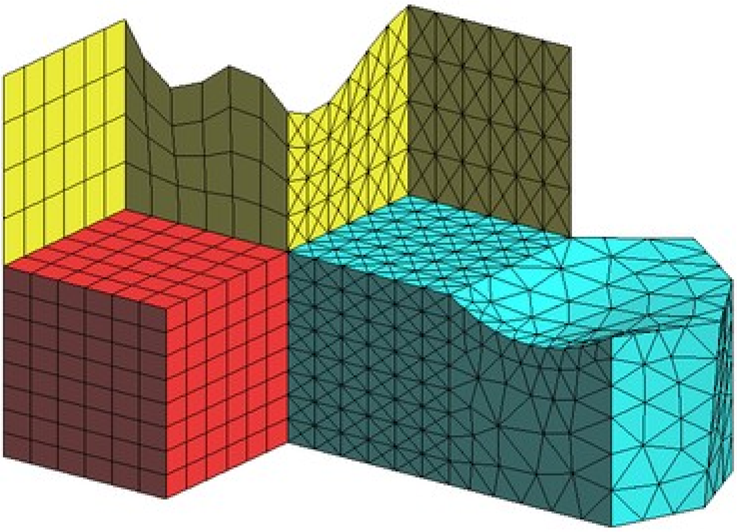
[5] When selecting a line, GID automatically selects all lines 'topologically parallel' to it.

Generating structured meshes (volumes)

- To mesh volumes with a structured mesh, select the option **Mesh->Structured->Volumes->Assign number of divisions to volume lines**.
- Select volumes 1 and 2 and press **ESC**.
- A window appears in which to enter the number of divisions that the lines to be selected will have. Enter 6 and click **Assign**.
- Select lines of both volumes parallel to the **X** and **Z** axes. GiD automatically selects all the lines in each volume parallel to these in order to create the structured mesh. Press **ESC**.
- Another window appears in which to enter the number of divisions on the lines. Divide the lines parallel to the **Y** axis into 8 segments. Enter 8 and click **Assign**.
- Select an edge of volume 1 or 2 parallel to the **Y** axis and press **ESC** . Again, the line-division window comes up. Since we have already finished the assignments, click **Close**.
- For structured volumes, GiD generates tetrahedron meshes by default, but hexahedron structured meshes can also be assigned. Let's assign the element type that we wish to volume 1 and 2. Select **Mesh->Element type->Tetrahedra**, then select volume number 2 and press **ESC**. Select **Mesh->Element type->Hexahedra**, then select volume number 1 and press **ESC**.
- Select **Mesh->Generate mesh**.
- A window appears asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window comes up in which to enter the desired element size. Leave the default value unaltered and click **OK** . The result is the mesh shown.

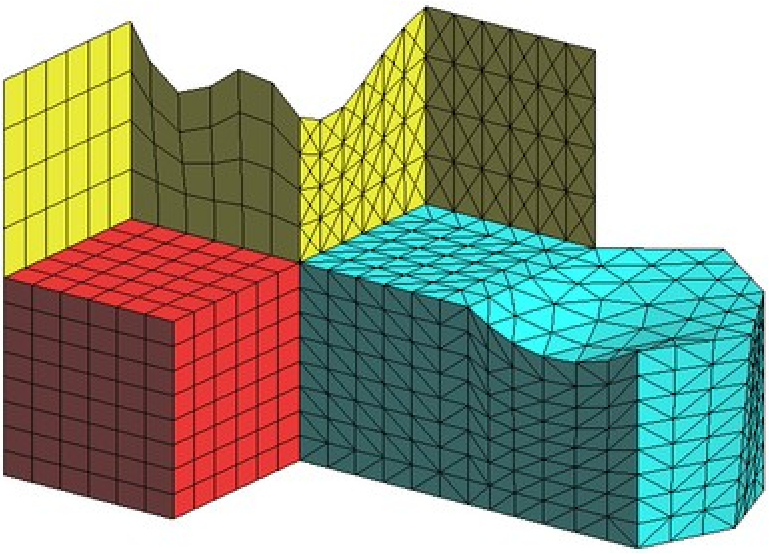


- GiD can obtain volume structured meshes made of hexahedra, tetrahedra or prisms. As can be seen in the figures, there are two kinds of tetrahedron structured mesh: the next one shown is obtained when the option **Utilities->Preferences->Meshing->Structuration type->Symmetrical structured->tetrahedra** is set. If this option is not set, the mesh presented in previous figure is produced (with fewer nodes).



Generating semi-structured meshes (volumes)

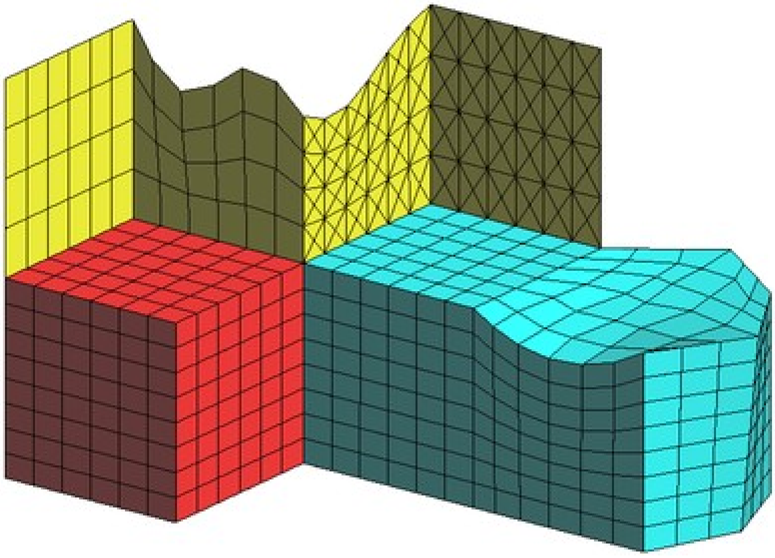
- To mesh volumes with a semi-structured mesh, select the option **Mesh->Semi-structured->Volumes->Assign number of divisions in structured direction**.
- A window appears8. Enter 8 and click **Assign**.
- Select volume 3 and press **ESC**. As volume 3 is prismatic in one direction only (i.e. parallel to **Y** axis) GiD will automatically detect this fact and will select it to be the direction in which the semi-structured volume mesh is structured.
- Another window appears in which to enter the number of divisions in the direction of the structure. In this case we do not want to select any more volumes, so click **Close**.
- Select **Mesh->Generate mesh**.
- A window appears asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which to enter the desired element size. Leave the default value unaltered and click **OK**. The result is the mesh shown.



As can be seen, volume 3 has been meshed with tetrahedra. Semi-structured volumes can be meshed with tetrahedra, prisms or hexahedra, if the compatibility with the element types of adjacent entities allows it.

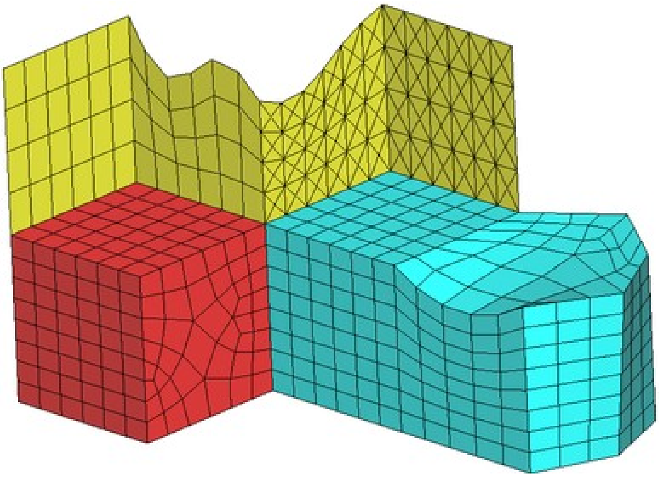
In the following steps a hexahedron mesh is produced.

- Select **Mesh->Element type->Hexahedra.**
- Select volumes 2 and 3 and press **ESC**.
- Select **Mesh->Generate mesh**.
- A window opens asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which to enter the desired element size. Leave the default value unaltered and click **OK**. The result is the mesh shown in next figure.



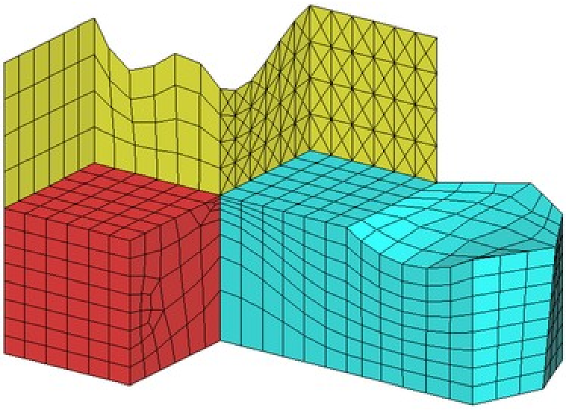
In case of volume number 3 there is only one direction in which it can possibly be structured (i.e. in the direction of the prism). If the volume is prismatic in more than one direction, there are two ways to choose between them: selecting one top surface (**Mesh->Semi-structured->Volumes->Set master surface**) or the direction of the structure (**Mesh->Semi-structured->Volumes->Set structured direction**). The following example explains this procedure.

- Select the option **Mesh->Semi-structured->Volumes->Assign number of divisions in structureddirection**
- A window opens in which to enter the number of divisions in the structured direction (prismatic). Enter 6.
- Select volume 1 and press **ESC**.
- Another window appears, click **Close**.
- Select **Mesh->Semi-structured->Volumes->Set structured direction**.
- Select one line parallel to the **X** axis of volume number 1 (for example line number 11) and press **ESC.**
- Select **Mesh->Unstructured->Assign entities->Surfaces** and select surfaces 1 and 6 and press **ESC**.
- Select **Mesh->Generate mesh**.
- A window opens asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which to enter the desired element size. Leave the default value unaltered and click **OK**. It can be appreciated in the following figure that the mesh in volume 1 is semi-structured, extruded along the x direction, now with an unstructured quadrilateral mesh on the surfaces 1 and 6.



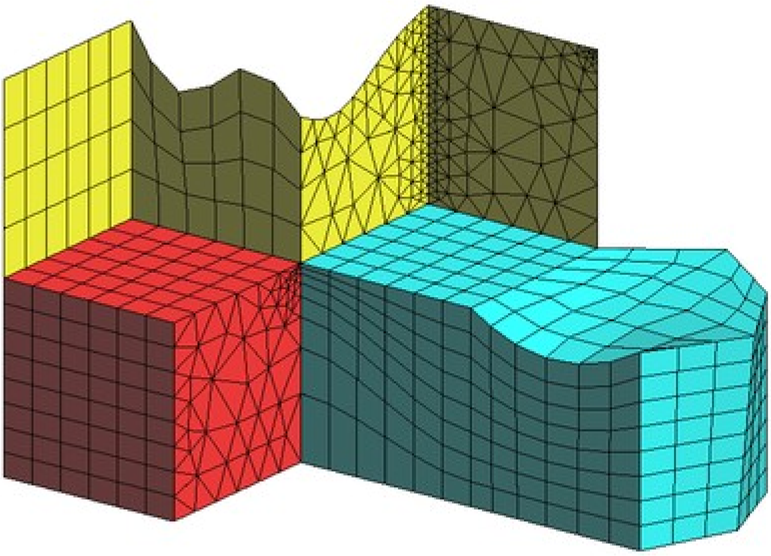
Concentrating elements and assigning sizes

- Select **Mesh->Structured->Lines->Concentrate elements**.
- Select some structured lines, for example line 43. Press **ESC**.
- A window comes up in which to enter two values for the concentration of elements. Positive values concentrate the elements and negative values spread them. Enter 0.5 as **Start Weight** and –0.5 as **End Weight**[10]. Click **Ok** and press **ESC**.
- Select **Mesh->Generate mesh**.
- A window opens asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which to enter the desired element size. Leave the default value unaltered. The result is the mesh shown.



It is also possible to assign sizes to geometrical entities, so that mesh elements can be concentrated in certain zones. In the following steps some examples are given.

- Select **Mesh->Unstructured->Assign sizes on points.**
- A window appears in which to enter the size to be assigned to points. Enter 0.1 and click **Assign**.
- Select point number 15 and press **ESC**.
- Another window appears in which to enter the size to be assigned to points. In this case, we do not want to assign sizes to any other points, so click **Close**.
- Select **Mesh->Unstructured->Assign sizes on lines.**
- A window appears in which to enter the size to be assigned to lines. Enter 0.5 and click **Assign**.
- Select line number 25 and press **ESC**.
- Another window appears in which to enter the size to be assigned to lines. In this case, we do not want to assign sizes to any more lines, so click **Close**.
- Select **Mesh->Generate mesh**.
- A window appears asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which the desired element size should be entered. Leave the default value unaltered and click **OK**. The result is not the desired, we just get the previous mesh. This is because surrounding surfaces and lines are structured, so they do not have enough freedom to achieve the given sizes.
- Select **Mesh->Unstructured->Assign entities->Surfaces.**
- Select Surfaces 26 and 12. Press **ESC**.
- Select **Mesh->Unstructured->Assign entities->Lines.**
- Select lines 48, 26 and 27. Press **ESC**.
- Select **Mesh->Generate mesh**.
- A window appears asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which the desired element size should be entered. Leave the default value unaltered and click **OK**. The result is the mesh shown.

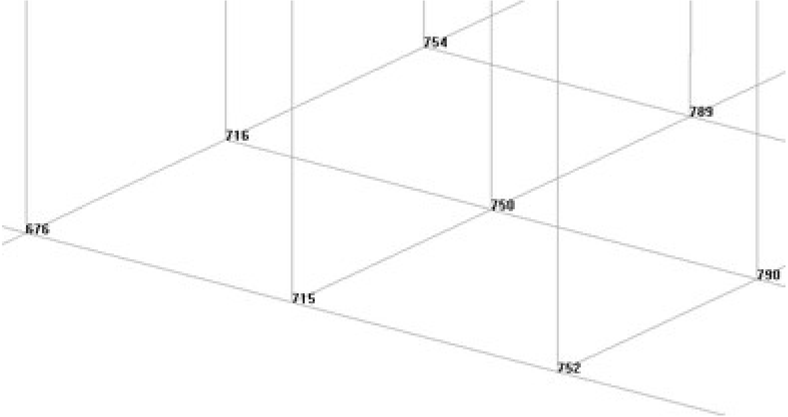


[10] Start Weight and End Weight refer to the start point and end point of the line, oriented as it is drawn with an arrow in light blue when you select it.

Generating the mesh using quadratic elements

Enlarge one area of the mesh with the zoom.

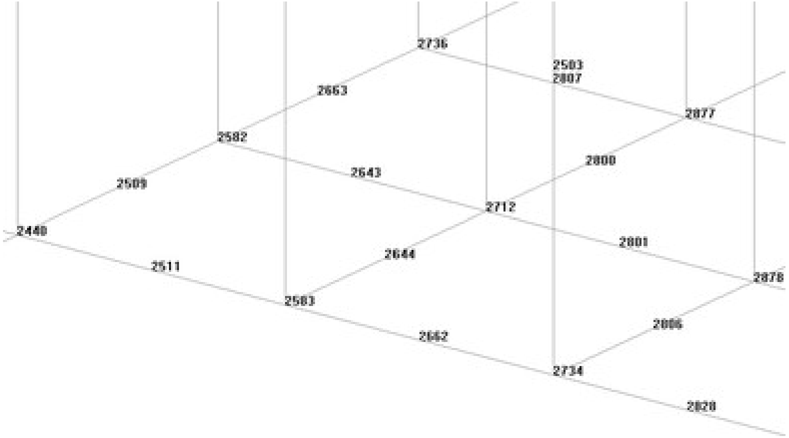
- Select **Label->All on in->Nodes .** The result is shown.



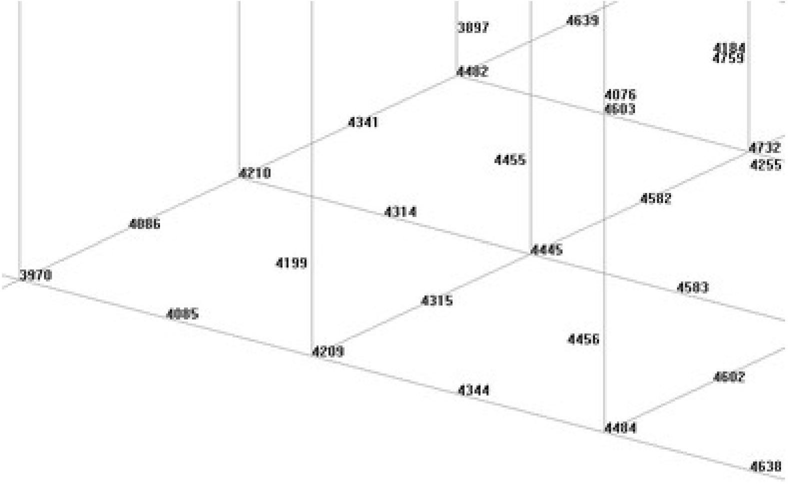
- The node identifiers created by generating the mesh appear on the screen. There is one identifier for each vertex of each element.
- Select **Mesh->Quadratic type->Quadratic.**

NOTE : By default GiD meshes with polynomial degree=1 (linear) elements. To find out which mode GiD is working in, go to **Mesh->Quadratic type**.

- Select **Mesh->Generate mesh**.
- A window opens asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which the desired element size should be entered. Leave the default value unaltered and click **OK**.
- Once the mesh has been generated, select **Label->All on in->Nodes**. The result is shown in next figure. Now, there are not only nodes at the vertices, but also at the midpoints of the edges of the elements.



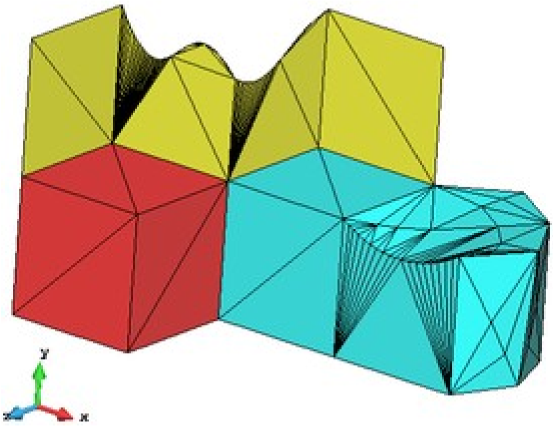
- Select **Mesh->Quadratic type->Quadratic9**.
- Select **Mesh->Generate mesh**.
- A window opens asking whether the previous mesh should be eliminated. Click **Yes**.
- Another window appears in which the desired element size should be entered. Leave the default value unaltered and click **OK**.
- Select **Label->All on in->Nodes** (see next figure).
- Notice that the four-sided elements (quadrilaterals) also have a node in the center, in addition to the nodes at the vertices and midpoints of the edges. Similarly, hexahedra also have a node at their center point.



Generating minimum elements meshes (surfaces)

The MinElem unstructured surface mesher generate triangular meshes considering just the chordal error parameters defined in the Meshing branch of the preferences window. This implies the triangles may not have a good aspect ratio, but the mesh represents accurately the shape of the surfaces with as less elements as possible.
This kind of meshes are often used for visualization purposes, where the angular quality of the triangles is not so relevant.

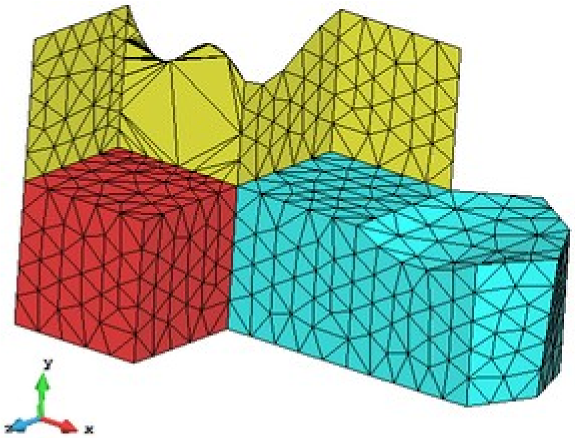
1. First of all, reset the mesh properties already assigned in previous steps of the tutorial: select **Mesh->Reset mesh data**.
2. Select **MinElem** as the unstructured surface mesher in the **Meshing->Structuration type** branch of the **Preferences** window, and set the **Default** options in the **Chordal error** branch of it. Set to **None** the Automatic correct sizes parameter of **Meshing->General** branch in order to see the result of this mesher in a more exaggerated way. (Remember to click **Apply** in the preferences window in order to set the parameters.)
3. Then, click on **Mesh->Generate** to generate the mesh and set 1000 as general mesh size. The resulting mesh is depicted. It can be appreciated that the straight lines are meshed with only one element, and the planar surfaces present a mesh only connecting their contour elements. This is due to the fact that they have no curvature, so the chordal error of their elements is always zero, and there is no need to refine the mesh there in order to represent the shape of the geometry.



Assigning different unstructured meshers (surfaces)

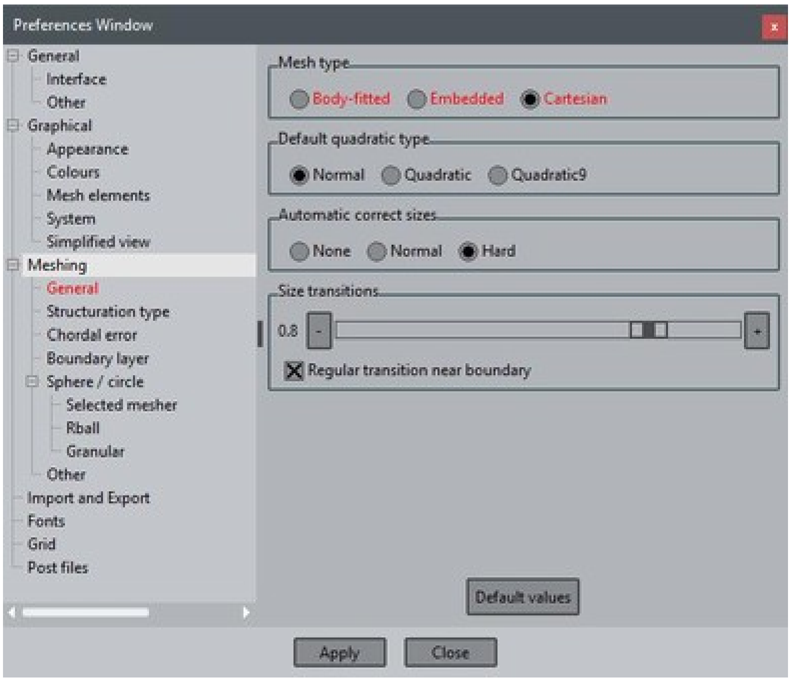
As it can be appreciated, the meshes generated using MinElem mesher are rather special. It is common to use them for visualization purposes, or to simulate rigid body kinematics in some kind of simulations. Using GiD, the user can assign different meshers to different entities:

1. Set the **Default** values in the **Meshing** branch of the **Preferences** window (note that the **RFast** surface mesher is set).
2. Assign the **MinElem** mesher to the surface number 24 by setting **Mesh->UnstructuredSurface mesher->MinElem** and selecting surface number 24.
3. Assign an unstructured size of 1000 to surface number 24 using **Mesh->Unstructured->Assign sizes on surfaces**.
4. Generate mesh with a general size of 2. The resulting mesh should be similar to the one depicted.



Cartesian mesh

To swap to the mesh of cartesian type is necessary to select in
Utilities->Preferences...
Meshing->General: Cartesian



Mesh type: ○ Body-fitted ○ Embedded ● Cartesian

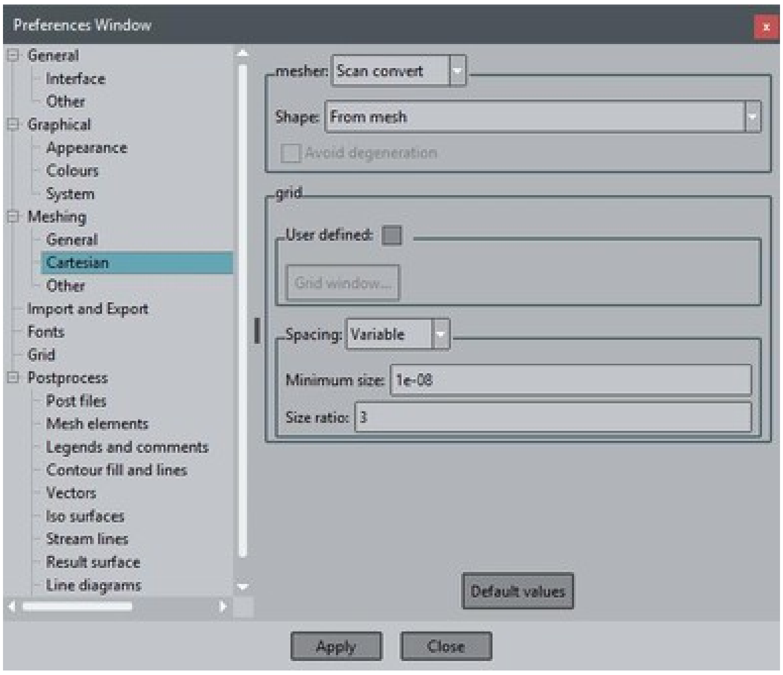
Default quadratic type: ● Normal ○ Quadratic ○ Quadratic9

Automatic correct sizes: ○ None ○ Normal ● Hard

Size transitions: 0.8

☒ Regular transition near boundary

Changing the kind of mesh type some other branch of Meshing could change: once applied the change it will appear a new 'cartesian' branch, and other options that does not have sense in this case will disappear.



mesher: Scan convert

Shape: From mesh

□ Avoid degeneration

grid

User defined: □

Grid window...

Spacing: Variable

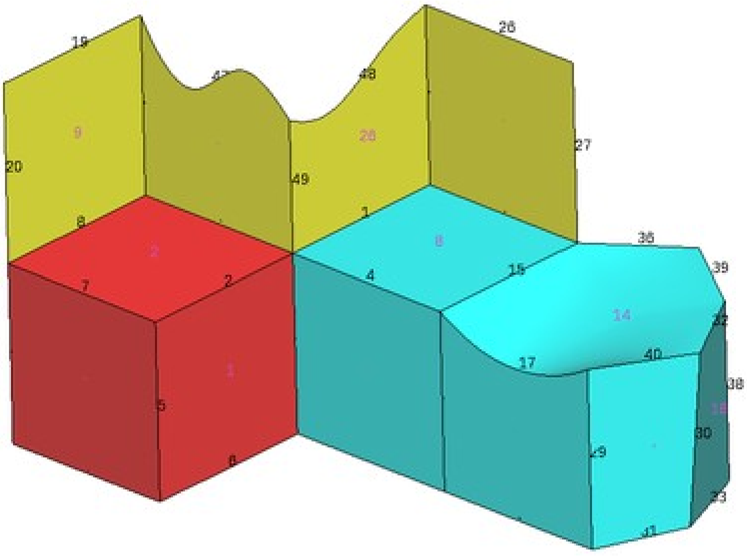
Minimum size: 1e-08

Size ratio: 3

A cartesian mesh is special kind of meshes, used for example by Finite Differences solvers, it is a grid with all edges aligned to global X,Y,Z axes.

Generating the cartesian mesh

We are going to use the previous model named "ToMesh3.gid", open this model.



Set in preferences

mesh type: Cartesian,

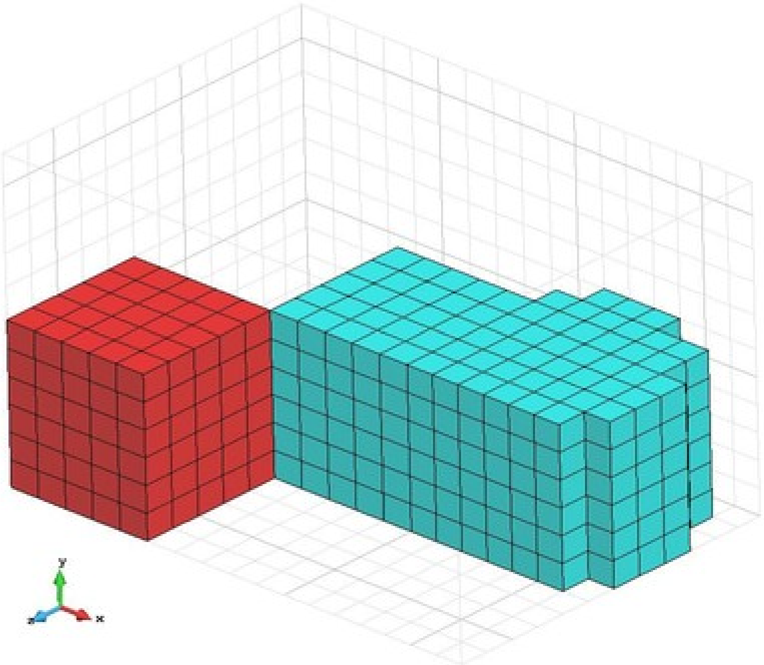
then in the cartesian branch set

mesher: Scan convert

The **Shape** to be meshed could be set to: **From mesh** or **from geometry**.
In case of 'From mesh' an auxiliary unstructured mesh of triangles will be generated and will be used as input forthe cartesian mesher. This intermediate mesh will be deleted at the end, and in any moment will be visible for the user.

In case of 'From geometry' then the triangles of the 'render mesh' of the geometry will be used. This option could be faster if the render mesh was previously created. A drawback is that in GiD the render mesh may not be conformal between entities, and has small gaps, but usually do not affect the cartesian mesher.
Usually both shapes will generate similar cartesian meshes.

Left initially all default values and generate a mesh (menu: **Mesh->Generate mesh**) with general size=2 units

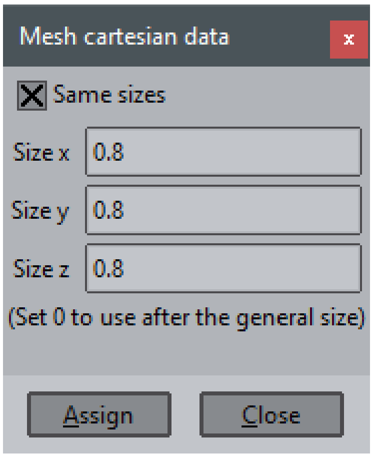


Then automatically is calculated a grid fitted (approximately) to the bounding box of the shape, with a cell size of 2 units in all directions X Y Z, and a cartesian mesh of hexahedra is generated. Note that the green surfaces were not meshed.
Note: This mesher could only generate 3D meshes for volumes (hexahedra), or 2D meshes for surfaces, all in z=0. It cannot generate in 3D meshes of faces for surfaces or edges for curves.

The 'lower-right' grid icon ▦ switch on/off the drawing of the underlying grid

In this case, the default **Spacing** was set to **Variable**, but the result looks very similar to a **Uniform** mesh (all cells seems equal-spaced in all directions)

Now assign to the volume number 3 a specific size=0.8 in all directions. Select Mesh->Assign cartesian size->Volumes



Mesh cartesian data

☒ Same sizes

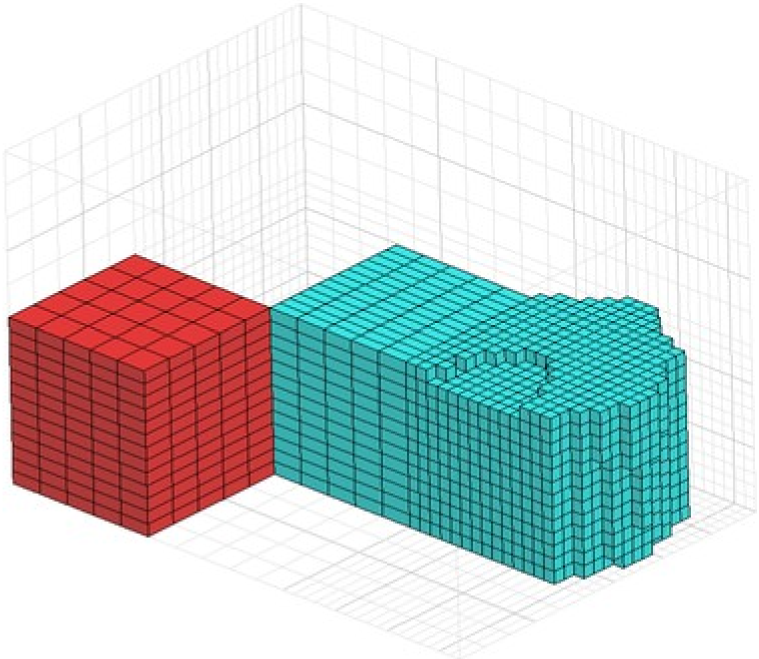
Size x 0.8

Size y 0.8

Size z 0.8

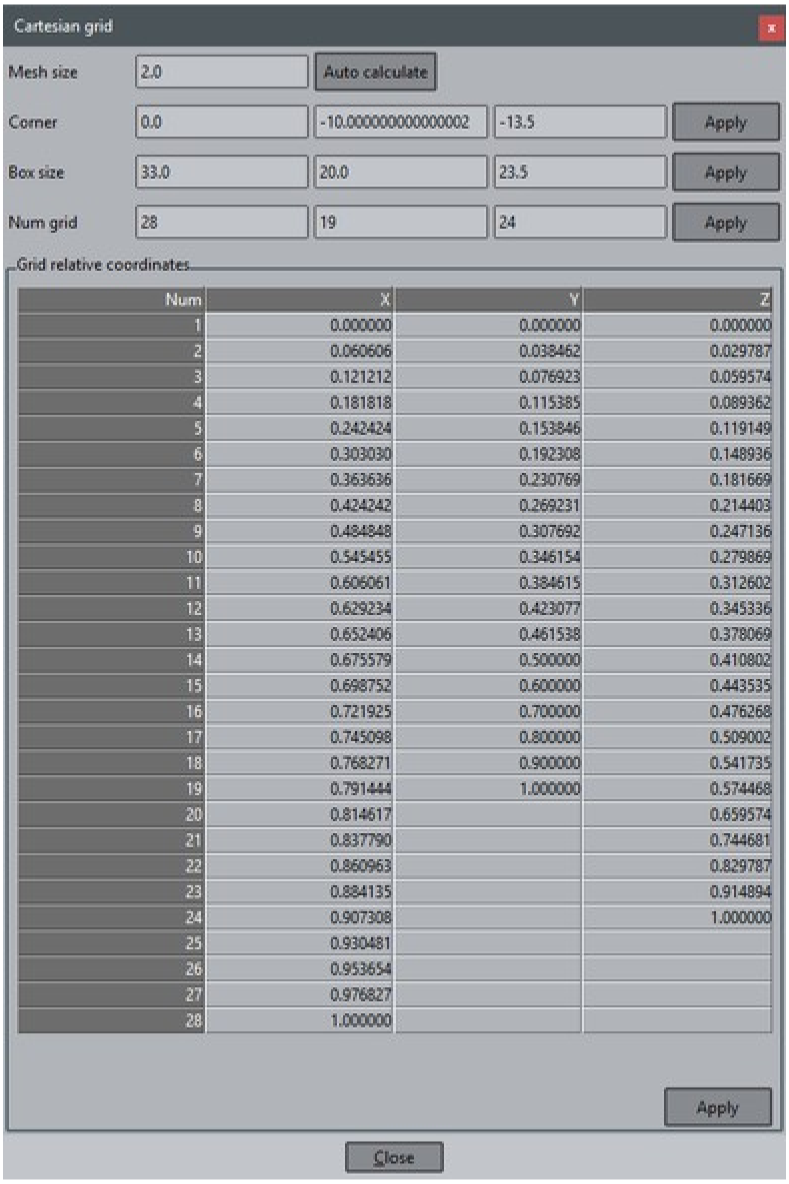
(Set 0 to use after the general size)

and select the volume number 3 (on the right of the images) finishing with a double **<Esc>**
and generate again a mesh with general size=2.0. You will obtain a non-uniform mesh like these:



In case of Variable spacing, the **Minimum size** try to avoid a too small edge size, and the **Size ratio** try to avoid big differences of sizes between two consecutive edges.

The grid is automatically defined when generating the mesh (based on the current geometry and mesh sizes), but the user could set its own definition:
In preferences check **User defined**, press **Apply** to accept it, and then the **Grid window...** button become enable. Pressing it, a window like the following one appears, filled with the current values:



Mesh size: 2.0

Corner: 0.0 | -10.000000000000002 | -13.5

Box size: 33.0 | 20.0 | 23.5

Num grid: 28 | 19 | 24

Grid relative coordinates

Num	X	Y	Z
1	0.000000	0.000000	0.000000
2	0.060606	0.038462	0.029787
3	0.121212	0.076923	0.059574
4	0.181818	0.115385	0.089362
5	0.242424	0.153846	0.119149
6	0.303030	0.192308	0.148936
7	0.363636	0.230769	0.181669
8	0.424242	0.269231	0.214403
9	0.484848	0.307692	0.247136
10	0.545455	0.346154	0.279869
11	0.606061	0.384615	0.312602
12	0.629234	0.423077	0.345336
13	0.652406	0.461538	0.378069
14	0.675579	0.500000	0.410802
15	0.698752	0.600000	0.443535
16	0.721925	0.700000	0.476268
17	0.745098	0.800000	0.509002
18	0.768271	0.900000	0.541735
19	0.791444	1.000000	0.574468
20	0.814617		0.659574
21	0.837790		0.744681
22	0.860963		0.829787
23	0.884135		0.914894
24	0.907308		1.000000
25	0.930481		
26	0.953654		
27	0.976827		
28	1.000000		

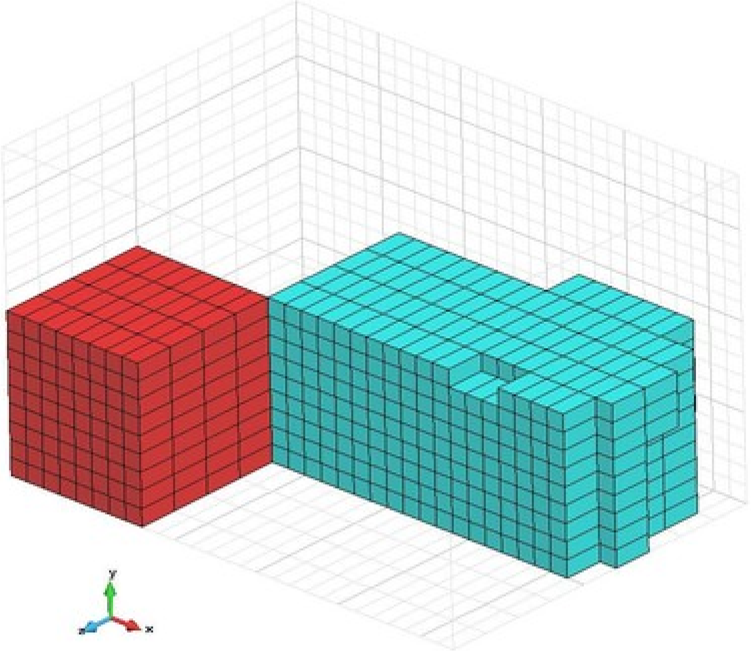
In this window it is allowed to modify the location and size of the box, set the number of divisions and set the values of the ticks editing the values of the table.

The 'Auto calculate' button allows to fill automatically all fields, based on the geometry and mesh sizes (similar to the case of generating a mesh with this size)

Changing only the (lower left) corner, the box size or the amount of divisions and pressing 'Apply' will try to set the data of the row, maintaining the other data.

E.g. if we modify the **Num grid** Z (third value) from 24 to 10 ticks and press the **Apply** button of its row then the Z grid values of the table change (to have an increasing array of 10 values normalized from 0.0 to 1.0)

Then if we generate again a mesh with general size=2.0 we obtain this mesh:



Zmesher cartesian mesher

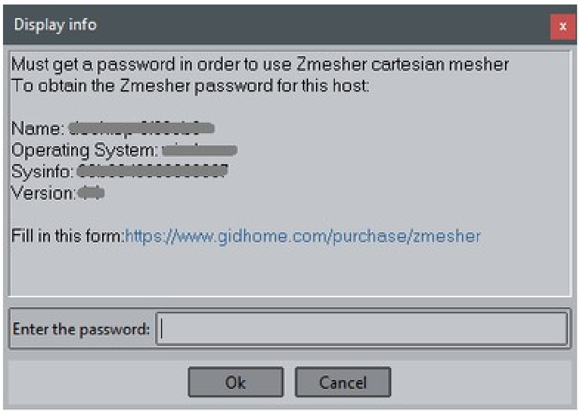
Current GiD has two cartesian algorithms available: 'Scan convert' and 'zmesher'.

Zmesher is able to generate 3D cartesian meshes of volumes, surfaces, lines and/or points. It is especially suitable for electromagnetic simulations of Finite Differences.

Zmesher has been implemented by a third part partner (University of Granada), and the first time that is used it will require an extra password.

Set **mesher: Zmesher** in the preferences window
and generate again the mesh, disabling the 'User defined' grid and with general size=2.0

If it is the first time that this mesher is used this window ask a password



Must get a password in order to use Zmesher cartesian mesher
To obtain the Zmesher password for this host:

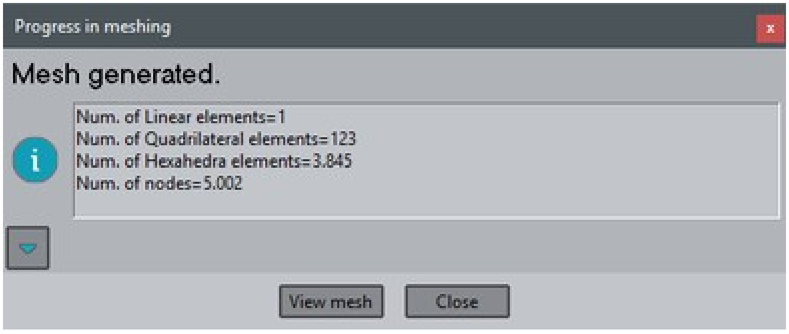
Name: [redacted]
Operating System: [redacted]
Sysinfo: [redacted]
Version: [redacted]

Fill in this form:https://www.gidhome.com/purchase/zmesher

Enter the password:

The password could be freely obtained from the related web (and will be stored in scripts/TemporalVariables file, so it won't be required again to get it).

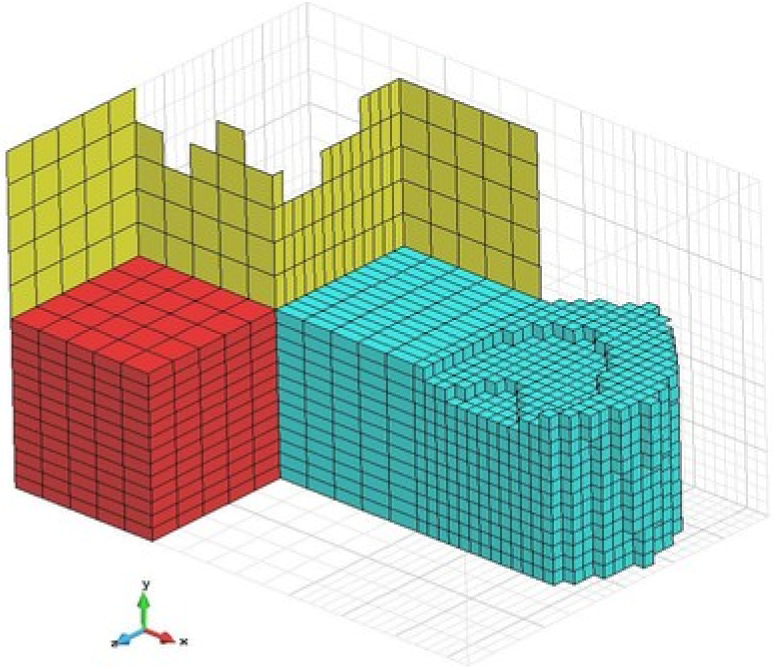
Enter the password and generate again the mesh. When the mesh is generated is show this information



Mesh generated.

Num. of Linear elements=1
Num. of Quadrilateral elements=123
Num. of Hexahedra elements=3.845
Num. of nodes=5.002

Now the mesh will have also quadrilateral faces for the isolated surfaces



In fact, this mesher could create some 'degenerated' cells, to preserve topological connections. E.g. 1 line element is generated although there is not any isolated geometrical curve, and also 1 extra quadrilateral is generated from a volume.
If the 'Avoid degeneration' option is set, then these degenerated elements are automatically deleted.

Embedded mesh

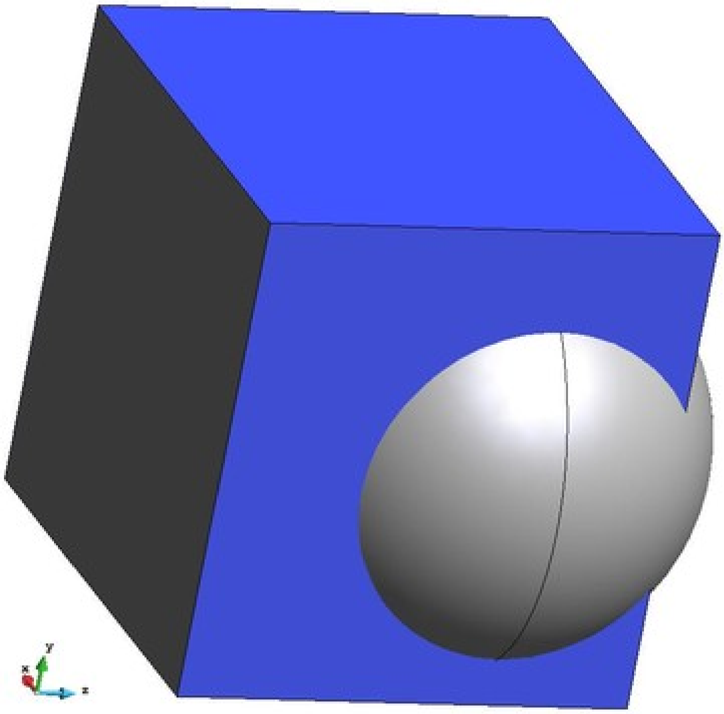
GiD can generate three types of mesh: body-fitted, embedded and Cartesian. We will focuse on the **embedded** type in this chapter.

This mesh type corresponds to an embedded mesh (not fitted to the contours of the embedded closed region). This kind of models have one or more geometrical volumes (the calculation volumes) and an embedded closed region. The calculation volumes are meshed in a body-fitted way, and its nodes have the extra information of (signed) distance to the contours of the embedded region.

It has to be noted that the definition of the embedded region is not needed to be watertight, so GiD can manage definitions with small gaps or overlapping entities.

Reading the model

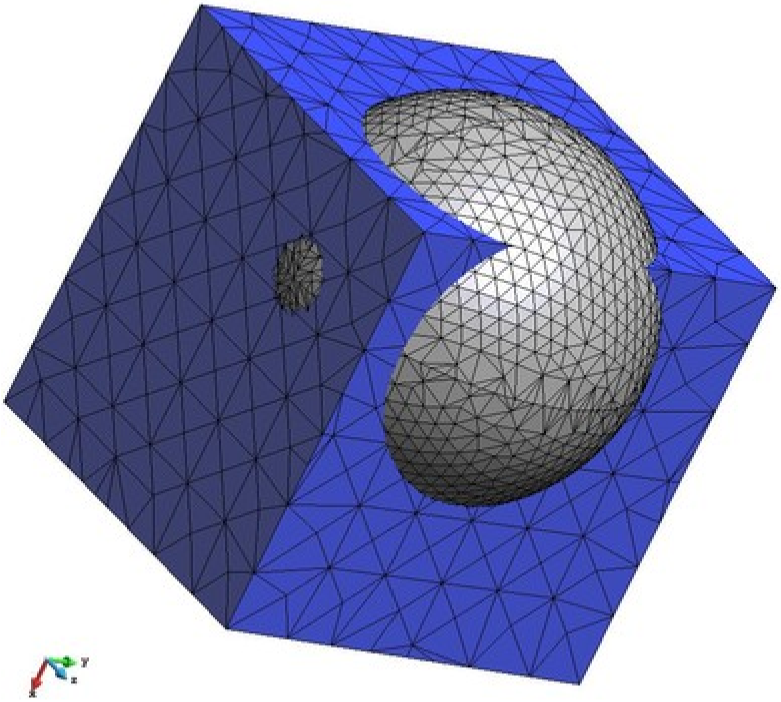
In the **Files** menu, select **Read**. Select the project "gid_embedded_example.gid" and click **Open**.
The following goemetrical model should be opened.



In this model, the cube is the calculation volume (which will be meshed body-fitted), and the sphere skin is defining the embedded region.

Generating the embedded mesh

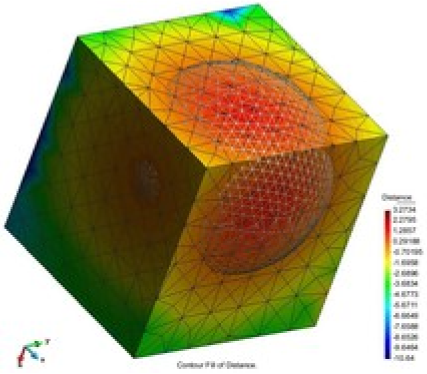
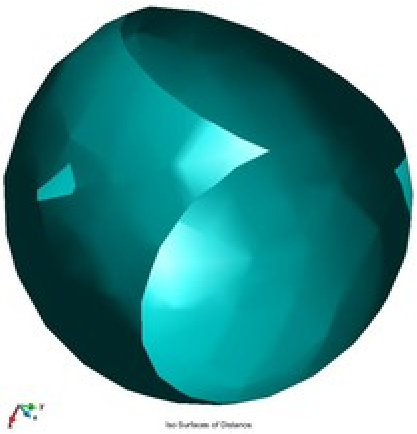
First of all, et the **embedded** mesh type in the *Meshing->General branch* of the preferences window.



Click on Mesh->Generate mesh, and generate a mesh with size equal to 1. The following mesh should appear:

As it can be seen, the volume mesh is body-fitted to the calculation volume, but is not constrained by the contours of the embedded region.

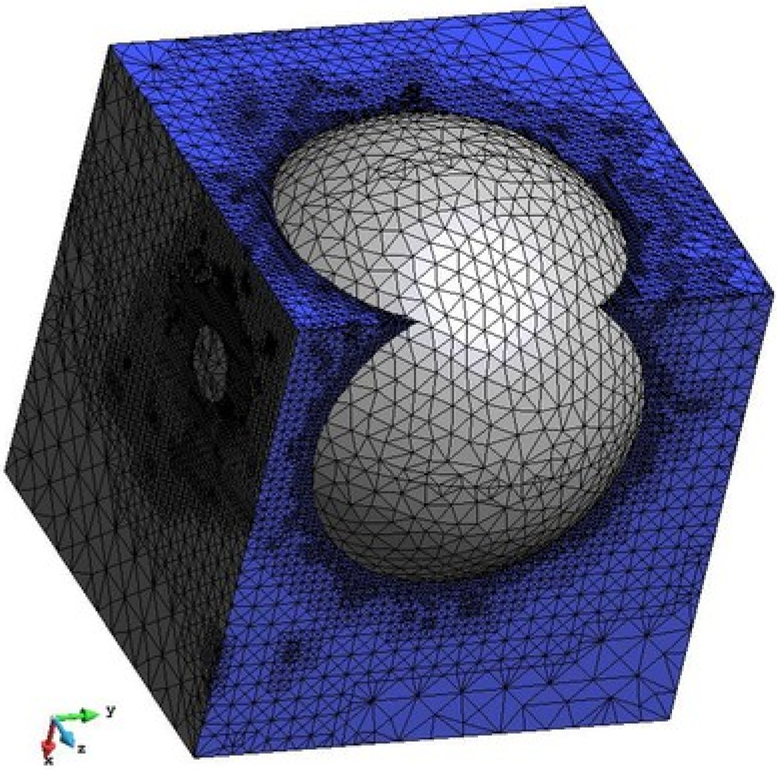
The volume mesh has also the information of the signed distances from its nodes to the embedded region (negative values indicates the nodes are inside the embedded region. To see this information, you can go to the post-processing mode of GiD, and see a contour fill of the distances, or the iso-surface of distance equal to 0, as shown hereafter.

 Distance 3.2734 2.2795 1.2857 0.29188 -0.70195 -1.6958 -2.6896 -3.6834 -4.6773 -5.6711 -6.6649 -7.6588 -8.6526 -9.6464 -10.64 Contour Fill of Distance	
View of the contourfill of distances on the calculation volume mesh.	View of the iso-surface of distance equal to 0 of the calculation volume mesh, where the sphere shape can be appreciated.

Assigning sizes

Although the embedded mesh is not constrained by the contours of the embedded region, one can assign mesh sizes to that contours, and the volume mesh will follow them.
As an example, assign an unstructured size of 0.1 (Mesh->Unstructured->Assign size to surfaces) to the surfaces of the sphere skin, and generate the mesh with general size of 1.0 again.

The following mesh should be generated:

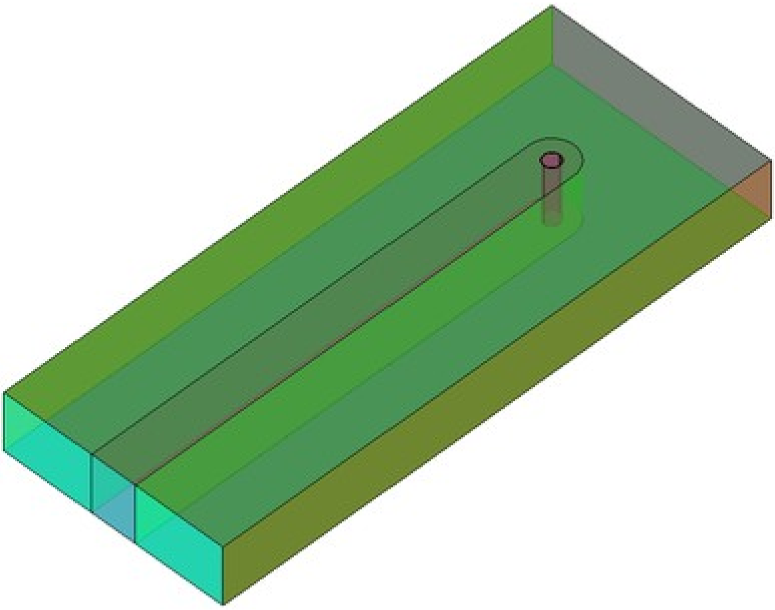

y
z
x

POSTPROCESSING

The objective of this tutorial is to do a postprocess analysis of an already calculated fluid simulations, no preprocess option is used.
Not only the model is already meshed and the constraints are assigned, but also the results have been calculated. For more information about the preprocess part of GiD, please check the preprocess tutorials.
In this tutorial, the model *Cylinder.bin* has been used. The problem type used to do this simulations is Tdyn, particularly the Ransol model. Tdyn is a fluid dynamic (CFD) simulation environment based on the stabilized Finite Element Method.

Steps followed in this tutorial:

- Loading the model
- Changing mesh styles
- Visualization of results
- Creating images

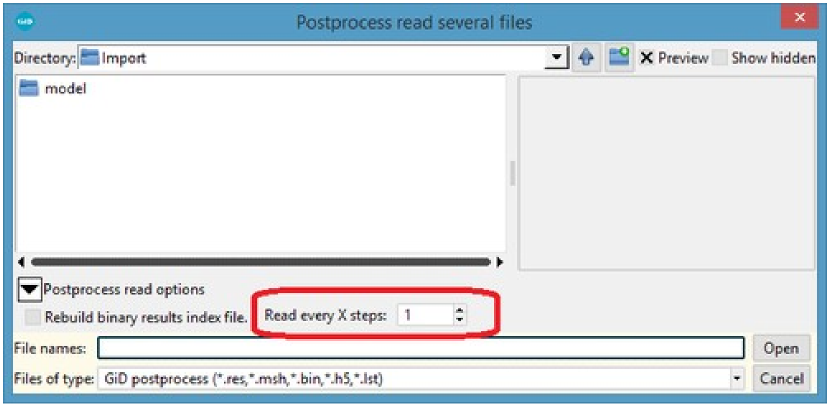


Loading the model

There are three ways to load the results simulation information into GiD:

- If there is a GiD model (a *modelname.gid* folder) and the results are also inside the GiD folder with the right nomenclature (e.g. *modelname.post.res*), then just load the GiD project and switch to postprocess mode.
- If only a mesh and results file(s) are present then GiD should be started, and switched to postprocess mode and then load the result or mesh file.
- You can also drag-and-drop the post-process file(s) into GiD.

For big models with lots of steps, there is an option to only read some of the steps:



Postprocess read several files

Directory: Import

model

Postprocess read options

□ Rebuild binary results index file.

Read every X steps: 1

File names:

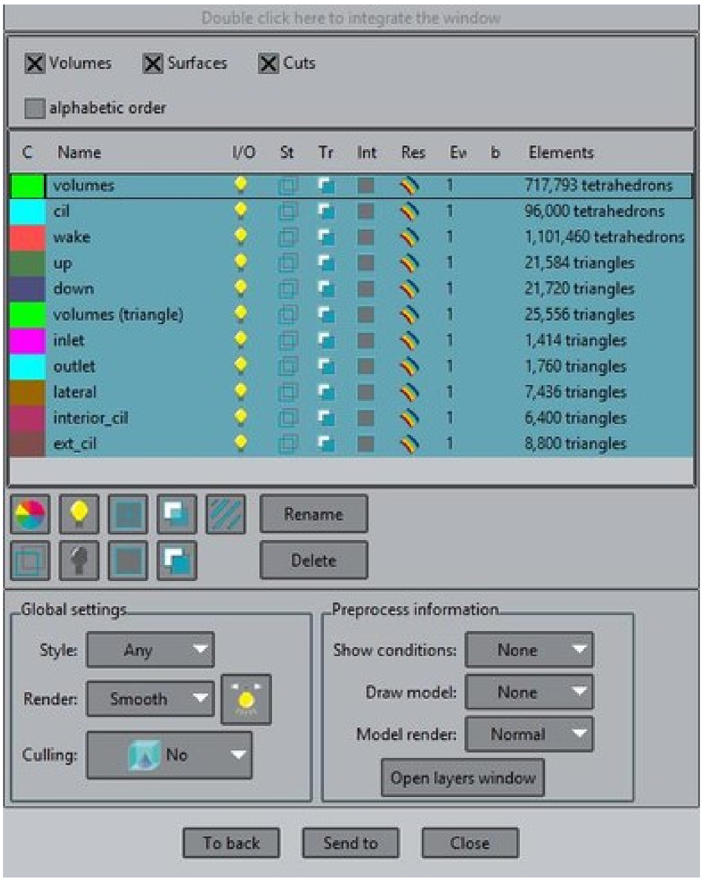
Files of type: GiD postprocess (*.res,*.msh,*.bin,*.h5,*.lst)

For this tutorial we will use the file called *Cylinder.bin* that contains the postprocess information of mesh+result in a single binary file, so the steps to follow are:

1. Start GiD
2. Switch to postprocess mode: from the **Files->Postprocess** menu or the icon
3. Open the model with: **Files->Open**, **Ctrl-o** or clicking on

Changing mesh styles

- Select **Window-->View style...**
- Select all the layers.
- Change the style to **Boundaries** (Global settings, to be applied to all sets).
- Play a little with the options of these windows, but to continue the tutorial, select the **Boundaries** style for all meshes.
- Change render mode to **Normal.**



☒ Volumes ☒ Surfaces ☒ Cuts

☐ alphabetic order

C	Name	I/O	St	Tr	Int	Res	Ev	b	Elements
🟩	volumes	💡	□	◪	■	⟋	1		717,793 tetrahedrons
🟦	cil	💡	□	◪	■	⟋	1		96,000 tetrahedrons
🟥	wake	💡	□	◪	■	⟋	1		1,101,460 tetrahedrons
🟩	up	💡	□	◪	■	⟋	1		21,584 triangles
🟦	down	💡	□	◪	■	⟋	1		21,720 triangles
🟩	volumes (triangle)	💡	□	◪	■	⟋	1		25,556 triangles
🟪	inlet	💡	□	◪	■	⟋	1		1,414 triangles
🟦	outlet	💡	□	◪	■	⟋	1		1,760 triangles
🟫	lateral	💡	□	◪	■	⟋	1		7,436 triangles
🟥	interior_cil	💡	□	◪	■	⟋	1		6,400 triangles
🟫	ext_cil	💡	□	◪	■	⟋	1		8,800 triangles

Global settings

Style: Any

Render: Smooth

Culling: No

Preprocess information

Show conditions: None

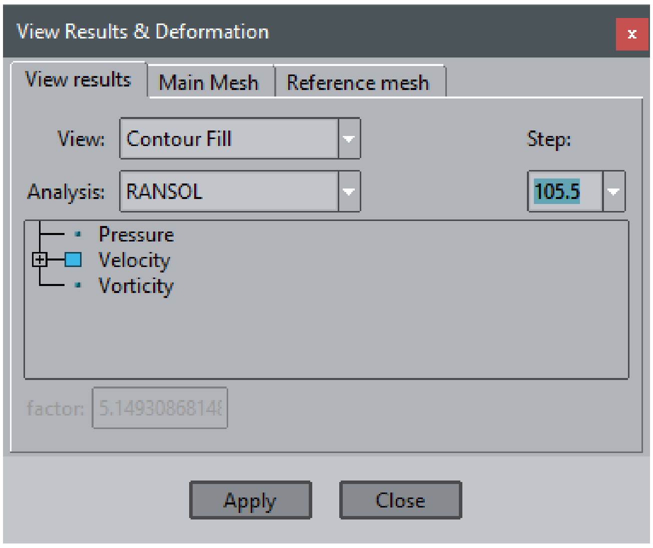
Draw model: None

Model render: Normal

Viewing the results

In the example, several results have been calculated for several time steps. You can check these results using the **View Results** menu options, or using the **Window-->View results** window or clicking the results view icon bar.

Menu: View results
Window-->View results...



View: Contour Fill

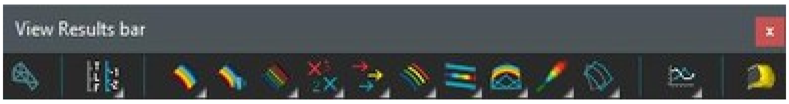
Step: 105.5

Analysis: RANSOL

- Pressure
- Velocity
- Vorticity

factor: 5.14930868148

Results view icon toolbar



Iso surfaces

Menu: View results->Iso surfaces
With this result visualization an iso-surface is drawn passing through all the points which have the same result'svalue inside a volume mesh (iso-lines for a surface mesh). To create isosurfaces there are several options.

- Select **View results->Iso surfaces->Automatic width->Velocity->|Velocity|** through the menu bar or clicking on on the results view icon bar.

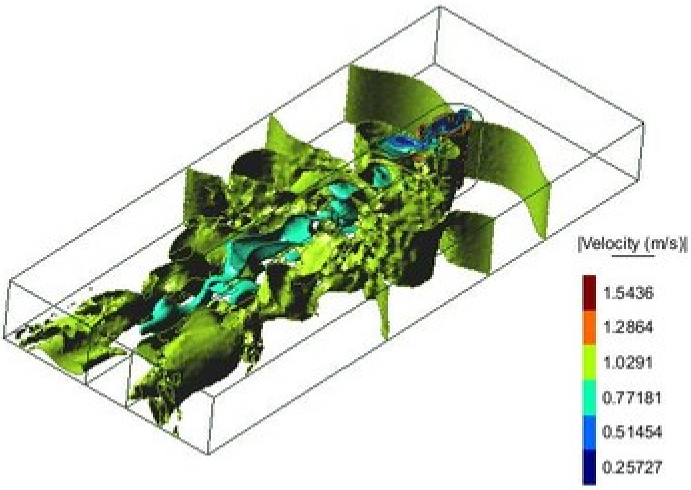
After choosing the result, you are asked for a width. This width is used to create as many isosurfaces as are needed between the Minimum and Maximum defined values (these are included).

- Enter the value 0.25727 to get the picture below.
- Select **View->Render->Smooth** in order to get a better view.

Several configuration options can be set via the preferences window .

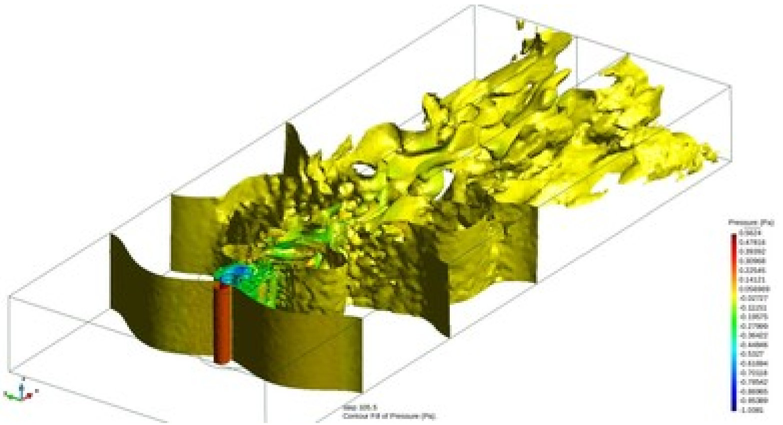
Menu: Utilities-->Preferences
In the **Postprocess-->Iso surfaces** tree
Display style the style of the iso-surface (similar options to the style of each set), set it to **Body**.
In order to see the inner zones we will set the transparency on the iso surfaces.

- Select **Transparency: Transparent**
- Move the model to see the inner zones
- Restore again **Transparency: Opaque**



Other interesting options are:

- **Color mode** allows to draw the iso-surfaces with a single colour (**Monochrome**), according to the results used to create the iso-surface (**Result color**) or using the color map of the visualized contour fill result (**Contour fill color**).
- For instance, select **Color mode = Contour Fill Color** and
- Do a **contour fill** of *Pressure*:

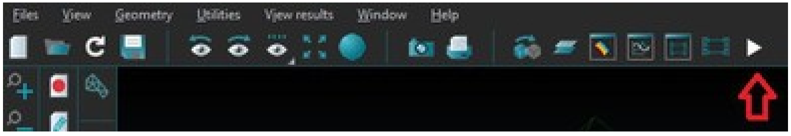


- **Show isolines** this option allows the user to switch isolines of surfaces on or off. Switch it **off**.
- **Draw always** if this option is selected the iso-surfaces are always drawn even though all the meshes are switched off.

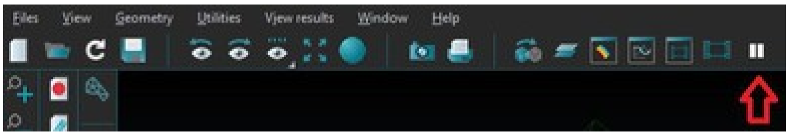
It is possible to convert the iso-surfaces in a mesh of triangles, with the menu: **Geometry->Convert->Iso surfaces to sets**
Then a new mesh will appear in the view style window. This mesh could be for example exported to a file, or use the rest of mesh operations.

Animate

Menu:Window->Animate...
You can animate a visualized results pressing 'play' in the animation icon on the Standar Bar:

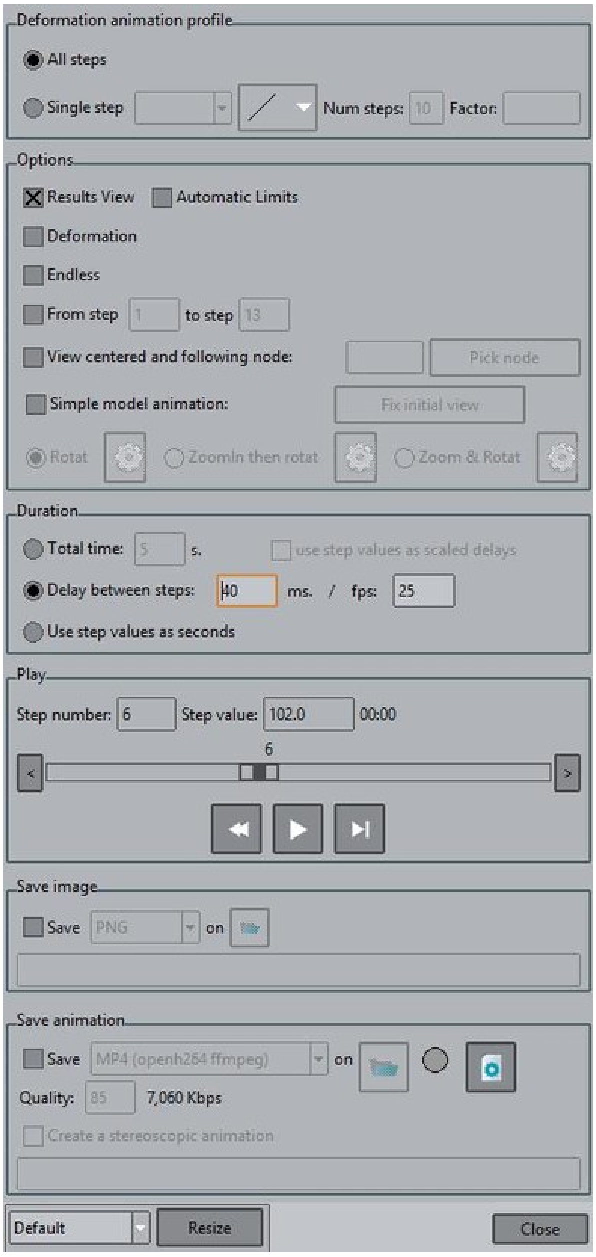


Pres again to stop the animation:



The Window-->Animate window allows a more complete control of all animation options of the current visualizedresult.

If only one step is present (or if the model has been deformed using a result's vector), then the static animation options are enabled, so that a custom animation profile can be set up to animate that one (deformation) step.



Deformation animation profile

● All steps

○ Single step | Num steps: 10 | Factor:

Options

☒ Results View ☐ Automatic Limits

☐ Deformation

☐ Endless

☐ From step 1 to step 13

☐ View centered and following node: | Pick node

☐ Simple model animation: | Fix initial view

● Rotat ○ ZoomIn then rotat ○ Zoom & Rotat

Duration

○ Total time: 5 s. ☐ use step values as scaled delays

● Delay between steps: 40 ms. / fps: 25

○ Use step values as seconds

Play

Step number: 6 Step value: 102.0 00:00

6

Save image

☐ Save PNG on

Save animation

☐ Save MP4 (openh264 ffmpeg) on

Quality: 85 7,060 Kbps

☐ Create a stereoscopic animation

Default | Resize | Close

If one result has several steps you can visualize them in an animation. In this case we will use the iso surfaces result.

- Select **View->Render->Smooth**
- Select **Window->Animate...**to open the animation window

Please notice that we have from step 1 to 13. We will do the animation only of some of these steps.

- Check the **From step** option and set 3 **to step** 8
- Try it clicking on the **play** icon

We will record a video during the animation.

- Once the animation is finished check the **Save** option on the **Save animation** part

You can choose from several video formats.

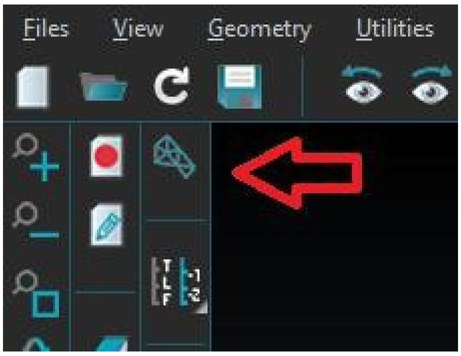
- Select one of the video formats (codec and container)
- Select a folder where the video will be saved clicking on the **folder** icon or writing the path in the text entry
- Select Delay between steps and enter the value 1000 ms.
- Click on the **play** button and the recording will begin. This step could take a little bit long. Wait until the red circle turns to green
- **Close** the Animate window

Now we will visualize another result but before we will clear all the results.

- Select **View results->No results** through the menu bar or using the icon

No results

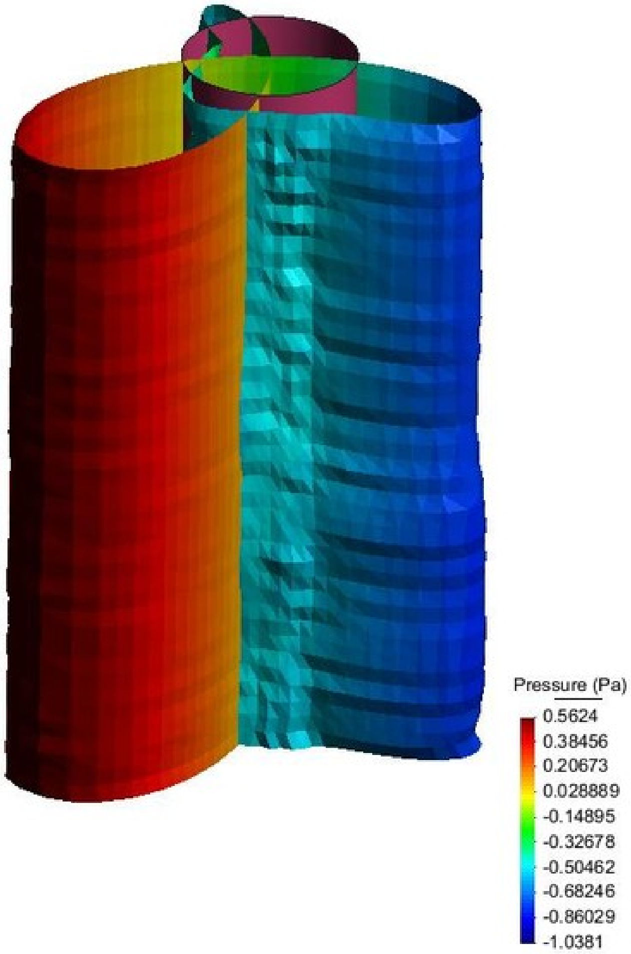
You can always reset the results visualization with the **no results** icon:



or with the menu option **View results->No results.**
This option clears almost any result visualization, like contour fill, showing the mesh in it's original state, i.e. without deformation vector.
Only stream lines and node traces are unaffected. To delete stream lines or node traces select the menu option **View results->Delete->Stream lines / Node trace.**

Result surface

Another result visualization of interest is this one:



To get this visualization follow these steps:

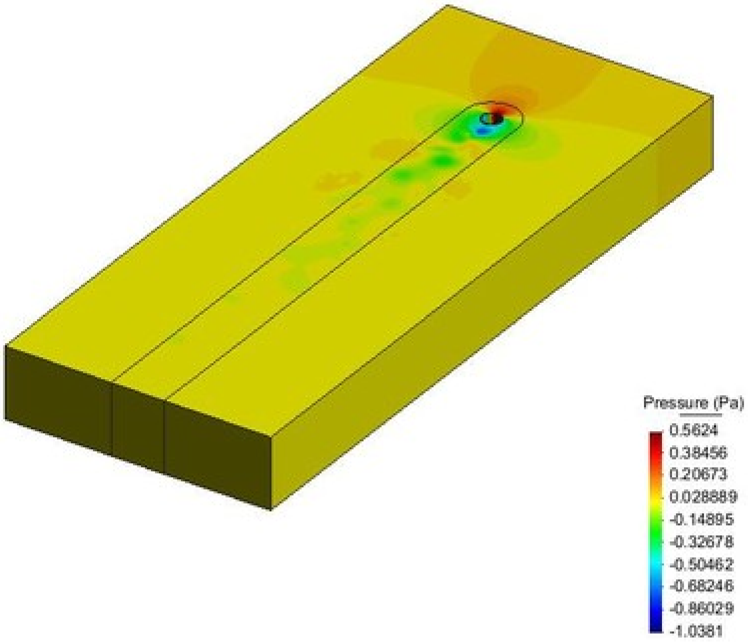
- Switch off all the sets except **interior_cil**. To do this:
    - Select **Window->View style...** in the menu bar.
    - Select all the sets except **interior_cil** pressing Ctrl while selecting with mouse.
    - Click on the **bulb light** icon on the **I/O** column or click on the icon .
    - You may need to **Zoom frame** the model to center the cylinder.
- Through the "View style" window change the Style to **Body Boundaries** .
- Select **View results->Result surface-> Pressure**. A surface will be drawn which results from moving the nodes along its smoothed normal according to the results value for this node.
- Enter **5** as scale factor in the bottom command line.
- In the preferences window there are some options in the **Postprocess->Result surface** panel

Select **Show elevations: None**

- Set the checkbox **Draw contour fill**. With this last option the surface is colored according to the pressure value. Press Apply to accept the current values.
- Select **View results->No results** through the menu bar or using the icon
- Switch on all the sets again through the "View style" window by selecting all sets and clicking on the icon. You may need to **Zoom frame** again to center the model view.

Contour fill, cuts and limits

Contour fill



Menu: View results->Contour fill

- Please select **View results->Contour fill->Pressure** through the menu bar, or clicking on or using the **Window->View results...** window.
- If not all sets show the contour fill like the picture above, remember to select **Body Boundaries** mesh style for all the sets.

This option allows the visualization of result using a colour map, in which a scalar variable or a component of a vector varies between two defined values. GiD can use as many colours as permitted by the graphical capabilities of the computer The number of colours can be set through **Utilities->Preferences** - **Postprocess->Contour fill and lines**
By number of colors.
A menu of the available results to be represented will be shown, and the one that is chosen will be displayed using the current step selected.
In the model the pressure has been calculated. We can visualize the result for each step in a contour fill.

You can choose the step that you want to view through the **View results** window or clicking on the icon

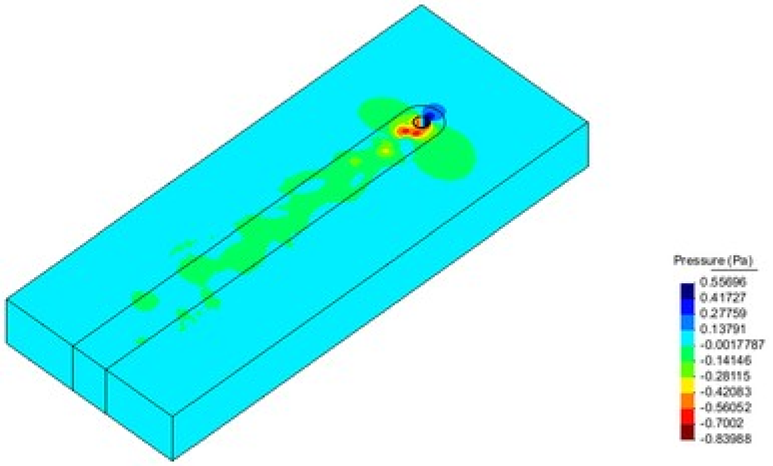
- Select the the Analysis named **RANSOL** and its **step 103.0**

Several configuration options can be set via the preferences window .
You can change the color scale in order to get a more comfortable view. You can select several predefined color maps. The default scale is **rainbow**, which is a rainbow colour map starting from blue (minimum) throughgreen and yellow, to red (maximum).

- Select **Color map: Inverse rainbow**

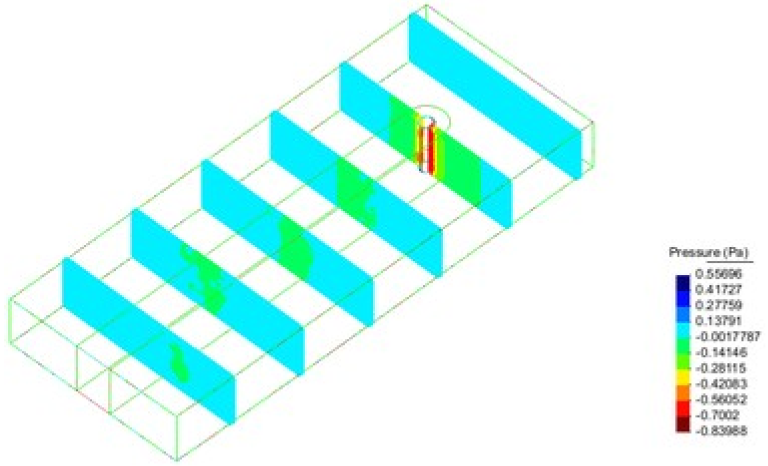
You can also define your own scale.

- Select **Color map: User defined...** then the button 'More color options' could be picked to open a colors window. In this window you can change the number of different colors used in the scale. If you need more accuracy you can increase this number, or decrease it for a higher contrast.
- Change the number of colors to **10**
- Click on **Apply** button
- Click on **Close** button



Cuts

In order to view the inner zone we will do several cuts along the model.



Menu: Geometry->Cut plane

In order to make it easier first we will change the plane visualization.

- Please select **View->Rotate->Plane XY(Original)** through the menu bar, **Rotate->Plane XY(Original)** through the mouse menu or clicking on and XY . Now you have a top view of the model.
- Cuts are done to both the volume sets and the surface sets resulting in triangle meshes and line meshes. In this case we only want to cut the volume meshes, so please switch off all surface meshes
- Select **Geometry->Cut plane->Succession** through the menu bar or clicking on and then

With the **succession** option you specify an axis that will be used to create cut planes orthogonal to this axis. The number of planes is also asked for.

- Draw a line through the X axis in the middle of the model and ask for 7 cuts. You should obtain 7 parallel planes to Y axis.

Note: after clicking the first point, pressing the *Alt* key while moving the mouse the dynamic line will be screen-axis aligned or at 45 degrees.

- Now open the display style window (**Window->View style**) to switch off all meshes except the created *Cuts*. You can see that several layers have appeared with a prefix like **Cut** indicating which mesh or set has been cut. These names can always be changed by selecting the mesh and pressing **F2** or *right-click* and *rename* option.
- Select all the layers except the cuts and change their style to **Boundaries** model in order to see the contour fill result on the

cut planes.

. You can rotate the

- In the same window select all the **Cuts** and click on **Delete** button in order to delete them.
- Select **Body Bound** as mesh style to visualize the **contour fill** of **pressure** again.
- In the preferences window , select **Postprocess->Contour fill and lines** and press the **Default values** button in order to restore the default options.

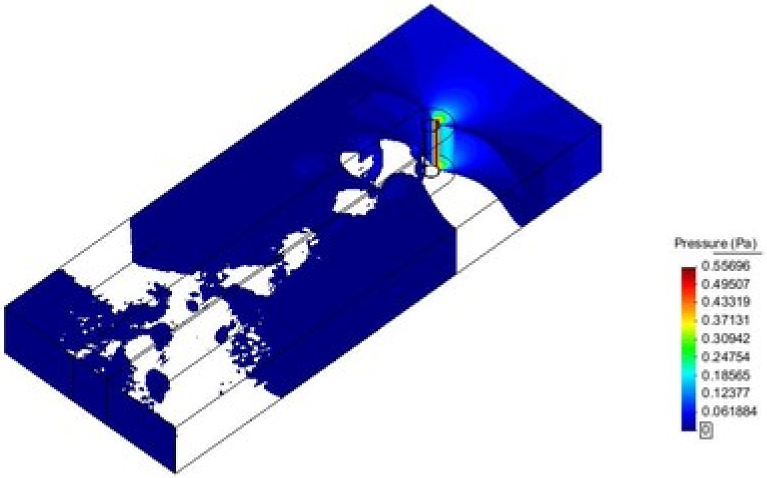
Define limits

You can set the minimum and maximum limit values for the contour fill. In our case we only want to see the positive values. In order to do this we will set the minimum value to 0.0 .

- Click on the icon of the toolbar, and an entry will ask you for the minimum value that Contour Fill should use. Change the value to 0.0 and press return to accept the value and close.

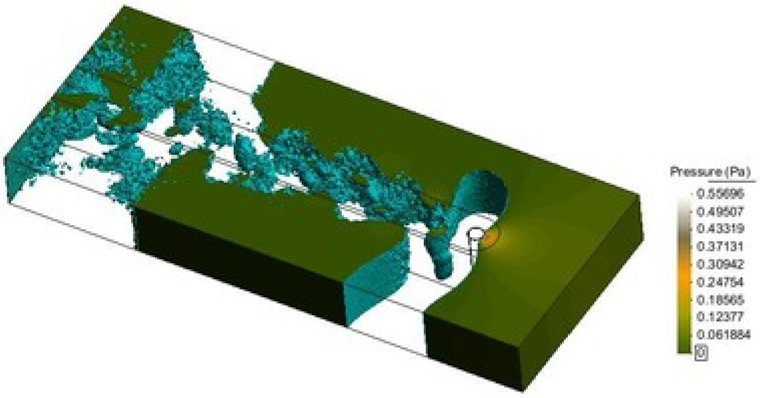
Outliers will be drawn in the colour defined in the **Out minimum colour** option of the preferences window . In order to filter out the negative values we will change this color to transparent.

- Select **Out minimum colour: Transparent** and press **Apply.**



Combined results

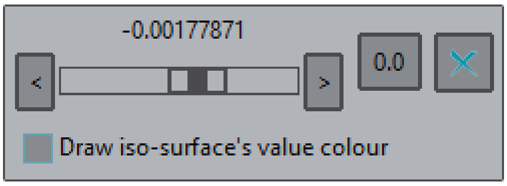
An interesting postprocess options is to combine several result visualizations, like this one:



To get this view follow these steps:

- Clear all results visualizations with **View results->No results** or the icon.
- Select **View results->Default analysis/step->RANSOL->103.0**

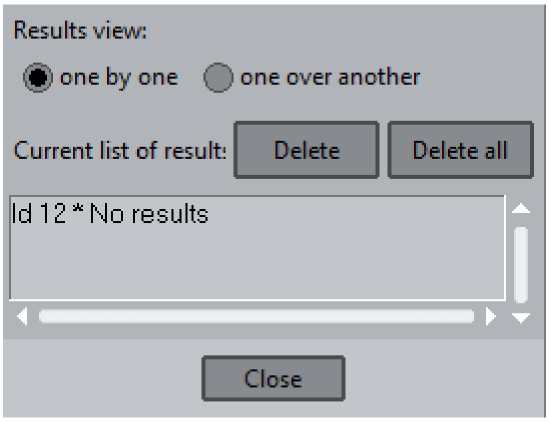
- Select **View results->Iso surfaces->Exact->Pressure** through the menu bar. If you click the icon a window like this will appear, that allows you to dynamically select the value used to visualize the iso-surface:



- In the following questions: How many **isosurfaces**? Enter 1
- Enter the 1 value ...? Enter 0
- To visualize the iso-surface you may change the display style to **Boundaries** , but for the contour fill visualization of the next steps remember to change to **Body Boundaries** style .
- Select **View results->Contour fill->Pressure**
- Fix the **Minimum** value to 0.0
- And in preferences of Contour fill set **Out minimum colour: Transparent**
- Select **Color map: Terrain**
- And in preferences of Iso surfaces select **Color mode: Monochrome**
- The **Mono color** button allows you to change the color of the iso surface.

For other result visualization combinations you need to enable the Several results options:

- Select **Window->Several results**. Then following window appears:

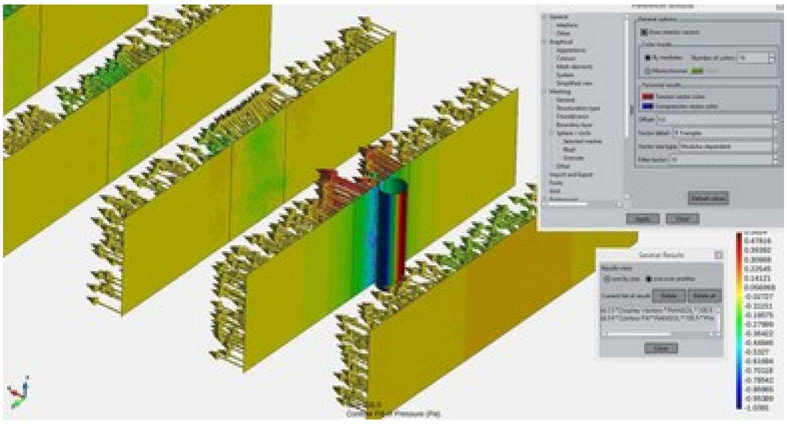

Results view:

● one by one ○ one over another

Current list of result:

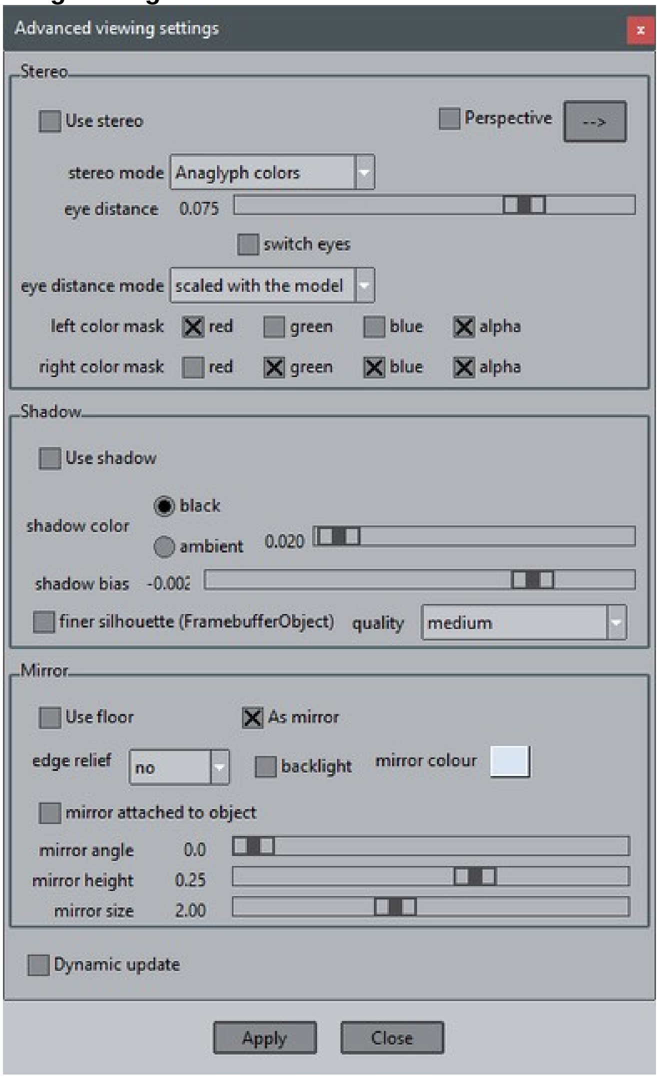
Id 12 * No results

- In this window select **one over another**. With this option GiD is told to visualize one result over another, . e.g. to simultaneously draw the value of a pressure and the vectors of a velocity:



Stereo mode (3D)

Menu: View->Advanced viewing settings...



☐ Use stereo
☐ Perspective
stereo mode: Anaglyph colors
eye distance: 0.075
☐ switch eyes
eye distance mode: scaled with the model
left color mask: ☒ red ☐ green ☐ blue ☒ alpha
right color mask: ☐ red ☒ green ☒ blue ☒ alpha

☐ Use shadow
shadow color: ● black ○ ambient 0.020
shadow bias: -0.002
☐ finer silhouette (FramebufferObject) quality: medium

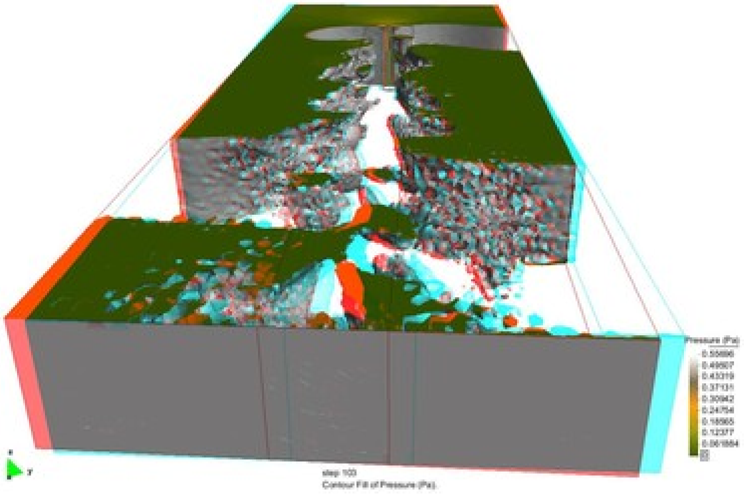
☐ Use floor
☒ As mirror
edge relief: no
☐ backlight
mirror colour
☐ mirror attached to object
mirror angle: 0.0
mirror height: 0.25
mirror size: 2.00

☐ Dynamic update

If you have an **anaglyphic** glasses you can try this option. The model can be set as an anaglyphic image in order to provide a stereoscopic 3D effect, when viewed with 2 color glasses (each lens a chromatically opposite color, usually red and cyan).
Anaglyphic images are made up of two color layers, superimposed. Since the glasses act as red and cyan filters we should be careful with the model's colors. To avoid problems confusing values we will change the contour fill color scale.

- Select in preferences the contour fill **Color map: 3D Anaglyphs.**
- Select **View->Advanced viewing settings...**
- Check the **Use stereo** option.

- Check the **Dynamic update** option in order to change the options without the need to click the Apply button.
- Adjust the **eye distance** to a value where you can see the 3D effect.



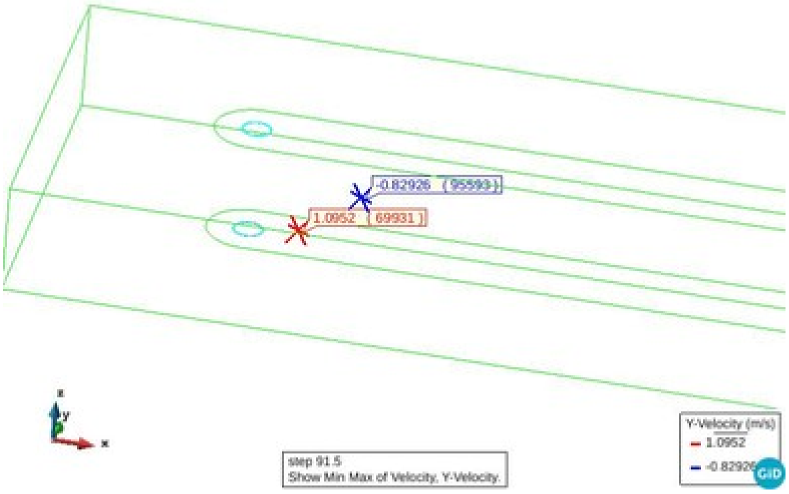
- This 3D option works best with **Perspective** enabled. Give it a try.
- Uncheck the **Use stereo** option to finish this kind of visualization.
- **Close** the window.
- Select **View results->No Results** **.**
- Change the **view style** to **Boundaries** for all the layers.

Show min max

Menu: View results->Show min max

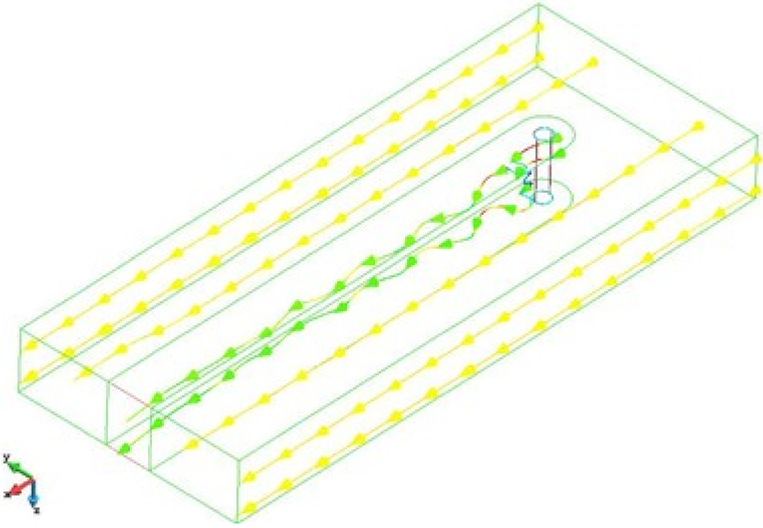
With this option you can see the minimum and maximum value of the chosen result in the chosen analysis step. In our case we will choose the Vy component of velocity result for the first analysis step.

- Select **View results->Default analysis/step->RANSOL->91.5** through the menu bar or clicking on
- Select **View results->Show min max->Show both->Velocity->Y-Velocity** through the menu bar or clicking on . The label shows the node number and the value of the result.



- To finish select **View results->No results** **.**

Stream lines



Menu: View results->Stream lines

With this option you can display a stream line, or in fluid dynamics, a particle tracing, in a vector field.

Note: stream lines are confined in a single volume mesh, i.e. they do not jump from one volume mesh to the next volume mesh, even if they are close neighbours. In the provided example there are three volume meshes and stream lines will not cross the volume boundaries. You can join the volume meshes into a new single volume mesh using **Geometry->Join->Volume sets** . Then you can delete the three separate volumes and switch the single joined volume mesh on.

The above image results from doing this tutorial with the three separated volumes. The image at the end of this *stream lines* tutorial is achieved if following step is done before the enumerated *stream lines* tutorial steps.

Select **Geometry->Join->Volume sets** to create a single volume mesh, and delete the three other volume meshes: *V volumes*, *V cil* and *V wake*.

- Select **View results->Default analysis/step->RANSOL->103.0** through the menu bar or clicking on
- Select **View results->Stream lines->Along line->Velocity** through the menu bar

With this option you can define a segment along which several start points will be chosen. The number of points will also be asked for, including the ends of the segment. In the case of just one start point, this will be the center of the segment.

NOTE: This action could also be done clicking on in the icon bar. In this case we have to select the way to define the start point through the mouse menu. In this case select **Contextual->Along line.**
We want to create several stream lines along the model doing 2 lines.

- Write the **initial** point in the command line 10,15,3
- Write the **final** point in the command line 10,-15,3
- You are asked for the **number of points** along the line. Enter **5** and click **Ok**.

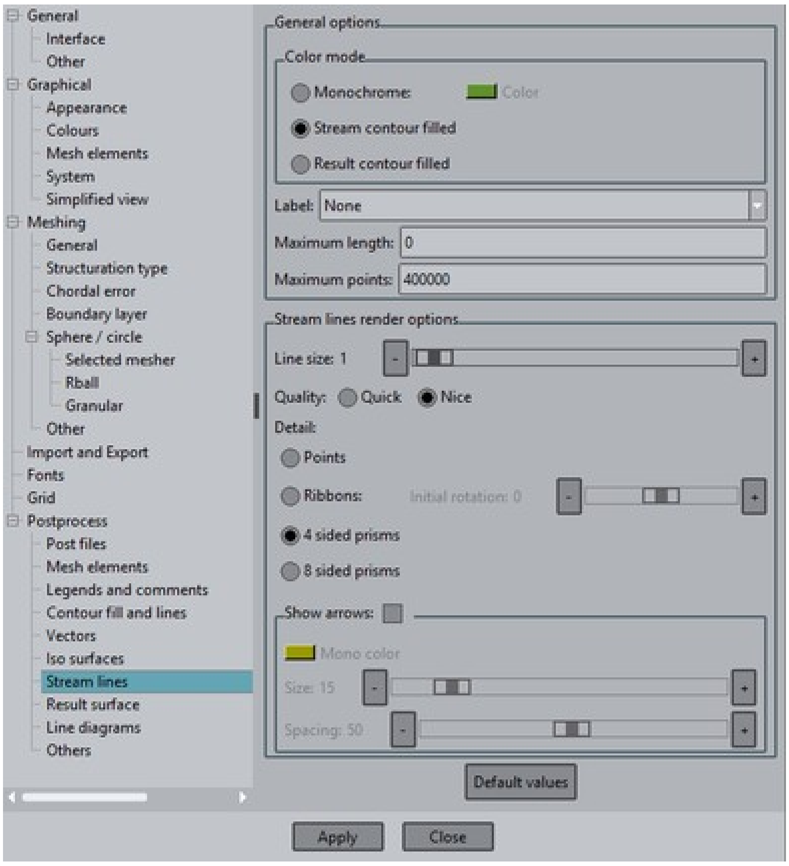
The first line with 5 stream lines is created, repeat it to sample other line:

- Write the **initial** point in the command line 10,15,7
- Write the **final** point in the command line 10,-15,7
- You are asked for the **number of points** along the line. Choose 3.

The second serie of 3 stream lines is created.

- Click the middle mouse button or press the **<Esc>** key in order to finish the operation.

Several configuration options can be set via the Preferences window : **Postprocess->Stream lines**



General options

Color mode

○ Monochrome: 🟩 Color

● Stream contour filled

○ Result contour filled

Label: None

Maximum length: 0

Maximum points: 400000

Stream lines render options

Line size: 1

Quality: ○ Quick ● Nice

Detail:

○ Points

○ Ribbons: Initial rotation: 0

● 4 sided prisms

○ 8 sided prisms

Show arrows: □

Mono color

Size: 15

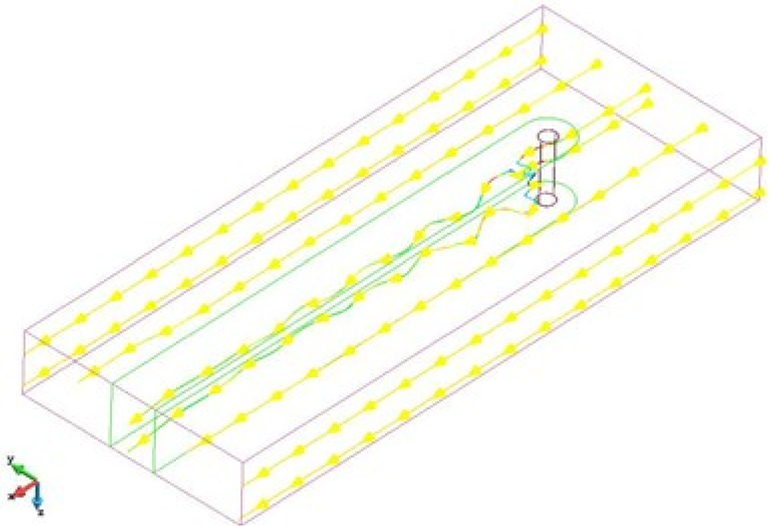
Spacing: 50

- Play with some of the **Stream lines** options, like:
- **Color mode:Stream contour filled** (remember to set the contour fill colour map mode to rainbow)**.**

The stream lines will be drawn with the colors used in the velocity contour fill.

- Set **Show arrows** , with **Size: 30** and **Spacing: 10**
- You may play with the different stream line **Detail level:** Points, Ribbons or 4 sided prisms. If the ribbons type is selected you may adjust the initial swirl angle to rotate the ribbon.
- Close the window
- Select **View results->Delete->Stream lines->All**

NOTE: A way to achieve the best results is to first create a cut of the volume mesh through the *region of interest* and then use these nodal information as support to create *stream lines* and its options: *along line, in aquad,* etc.



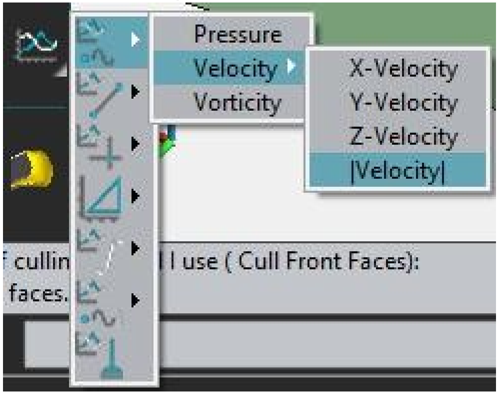
Graphs

Menu: View results->Graphs

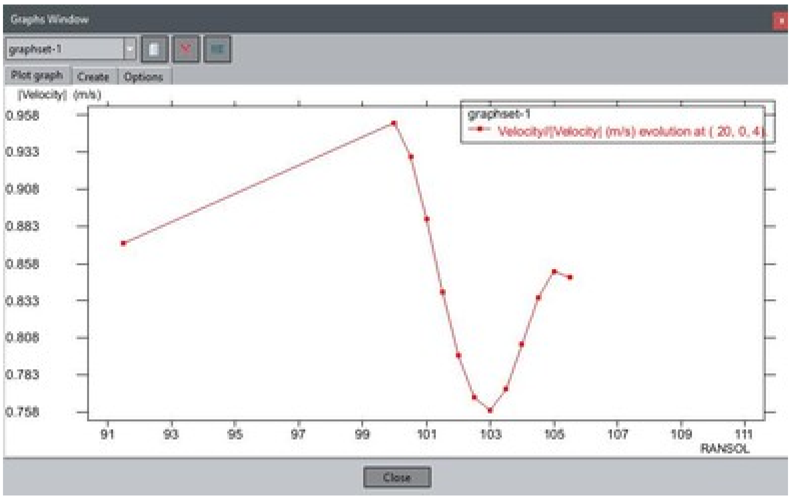
From this menu several graphs types can be created, we will try some of them. Graphs are supported for results defined over nodes because they represent a continuous field (gauss-element results represent a discontinuous field)
Graphs are organized into **graph sets** in order to ease the management. Each set shares the same units for each axis.
When a graph is created is placed in the current graphset if the units are the same, otherwise a new graphset is created.
In order to work with graphs we will use the 'graphs window'.

The **Point evolution** graph displays a graph of the evolution of the selected result along all the steps, of the default analysis, for the selected nodes.

- Select **View results->Graphs->Point evolution->Velocity->|Velocity| .** You can also click on the graphs icon and select the point evolution option:

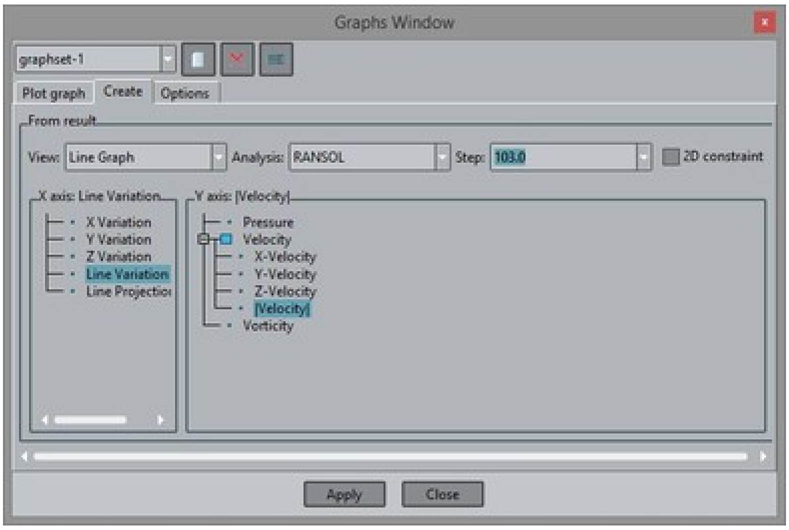


- Write 20,0,4 in the command line in order to specify the point.
- After pressing the **<Esc>** key, or the middle mouse button, the graph will be shown in a separate window:



The graph is created in the graphset-1. We will create another graph in the same graph set.
The **Line graph** displays a graph along the line connecting two selected nodes of surfaces or volumes, or any arbitrary points on any projectable surface and in any position.

- Switch all surface meshes off, and let only the three volume meshes on: volumes, cil, wake.
- Create this graph from the 'Create' tab of the Graphs window



Graphs Window

graphset-1

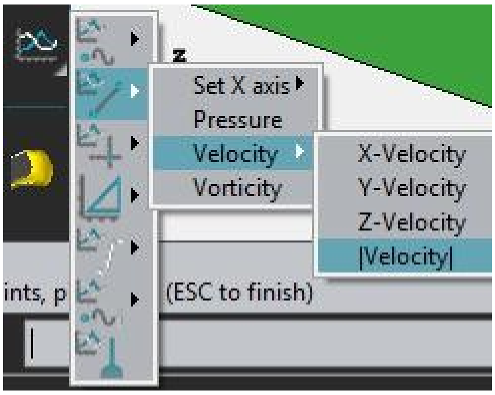
View: Line Graph   Analysis: RANSOL   Step: 103.0   □ 2D constraint

X axis: Line Variation   Y axis: |Velocity|

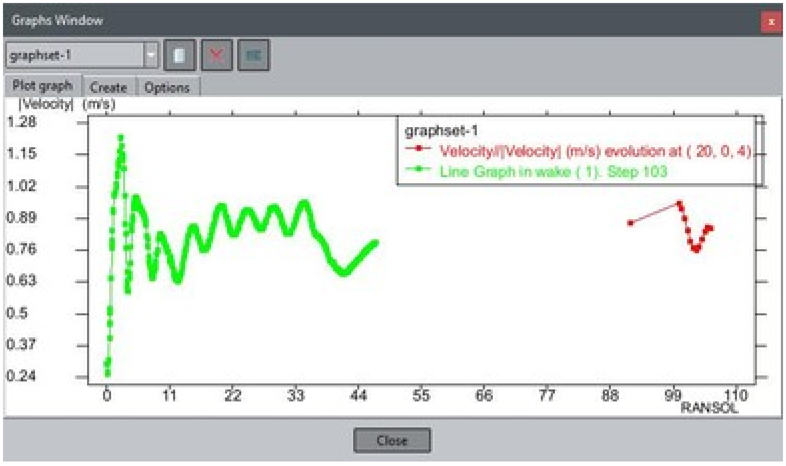
Press the Apply button to select the two points of the line:

- Write 3,0,4 in the command line in order to specify the initial point.
- Write 50,0,4 in the command line in order to specify the final point.

You can create this graph also from the results icon bar:



Now both graphs are showed in the same graph set:



We will rename the graph set.

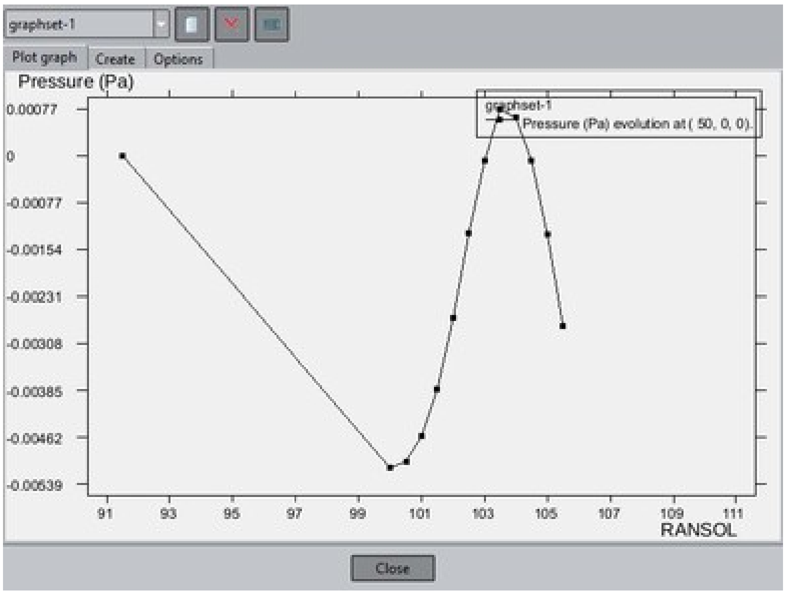
- In the top part of the window click the icon.
- A window will appear asking for a new name. Enter 'Velocity', for example.

We will create a new graph set.

- In the top part of the window click the icon.

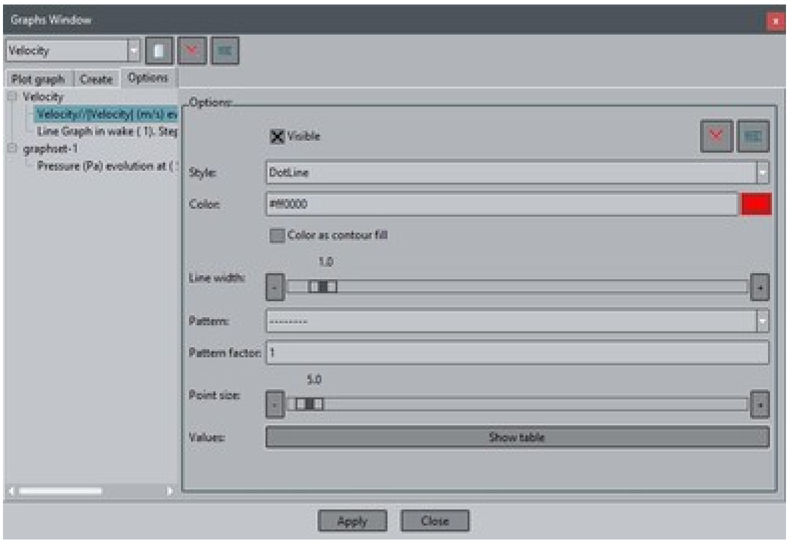
A new graph set is created with default name 'graphset-1'. When a new graph set is created becomes the current one. We can see that there are no graphs on this new graph set.
It's also possible to create graphs from the graph window.

- Go to **Create** tab and select **Point evolution** int **View** option.
- In **Y Axis** list double click **Pressure**.
- Press Apply to select the point and write 50,0,0 in the command line in order to specify the coordinates of a new point (<Ctrl>-a to swap to select an existent node by its id)
- Press **<Esc>** to finish the selection of points and create the graph:



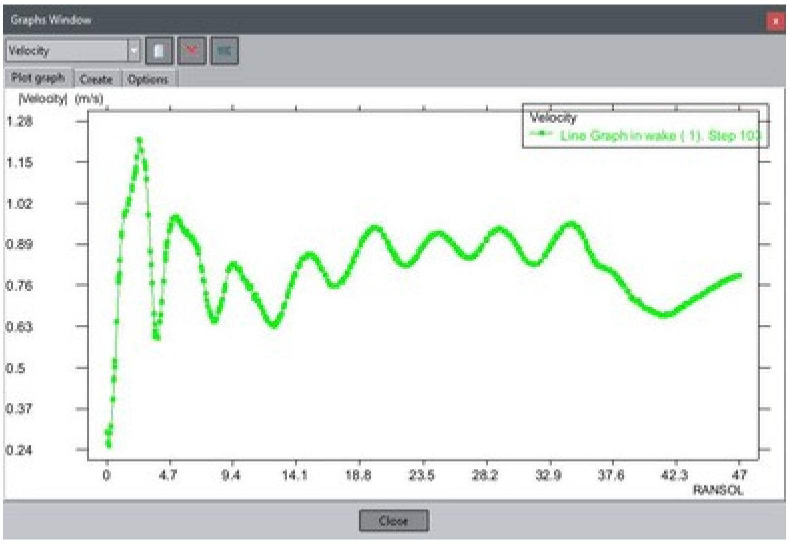
We can manage graphs and graphs sets in the Options panel. Depending if we are selecting a graph set or a graph in the tree we will see different options in the tab.

- Go to the **Options** panel, select the 'Velocity (m/s) evolution at (20, 0, 4)' graph and delete it pressing the button with the red cross.
- A confirmation window appears. Click **Yes**.


Graphs Window
Velocity
Plot graph
Create
Options
Velocity
Velocity//|Velocity| (m/s) ev
Line Graph in wake (1). Step
graphset-1
Pressure (Pa) evolution at (
Apply
Close

Options:
☒ Visible
Style: DotLine
Color: #ff0000
☐ Color as contour fill
Line width: 1.0
Pattern: ---------
Pattern factor: 1
Point size: 5.0
Values: Show table

- Please notice that the current graph set have been changed to 'Velocity'. Now the Plot graph panel will show only one graph:

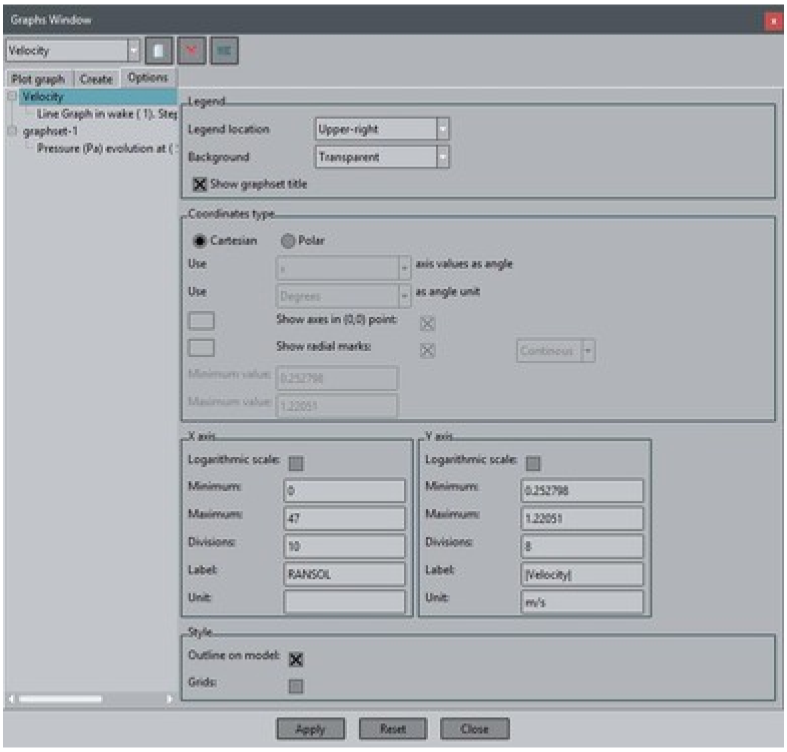


The graph size is readapted. We can will change several style options of a graph.

- **Double click** in any point of the graph and we will access to the **Options** tab.
- Set **Style: Line**
- Set to red the **Color** option. You can do it writing #ff0000 or selecting the red clicking on the right color window
- Set to 4.0 the **Line width**
- Click on **Apply** button

Graphset options can be managed selecting the set in the tree.

- Select *Velocity* branch. The options will change.
- For instance mark **Logarithmic scale** option in X axis.
- Click on **Apply** button



Legend

Legend location: Upper-right

Background: Transparent

☒ Show graphset title

Coordinates type

● Cartesian ○ Polar

Use x axis values as angle

Use Degrees as angle unit

Show axes in (0,0) point: ☒

Show radial marks: ☒ Continuous

Minimum value: 0.252798

Maximum value: 1.22051

X axis

Logarithmic scale: □

Minimum: 0

Maximum: 47

Divisions: 10

Label: RANSOL

Unit:

Y axis

Logarithmic scale: □

Minimum: 0.252798

Maximum: 1.22051

Divisions: 8

Label: |Velocity|

Unit: m/s

Style

Outline on model: ☒

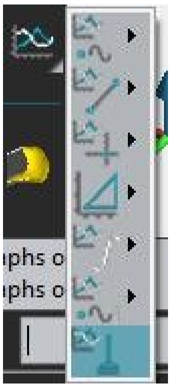
Grids: □

We can export the graph information in order to open it later with GiD or other third party programs.

- Select **Files->Export->Graph->All graphsets**. You are asked for the location where to save the *.grf* file. The *.grf* extension is the predefined by GiD, but it's a text file which can also be used in *gnuplot* or other programs.
- Choose the location

Now you can import the selecting **Files->Import->Graph.**

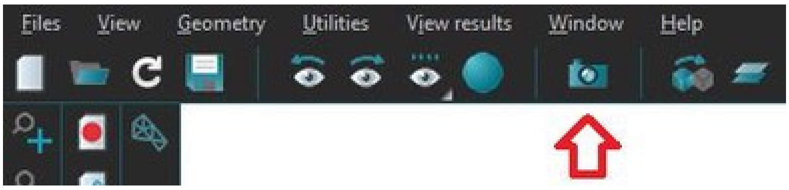
- Select **View results->Graphs->Clear** in order to delete all the graphs, or choose the clear graphs option from the graphs icon menu:



Creating images

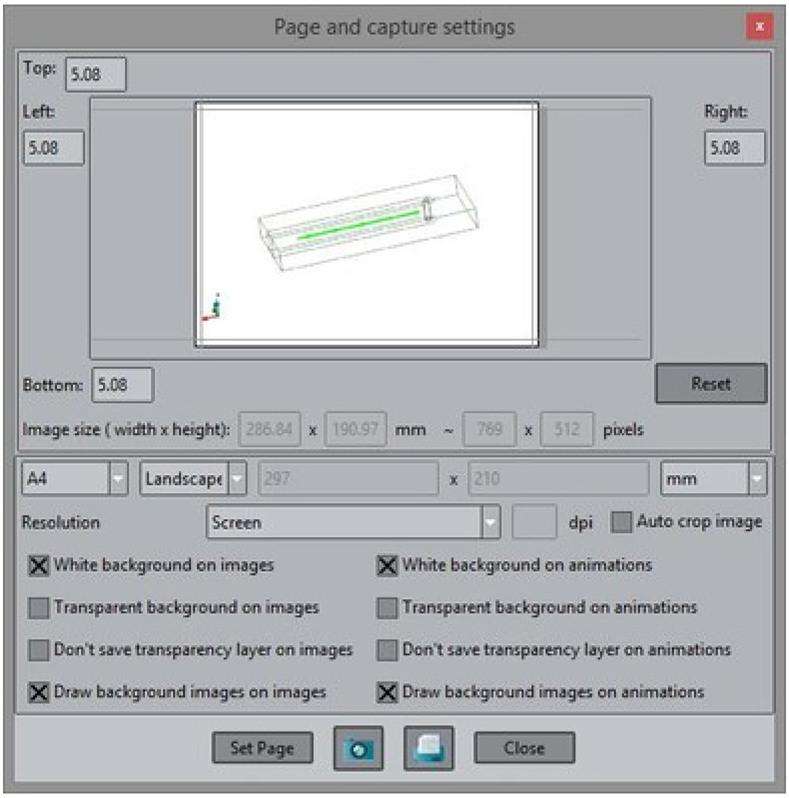
Menu: Files->Page and capture settings...

Finally we will take some snapshots of our model. You can save images in several formats.
The easiest way to capture images is by using the **camera icon** on the standard icon bar:



A file dialog window will appear from where you can choose the desired format to save the image.
A more detailed list of possible output formats can be found at **Files-> Print to file** top menu bar.
The properties of the image (resolution, size, background colour, etc.) can be adjusted in **Files-> Page and capture settings** option.

- Select **Files->Page and capture settings...**
- Check the **Auto crop image** option in order to cut the image in the model limits
- Click on **Set Page** button
- Eventually click on the **camera** button of this same window to take the snapshot.
- Click on **Close** button



Page and capture settings

Top: 5.08

Left: 5.08

Right: 5.08

Bottom: 5.08

Reset

Image size (width x height): 286.84 x 190.97 mm ~ 769 x 512 pixels

A4 | Landscape | 297 x 210 | mm

Resolution: Screen | dpi | □ Auto crop image

☒ White background on images | ☒ White background on animations

□ Transparent background on images | □ Transparent background on animations

□ Don't save transparency layer on images | □ Don't save transparency layer on animations

☒ Draw background images on images | ☒ Draw background images on animations

Menu: Files->Print to file

This option asks you for a file name and saves an image in the required format with the defined properties in **Page and capture settings**.

- Select **Files->Print to file->PNG...** through the menu bar
- Choose the location where you want to save the image
- Choose a name for the file
- Click on **Save** button

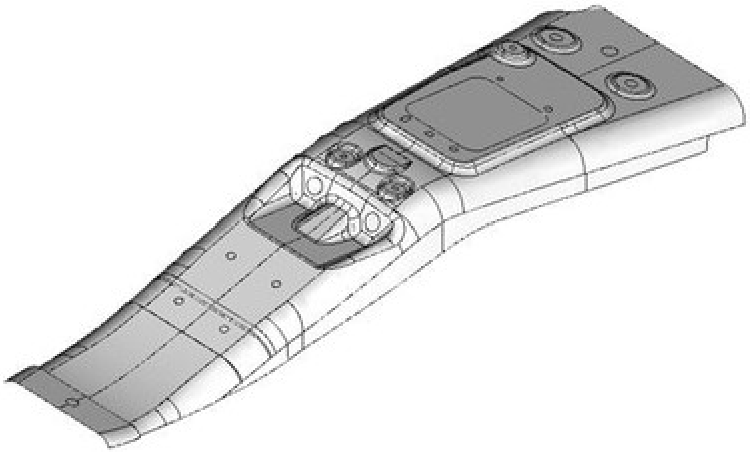
CAD CLEANING OPERATIONS

IMPORTING FILES

The objective of this case study is to see how GiD imports files created with other programs. The imported geometry may contain imperfections that must be corrected before generating the mesh.

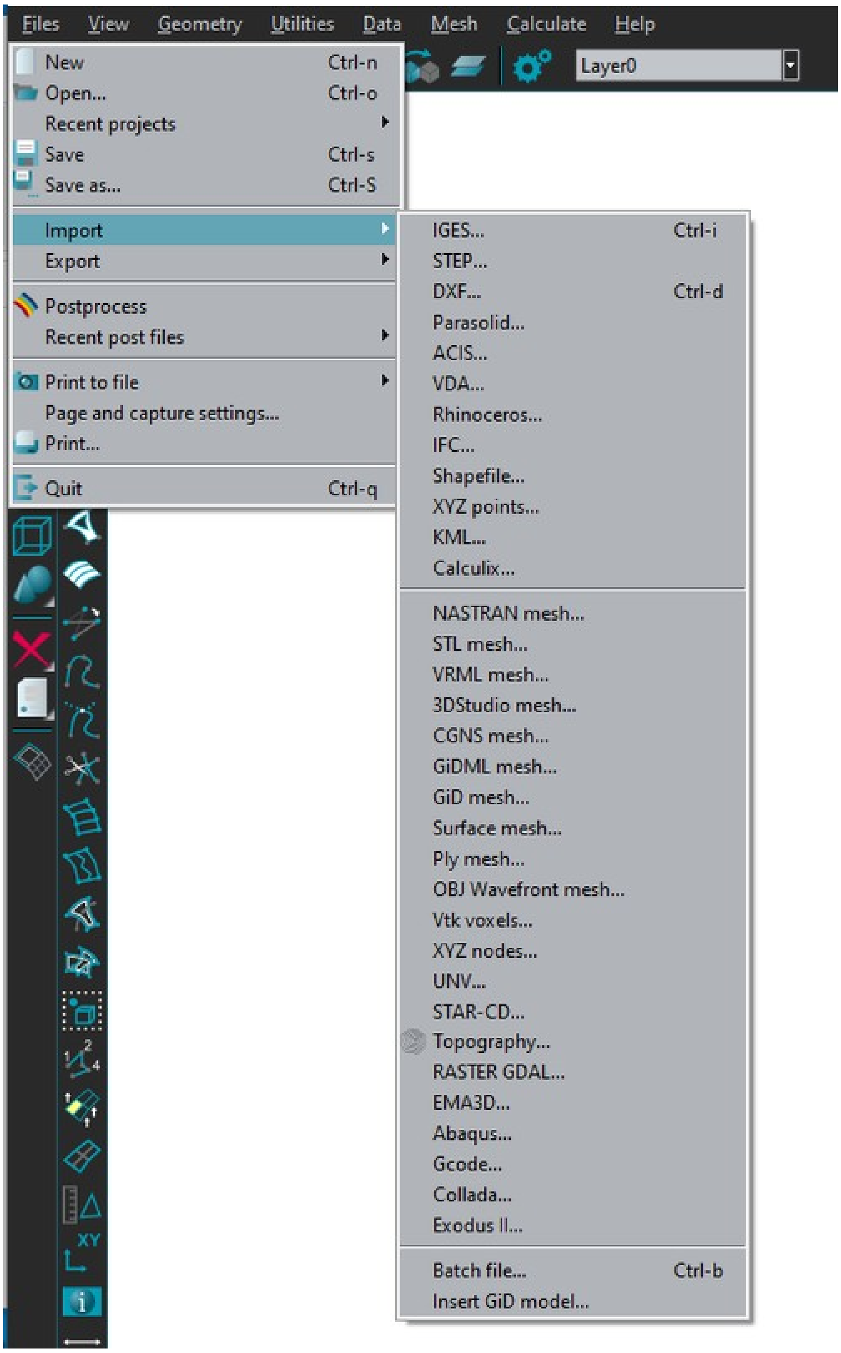
For this study an IGES formatted geometry representing a stamping die is imported. These steps are followed:

- Importing an IGES-formatted file to GiD
- Correcting errors in the imported geometry and generating the mesh
- Generating a conformal mesh and a non-conformal mesh



Importing on GiD

GiD is designed to import a variety of file formats. Among them are standard formats such as IGES, DXF, or VDA, which are generated by most CAD programs. GiD can also import meshes generated by other programs, e.g. in NASTRAN or STL formats.

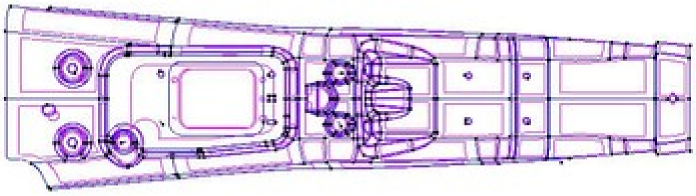


The file importing process is not always error-free. Sometimes the original file has incompatibilities with the format required by GiD. These incompatibilities must be overcome manually. This example deals with various solutions to the difficulties that may arise during the importing process.

Importing an IGES file

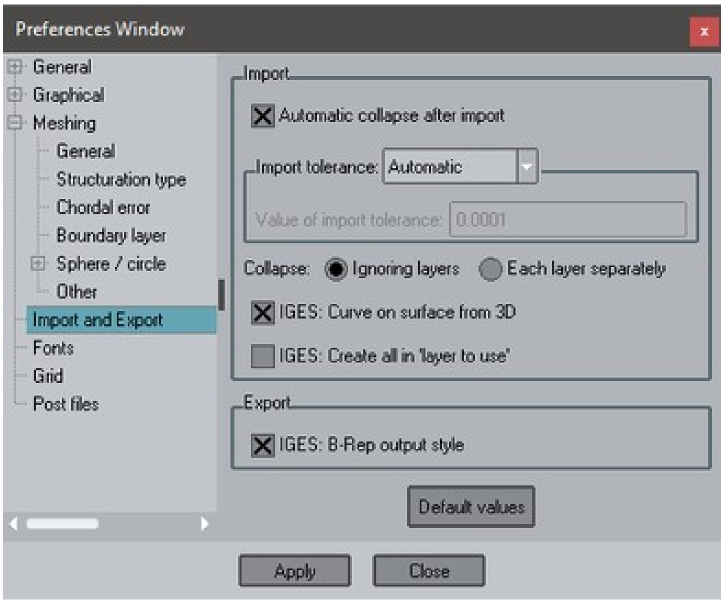
- Select **Files->Import->IGES ...**
- Select the IGES-formatted file "base.igs" and click **Open**.

After the importing process, the IGES file that GiD has imported appears on the screen.



NOTE: One of the operations in the importing process is repairing and collapsing the model. We say that two entities collapse when, the distance between them being less than the **Import Tolerance**, they become one.

The **Import Tolerance** value may be modified by going to the **Utilities** menu, opening **Preferences**, and selecting **Import and Export** from the tree. By default, the I**mport tolerance automatic value** is selected. With this option selected, GiD computes an appropriate value for the **Import tolerance** based on the size of the current geometry.
Collapsing the model may also be done manually. This option is found in **Geometry->Edit->Collapse->Model**.



Preferences Window

Import

☒ Automatic collapse after import

Import tolerance: Automatic

Value of import tolerance: 0.0001

Collapse: ● Ignoring layers ○ Each layer separately

☒ IGES: Curve on surface from 3D

☐ IGES: Create all in 'layer to use'

Export

☒ IGES: B-Rep output style

Correcting errors in the imported geometry

The great diversity of versions, formats, and programs frequently results in differences between the original and the imported geometry. With GiD these differences might give rise to imperfect meshes or prevent meshing altogether. In this section we will see how to detect errors in the imported geometry and how to correct them.

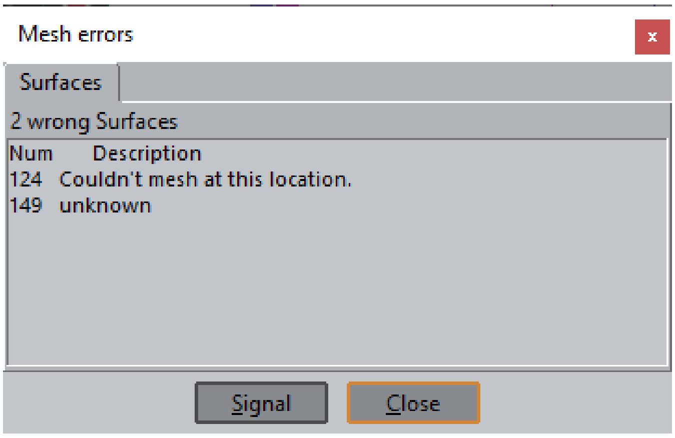
Importing the same file with different versions of GiD might produce slight variations in the results. For this reason from now we will use a project that contains the original IGES file translated into GiD format by an oldGiD version (in fact importing it with current versions will automatically fix some of the problems)

- Select **Files->Open...**
- If a dialog window appear asking to save changes to the project, click **No**.
- Select the project **"imported48.gid".**

Meshing by default

- Select **Mesh->Generate Mesh**.

A window comes up in which to enter the maximum element size for the mesh to be generated. Leave the default value provided by GiD unaltered and click **OK**.
When the **GiD** finishes the meshing process, an error message appears. This error is due to a defect in the imported geometry. As the window shows, there have been errors meshing surface number 124.



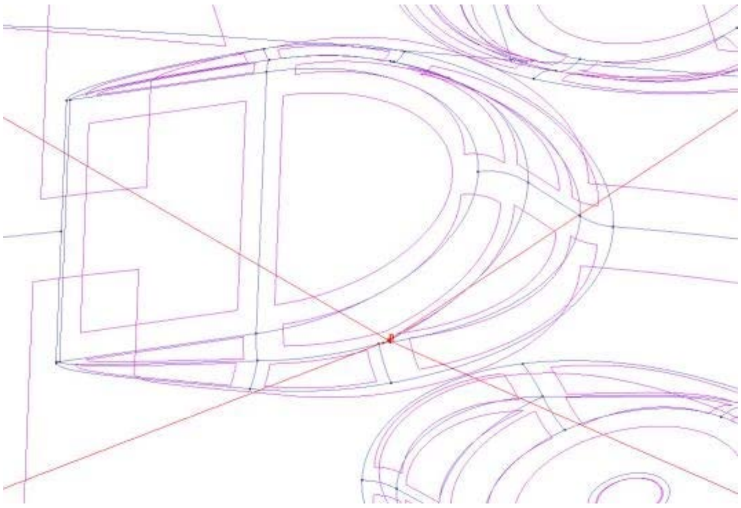
2 wrong Surfaces

```
Num     Description
124  Couldn't mesh at this location.
149  unknown
```

In this part of the tutorial we focus on repairing the surface number 124.
To locate surface 124, select the line with its message in the dialog box and press the **Signal** button (the same effect is obtained by double-clicking over the message with the left mouse button).

NOTE : If user clicks the right button over a message in the Mesh Errors window, three options are displayed: "Signal problematic point", "More help…" or "List…" The first option is the same as the Signal button, while the "List…" option presents a list of the problematic geometrical entities to make selection easier when performing some common procedures (like sending the entities to a separate layer, erasing the entities, etc.). The "More help…" option gives advice about to correct the geometrical model so the mesh can be generated.

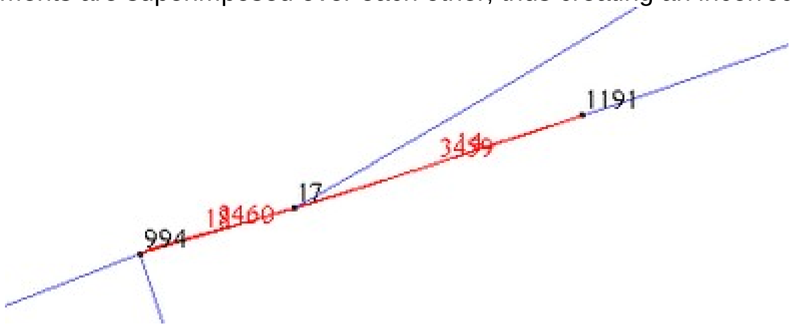
NOTE : *The* **Mesh Errors** *window can be recovered while dealing with the model by selecting the "***Show errors…***" option in the* **Mesh** *menu*.



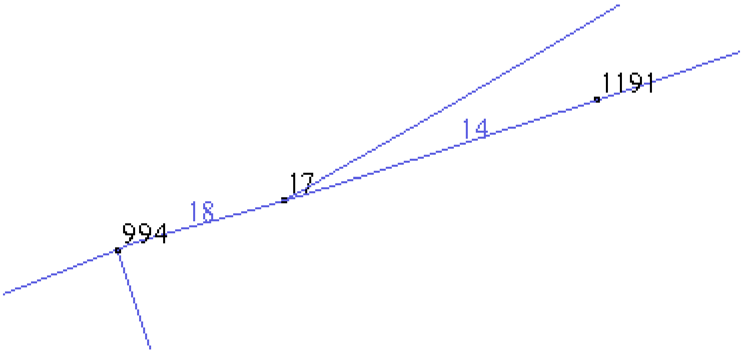
Correcting surfaces

- With the **View->Zoom->In** option in the menu or **Zoom->In** on the mouse menu, magnify the zone around surface 124.
- Set visible the labels of the lines and points on this part, with **View->Label->Select on** and selecting the lines and points of interest

- Several line segments are superimposed over each other, thus creating an incorrect surface boundary.



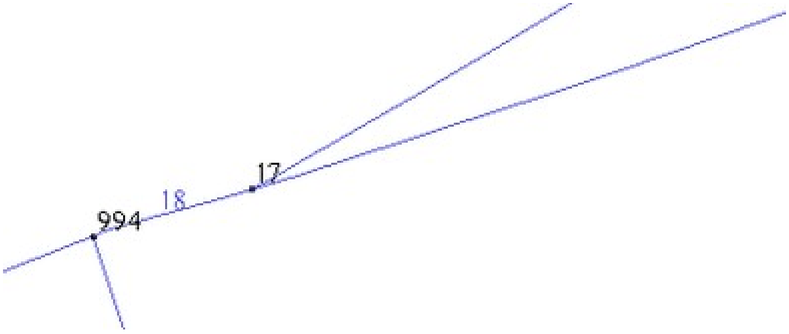
- To destroy repeated entities a local collapse may be executed. Select **Geometry->Edit->Collapse->Lines**. Then select the lines that appear in red and labeled in the previous image and press **ESC**.



- Now the four overlapped lines become only two. The point 1191 join two curves (14 and 133) that are aligned, it is possible to join them in a single curve, to avoid short curves and to facilitate future selections having less entities. We can do it only for these curves, but we will do it for all lines

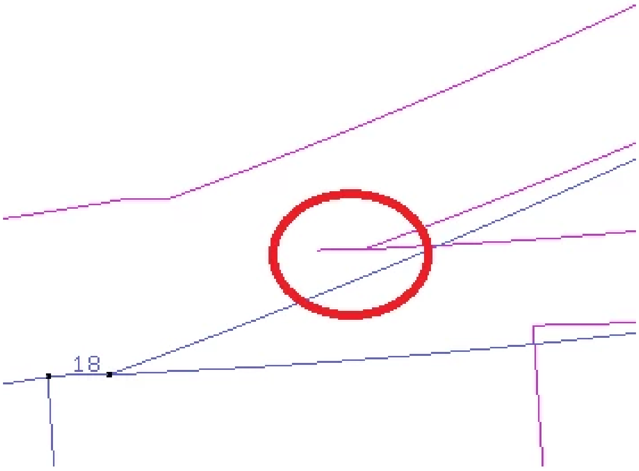
Geometry->Edit->Join->Lines
and select all in screen (or write 1:end in the command line) and press **ESC**

only will be joined the lines that are 'near tangents with a tolerance'

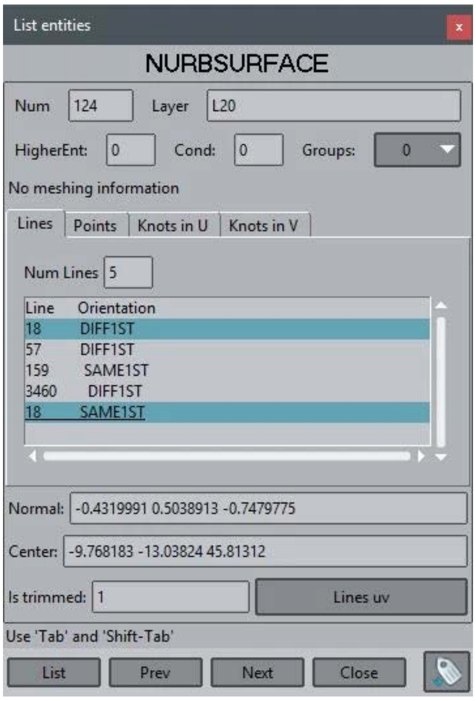


- The surface boundary 124 is wrong, because has repeated the line 18.

We can confirm it listing its data with **Utilities->List->Surfaces**



Wrong surface (repeated lines without inner space between them)


List entities

NURBSURFACE

Num: 124 Layer: L20

HigherEnt: 0 Cond: 0 Groups: 0

No meshing information

Num Lines: 5

Line	Orientation
18	DIFF1ST
57	DIFF1ST
159	SAME1ST
3460	DIFF1ST
18	SAME1ST

Normal: -0.4319991 0.5038913 -0.7479775

Center: -9.768183 -13.03824 45.81312

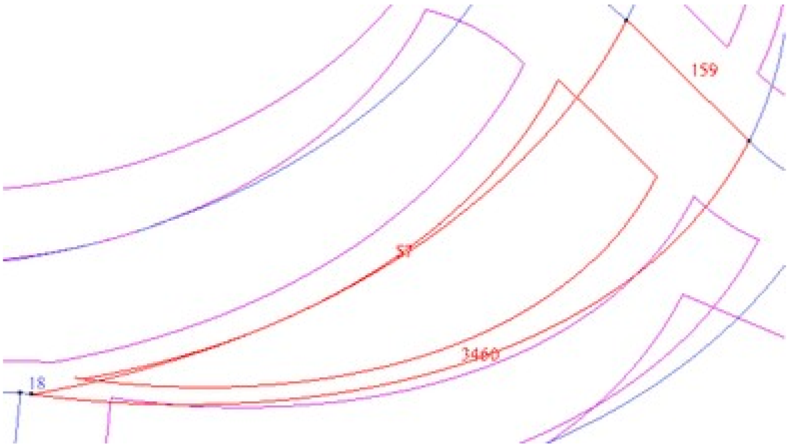
Is trimmed: 1

Use 'Tab' and 'Shift-Tab'

Information of the surface boundary lines and its orientation

A possible solution is to replace this surface by a new surface with the same underlying shape and a new trimming boundary (lines 57, 159 and 3460).

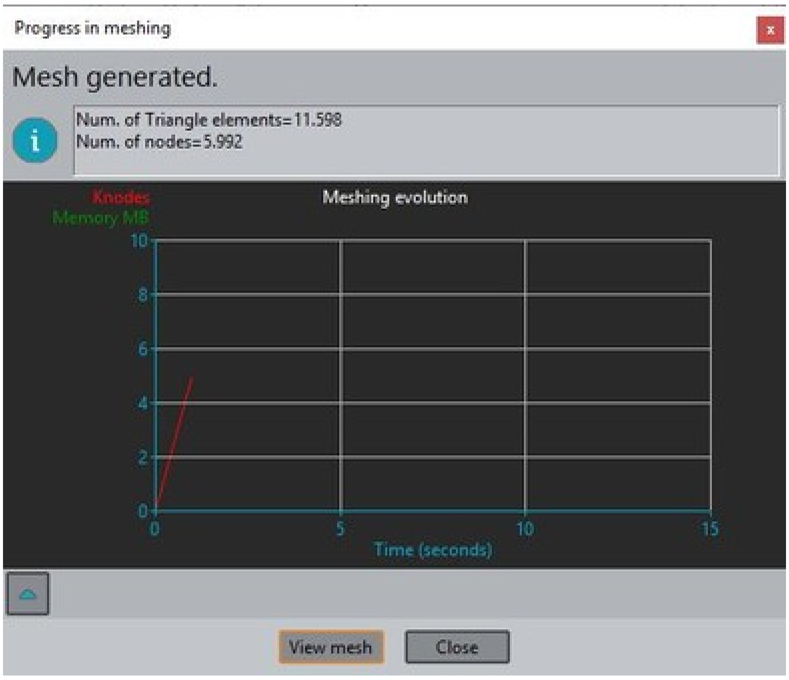
- Select **Geometry->Create->NURBS surface->Trimmed**. Select surface 124. Then select the lines in red to define its boundary. Press **ESC** twice.



Now we have two overlapped surfaces, the old one num 124 must be deleted.

1. Select **Geometry->Delete->Surfaces**. Select surface 124 and press **ESC**.
2. Do the same with the surface 149 (create a new trimmed surface with the appropriated boundary and after delete the bad one)
3. Mesh the geometry again with **Mesh->Generate Mesh**.
4. A window comes up in which to enter the maximum element size for the mesh to be generated. Leave the default value provided by GiD and click **OK**.

The mesh generating process may be carried out with no further errors found.

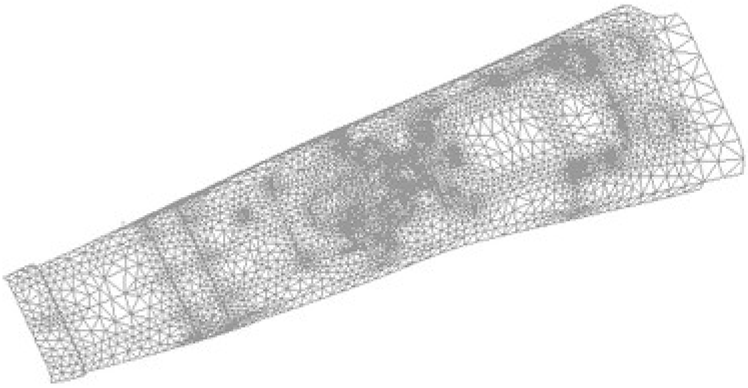


Progress in meshing

Mesh generated.

Num. of Triangle elements=11.598
Num. of nodes=5.992

- The imported piece is now completely meshed.

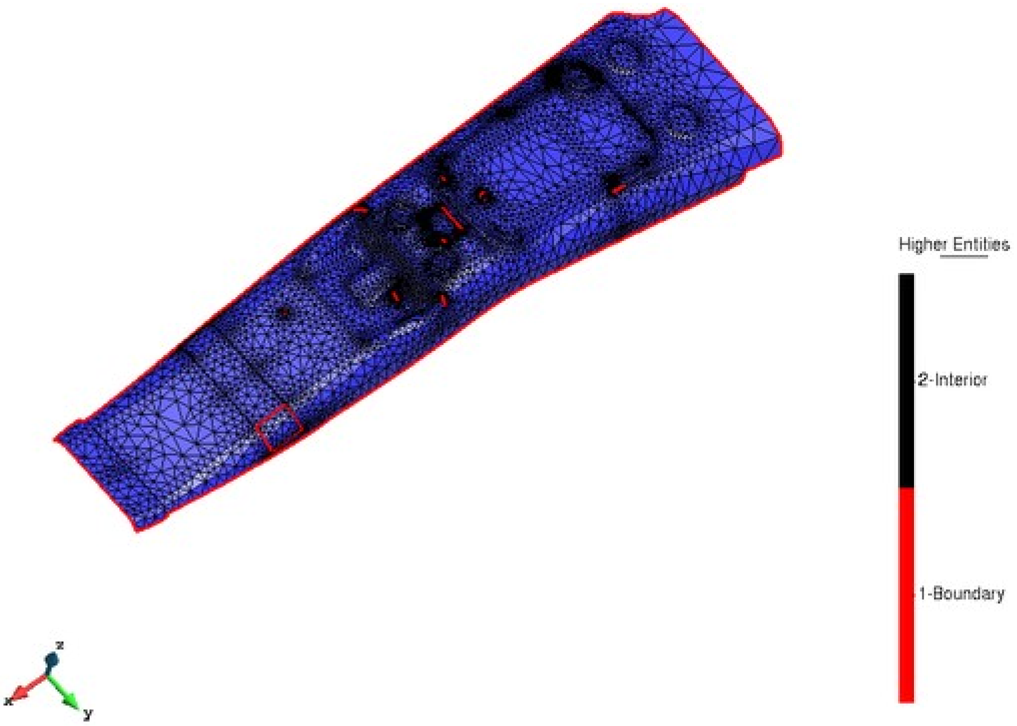


The conformal mesh and the non-conformal mesh

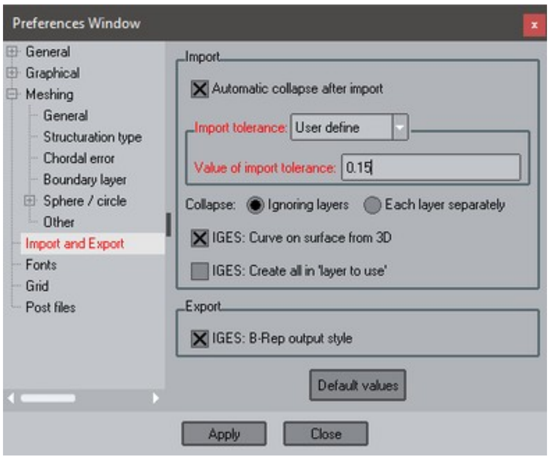
In the previous section, after correcting some errors, we were able to mesh the imported geometry, but the mesh has problems for simulations because it is a non-conformal mesh. A conformal mesh is one in which the elements share nodes and sides. To achieve this condition, contiguous surfaces (of the piece) must share lines and points of the mesh. Most calculating modules require conformal meshes; however, some modules accept non-conformal meshes. A non-conformal mesh normally requires less computation time since it generates fewer elements.

Global collapse of the model

- After generating the mesh, select **View->Higher entities->Edges**.



- Visualization of higher entities of edges shows that in the interior of the piece some surfaces are isolated.
- Press **ESC** to finish higher entities visualization.
- To generate a conformal mesh, first execute a global collapse of the model.
- The GiD collapse depends upon the **Import tolerance**. Two entities are collapsed (converted into one) when they are separated by a distance less than the **Import tolerance** parameter. To test this, enter a new value for the **Import tolerance** parameter.
- Go to **Utilities->Preferences**, and select **Import and Export** branch. Set the **Import tolerance** to **User define value** and enter 0.15 for the **Value of import tolerance**. Click **Apply**.



Import

☒ Automatic collapse after import

Import tolerance: User define

Value of import tolerance: 0.15

Collapse: ● Ignoring layers ○ Each layer separately

☒ IGES: Curve on surface from 3D

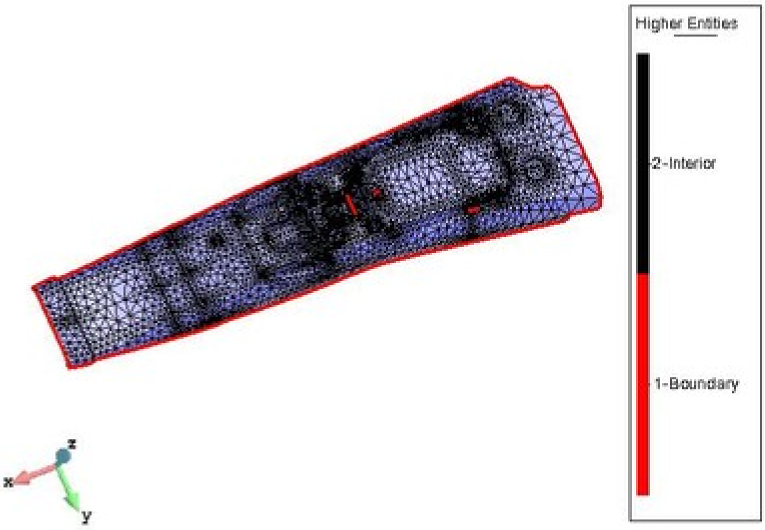
□ IGES: Create all in 'layer to use'

Export

☒ IGES: B-Rep output style

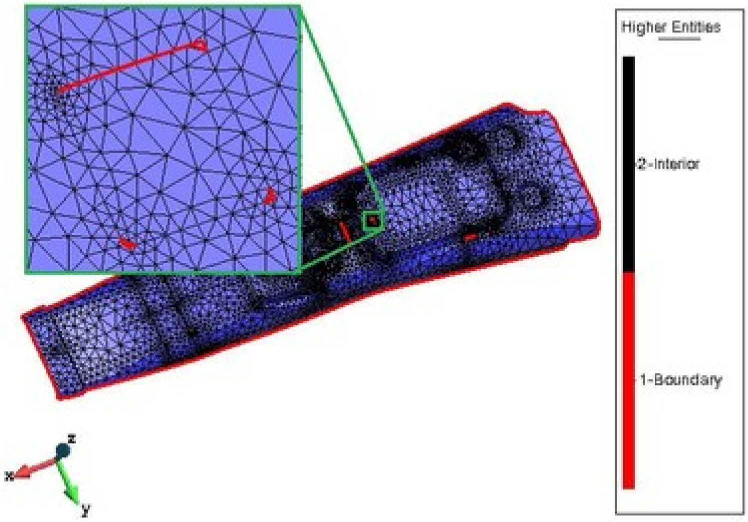
- Select **Geometry->Edit->Collapse->Model**.
- A dialog window appears to confirm the selection. Click **Ok**.
- Select **Mesh->Generate mesh**. Erase the old mesh and use the default element size.
- Visualize the results with **View->Higher entities->Edges**.

Some of the contiguous surfaces in the interior of the model have now being joined. However, there are still some surfaces that prevent the mesh from being completely conformal. These surfaces must be modified manually.

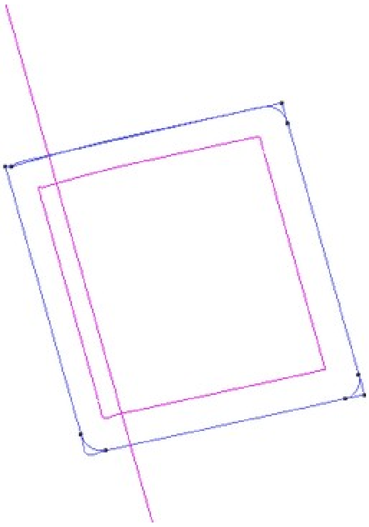


Correcting surfaces and creating a conformal mesh

- With the option **View->Zoom In**, magnify the zone illustrated.

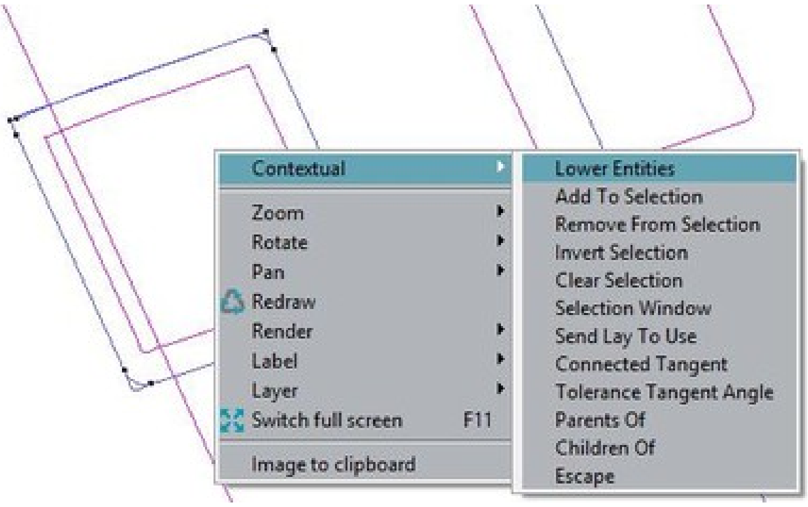


- Select **View->Mode->Geometry** to visualize the geometry of the piece.

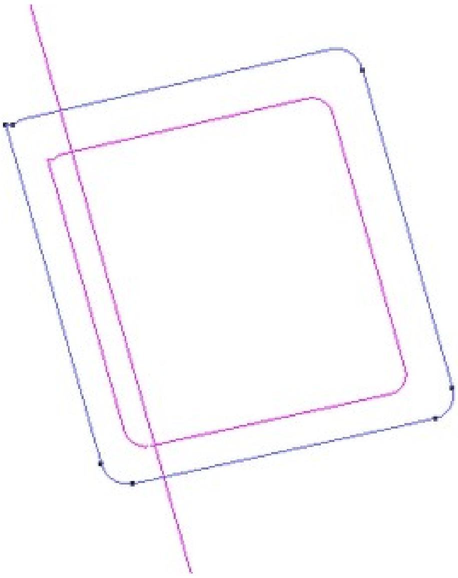


There is a rectangular surface that does not fit within the boundaries of a rounded-corner surface (a hole, in this case). We will suppose that the problematic surface is planar. This way, it can be erased and recreated only with the information of its boundary in order to fit the rounded-corner boundary.

- Select **Geometry->Delete->Surfaces**. Select the problematic surface, but before pressing **ESC**. Go to **Contextual** menu and select **Lower Entities.** With this option, the surface and lines and points that belongs to only this surface will be deleted.



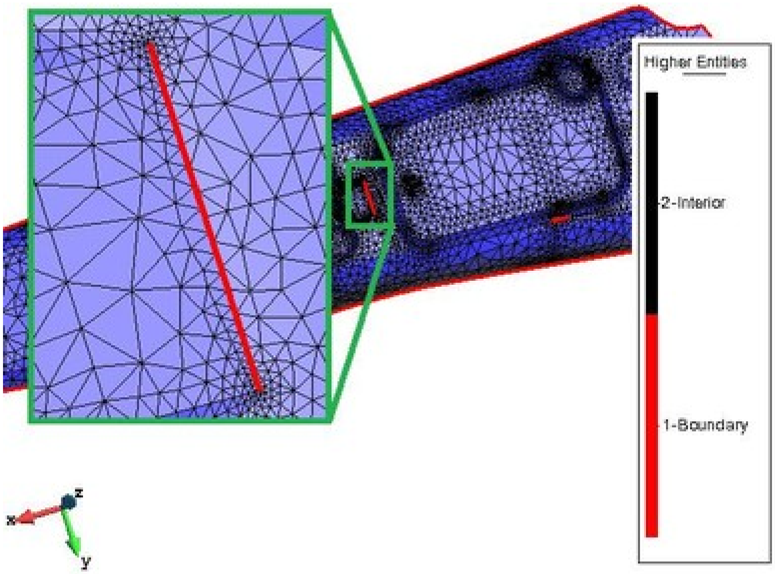
- With **Geometry->Create->NURBS surface->By contour** create a new surface. Select the lines defining the contour and press ESC.



- Visualize the mesh again **View->Mode->Mesh** You will see that the mesh of the deleted surface is also deleted, but a new surface is created and it is not meshed. If you want to see the results of the first correction, the mesh must be regenerated with **Mesh->Generate mesh**. The problem has been fixed on this part.

Note: only the mesh of the new surface will be really generated, the mesh of the rest of surfaces is cached to save resources

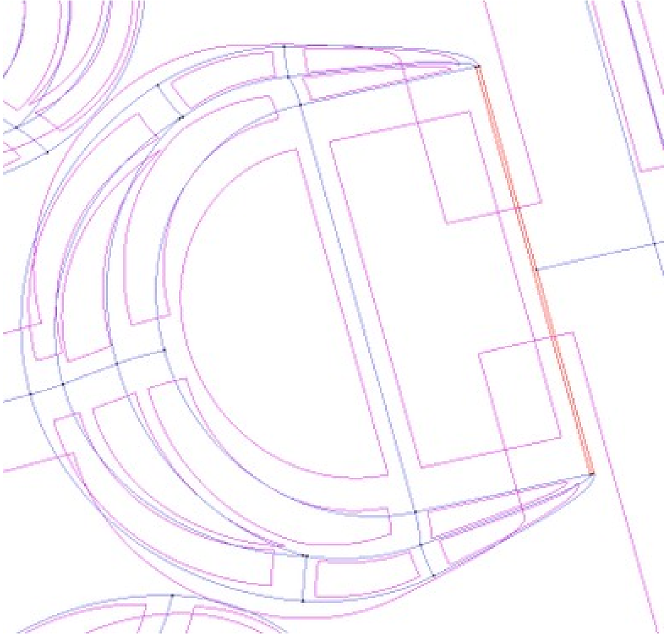
- Use **View->Higher entities->Edges** on mesh mode, and magnify the next problematic zone (see image).



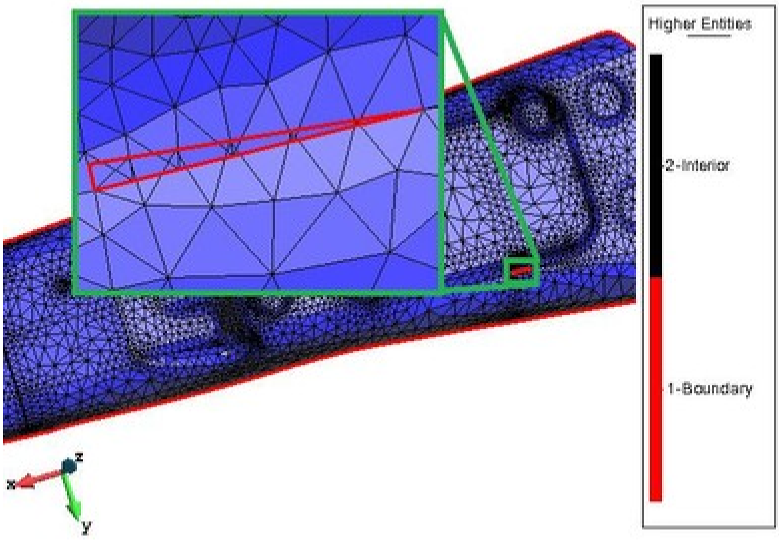
- Select **View->Mode->Geometry**.

In this example, the situation involves a contour of four lines that does not correspond to any real surface (of the piece). These lines were too far apart to be collapsed.

- Select **Geometry->Create->NURBS surface->By contour**. Select the lines. Press **ESC** twice.

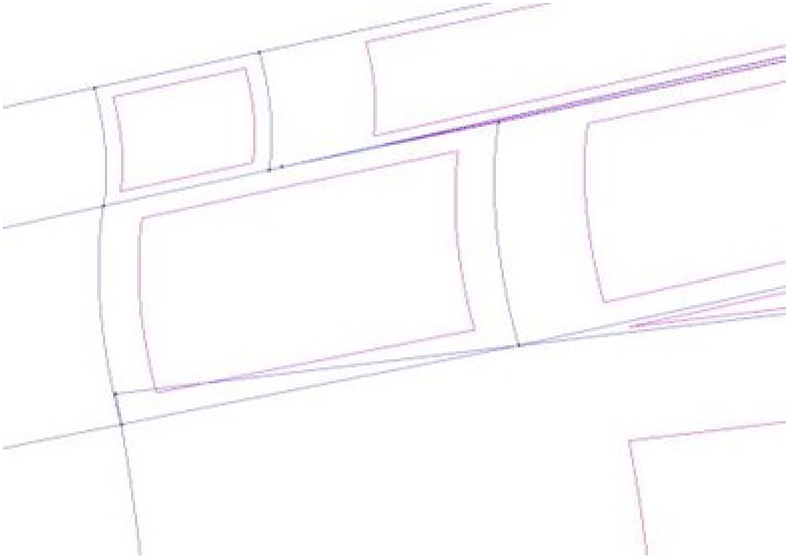


- Visualize again higher entities **View->Higher entities->Edges** and magnify the next zone indicated.



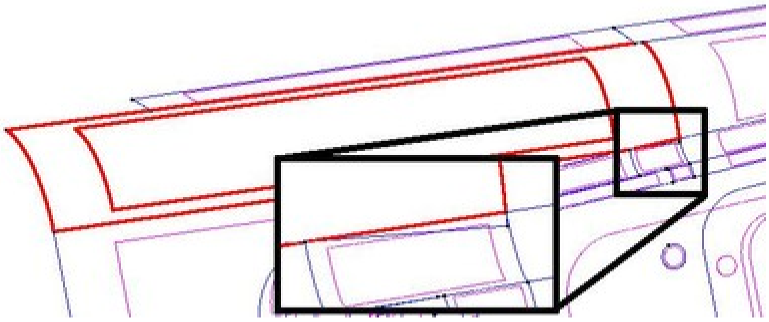
- Select **View->Mode->Geometry**.

There are two surfaces that overlap each other at one end.

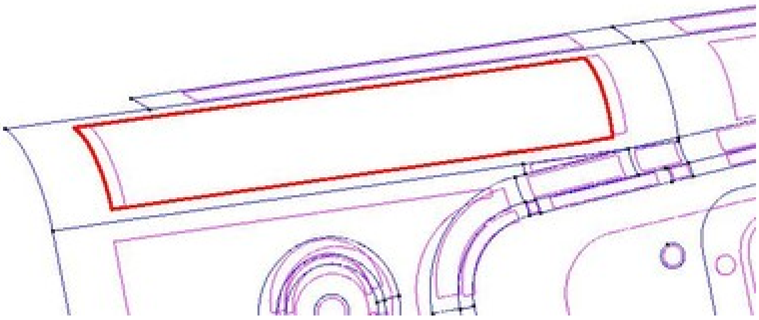


In this case the best solution for correcting the boundary is to trim the overlap.

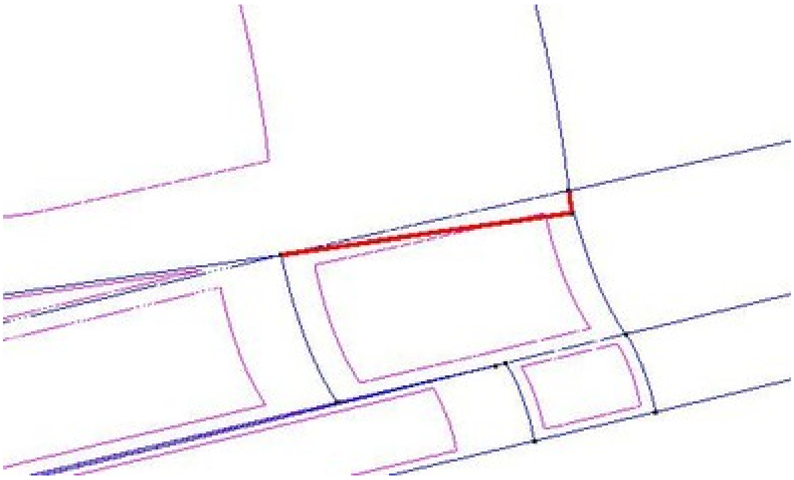
- Select **Geometry->Create->NURBS surface->Trimmed**.
- Select the surface to be trimmed. Then select the new boundary.



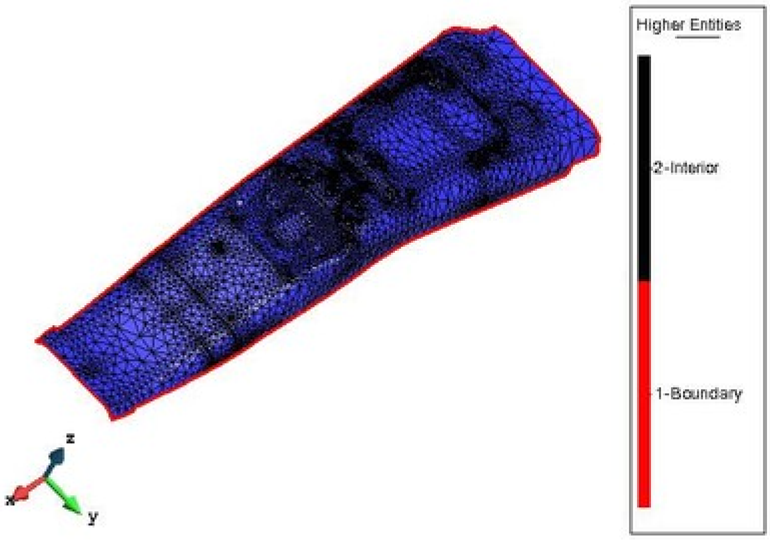
- Select **Geometry->Delete->Surfaces**. Select the original surface. Press **ESC** twice.



- Use **Geometry->Delete->Lines** and select all visible lines, and after delete the points with **Geometry->Delete->Points** and select all visible points. Only the isolated lines and isolated points will be deleted (lines that belong to surfaces or points that belong to curves won't be delete, also if they were selected).



- Select **Mesh->Generate mesh**. Then visualize the result using the option **View->Higher entities->Edges**



A conformal mesh has been achieved, all edges are interior, higher entities=2, except the ones on the boundary with higher entities=1.

There is a node that don't belong to any element, this is because an isolated point that don't belong to any line. This point can be deleted with

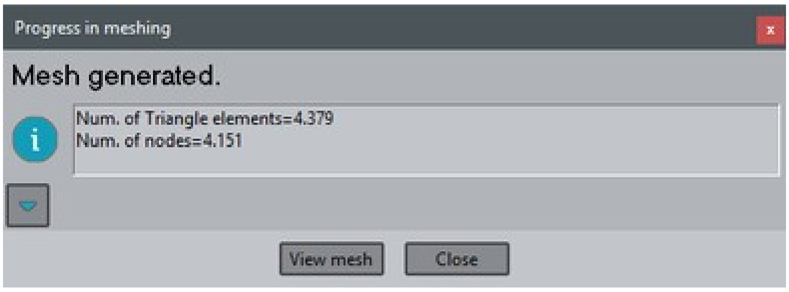
Geometry->Delete->Points

and can select all the points, only the ones with higherenties=0 will be allowed to be deleted.

Creating a non-conformal mesh

NOTE: Non-conformal meshes may be used with some calculating modules, i.e. stamping a plate. Using non-conformal meshes significantly reduces the number of elements in the mesh. This cuts down on computation time.

- Select **View->Mode->Geometry**.
- Select **Geometry->Edit->Uncollapse->Surfaces**. Select all the surfaces in the model. Press **ESC**. A sufficient number of lines and points is created so that no surface (of the object) shares lines with any contiguous surface.
- Select **Mesh->Generate Mesh**. When the mesh has been generated, a window appears with information about the mesh (Figure 29). The result is a non-conformal mesh composed of far fewer elements than the meshes generated in the previous section: about 4.000 elements instead of the 10.000 needed to generate the conformal mesh.

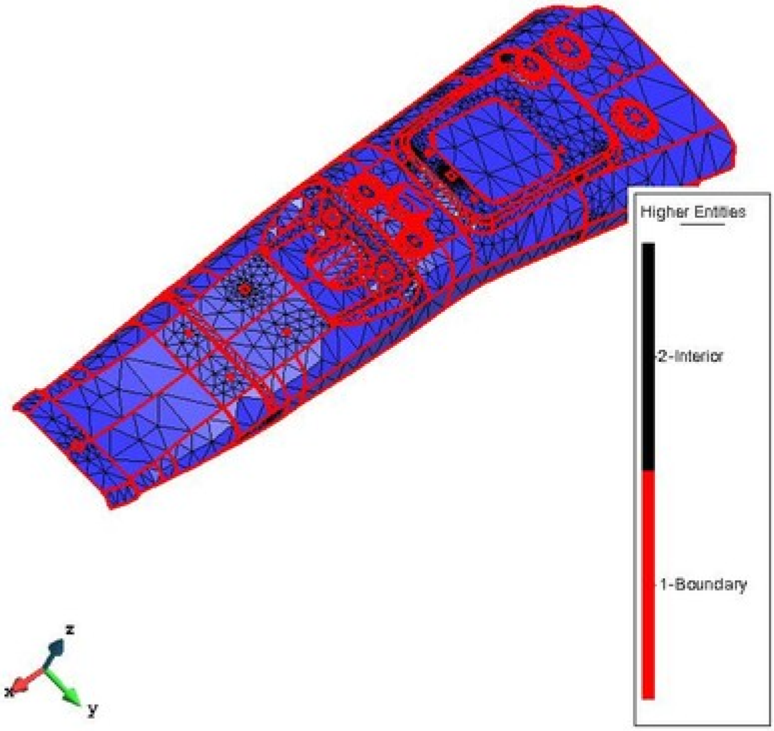


Progress in meshing

Mesh generated.

Num. of Triangle elements=4.379
Num. of nodes=4.151

- Visualize the result using **Mesh->View mesh boundary**.



Now each surface is independent of the other surfaces.

DEFINING A PROBLEM TYPE

This tutorial takes you through the steps involved in defining a problem type using GiD. A problem type is a set of files configured by a solver developer so that the program can prepare data to be analyzed.
A simple example has been chosen which takes us through all the associated configuration files while using few lines of code. Particular emphasis is given to the calculation of the centers of mass for two-dimensional surfaces a simple formulation both conceptually and numerically.

By the end of the example, you should be able to create a calculating module that will interpret the mesh generated in GiD Preprocess. The module will calculate values for each element of the mesh and store the values in a file in such a way as they can be read by GiD Post-process.

Our aim is to solve a problem that involves calculating the center of gravity (center of mass) of a 2D object. To do this, we need to develop a calculating module that can interact with GiD.
The problem: calculate the center of mass.
The center of mass (X_{CM},Y_{CM}) of a two-dimensional body is defined as

$$x_{CM} = \frac{{}_{S}\rho \;\; x \;\; x \;\; y + {}_{i=1}^{N}\, m_i \;\; x_i}{{}_{S}\rho \;\; x \;\; y + {}_{i=1}^{N}\, m_i} \qquad y_{CM} = \frac{{}_{S}\rho \;\; y \;\; x \;\; y + {}_{i=1}^{N}\, m_i \;\; y_i}{{}_{S}\rho \;\; x \;\; y + {}_{i=1}^{N}\, m_i}$$

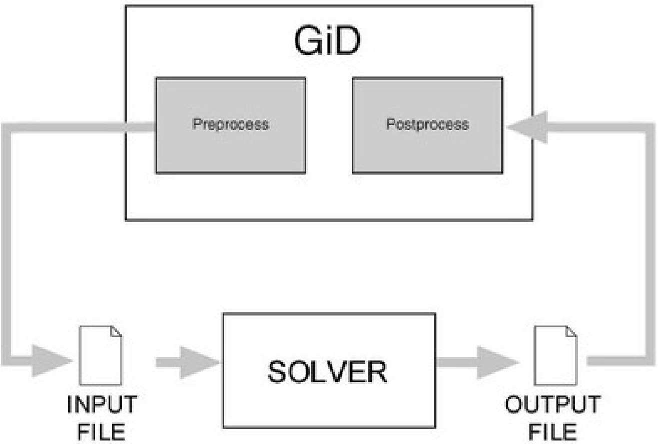
where (x,y) is the density of the material at point (x,y) and S is the surface of the body; mi are concentrated masses applied on the point (x_i,y_i).
To solve the problem numerically, the integrals will be transformed into sums:

$$x_{CM} = \frac{{}_{elm}\rho_{elm} V_{elm} \;\; x_{elm} + {}_{i=1}^{N}\, m_i \;\; x_i}{{}_{elm}\rho_{elm} V_{elm} + {}_{i=1}^{N}\, m_i} \qquad y_{CM} = \frac{{}_{elm}\rho_{elm} V_{elm} \;\; y_{elm} + {}_{i=1}^{N}\, m_i \;\; y_i}{{}_{elm}\rho_{elm} V_{elm} + {}_{i=1}^{N}\, m_i}$$

Each of the N elements is treated as concentrated weight whose mass is defined as the product of the (surface) density and the area of the element.

Interaction of GiD with the calculating module

GiD Preprocess makes a discretization of the object under study and generates a mesh of elements, each one of which is assigned a material and some conditions. This preprocessing information in GiD (mesh, materials, and conditions) enables the calculating module to generate results. For the present example, the calculating module will find the distance of each element relative to the center of mass of the object.
Finally, the results generated by the calculating module will be read and visualized in GiD Post-process.



GiD must adapt these data to deal with them. Materials, boundary and/or load conditions, and general problem data must be defined.
GiD configuration is accomplished through text formatted files.

From the 13th version of GiD, a new problemtype system has been developed (based on the customlib library). It offers more powerful integration tools, and a more efficient and user-friendly way to interact with the data. It uses a single .spd file to describe general properties, materials, conditions and units (as a tree with xml syntax). All this data is shown in a 'tree view', and materials and conditions are associated to groups of entities.

The 'classic' problemtypes system is currently also supported by GiD, but it is considered deprecated, as the new system offers many advantages. Information about how to create a problemtype using the 'classic style' can be found in the Annex of this manual.

The calculating module (in this example cmas2d.exe) solves the equations in the problem and saves the results in the results file. This module may be programmed in the language of your choice, 'C' in used in this example.

GiD Post-process reads the following files generated by the calculating module:
project_name.post.res: results file.

Each element of the mesh corresponds to a value.
project_name.post.msh: file containing the post-process mesh. If this file does not exist, GiD uses the preprocess mesh also for postprocess.

Implementation

A problemtype is composed by these files (most are not compulsory):

Configuration files

- <problem_type_name>.spd: Xml file defining conditions, materials, problem data and units definitions

Script files

- <problem_type_name>.tcl : Write data input file with a Tcl procedure

Tcl extension files

- <problem_type_name>.tcl Extensions to GiD written in the Tcl/Tk programming language

Command execution files

- <problem_type_name>.bat Operating system shell that executes the analysis process

Cmas2d example

Let's see our example: the **cmas2d_customlib** problemtype.
The problem-type is located in problemtypes/Examples/cmas2d_customlib.gid
Inside this folder, we can find several files. The most important ones are:

- cmas2d_customlib.xml

Declares some information about the problemtype, such as the name, the minimum GiD version, a description, etc.

- cmas2d_customlib.spd

A xml file that defines the tree interface.

- cmas2d_customlib.tcl

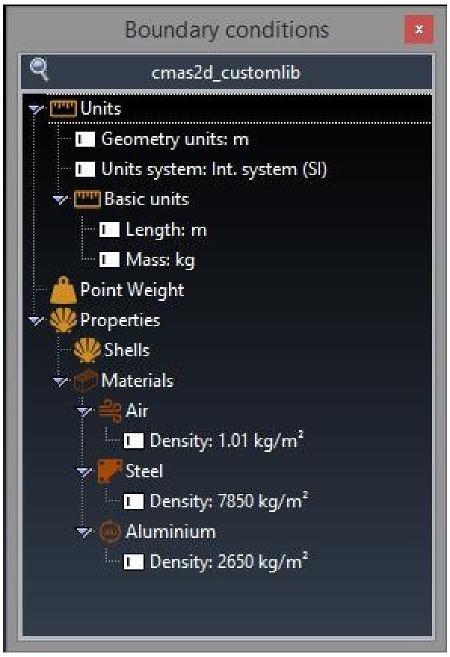
Contains Tcl code to process some GiD events, add some menu entries and the functions to write the input file.
This file also includes more tcl scripts files under *cmas2d_customlib.gid/scripts/* folder .

- cmas2d_customlib.win.bat | cmas2d_customlib.unix.bat

Script that launches the solver in MS Windows or in Linux/macOS .

Interface definition

In this section, we are going to prepare the interface definition document, called **cmas2d_customlib_default.spd**.
This file is in XML format and contains all the definition of all the data necessary for the analysis.
First of all, let's see the final result:



- Units
  - Geometry units: m
  - Units system: Int. system (SI)
  - Basic units
    - Length: m
    - Mass: kg
- Point Weight
- Properties
  - Shells
  - Materials
    - Air
      - Density: 1.01 kg/m²
    - Steel
      - Density: 7850 kg/m²
    - Aluminium
      - Density: 2650 kg/m²

Step by step. The first we see is **Units**. It's useful to set a global criteria, such as the geometry units, or the default units system.

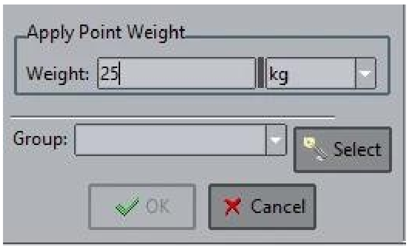
The next is **Point Weight**, to assign concentrated mass to a group that contains points. The spd code for this condition is:

```
<condition n="Point_Weight" pn="Point Weight" ov="point" ovm="node"
icon="constraints" help="Concentrated mass">
    <value n="Weight" pn="Weight" v="0.0" unit_magnitude="M" units="kg"
help="Specify the weight that you want to apply"/>
    <symbol proc="gid_groups_conds::draw_symbol_image weight-18.png"
orientation="global"/>
</condition>
```

Let's introduce some concepts. The **'condition'** tag is used to assign properties to groups. The properties are defined in the **'value'** tags inside the container. For example, there is the property called *'Weight'.* We can specify which entity is allowed (point, line, surface and/or volume) in the **'ov'** field of the condition. For further information, check the customLib description of fields in the Customization Manual.

the **'symbol'** tag allow to draw an image to represent the condition applied, in this case from a raster image file in PNG format.

When we 'double click' on Point Weight, we can see this window, to assign a weight to a group, that can be created by clicking on the Select button. This will allow us to select some points in the geometry. The condition will be applied after selection the points and clicking the Ok button.



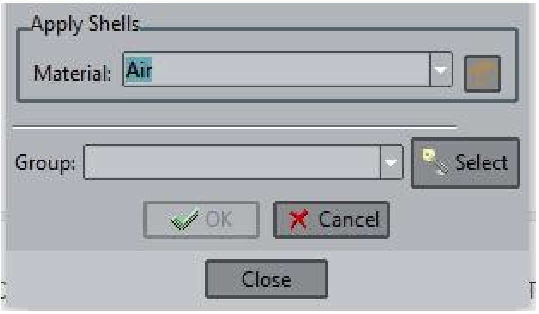
Weight: 25 kg

Group:

In the tree, below the *'Point Weight'* entry, we find **'Properties',** a folder or 'container', that contains **'Shells'** and **'Materials'**. It's code is:

```
<container n="Properties" pn="Properties" icon="darkorange-shellfish- 18"
help="Define your materials database and apply them to the surfacesof your
problem">
<condition n="Shells" pn="Shells" ov="surface" ovm="element"
ov_element_types="triangle" icon="darkorange-shellfish-18" groups_icon="
yelowish-group" help="Select your material and the surfaces related to it">
<value n="material" pn="Material" editable='0' help="Choose a material
from the database" values='[Cmas2d::GetMaterialsList %W]' v="Air">
<edit_command n="Edit materials" pn="Edit materials" icon="darkorange-
block1.png" proc='Cmas2d::EditDatabaseListDirect %W %DICT %BC'/>
</value>
</condition>
<include path="xml/materials.xml"/>
</container>
```

There is a 'container', another 'condition' called Shells, and a special 'value' called material. In this section, we want to assign a material from the database to a surface (see 'ov' field on the condition).
By 'double clicking' on Shells, we get a window like this:



Material: Air

After selecting the surfaces you want to assign this condition, press the Ok button to confirm this assignment.

In the Properties container code, we can see that it includes a file. The customLib library allows splitting the spd in different parts. You can find the materials database in the included file, in the same problemtype folder.

Writing the calculation files

Once a geometry is created and properties from the tree assigned, it is time to write the input file for the cmas2d solver.

For this example we will need to write the following information in the file:

- Number of elements and nodes of the model,
- Coordinates of the nodes of the mesh,
- Connectivities of the elements,
- Number of materials used,
- Density of each material used,
- Number of point conditions,
- Weight assigned to each point.

Let's open the *cmas2d_customlib.tcl* file and see how are we processing the event of GiD that is called when the user wants to calculate: **GiD_Event_AfterWriteCalculationFile**. After a few check of the environment, *'Cmas2d::AfterWriteCalculationFile $filename* ' is called (It is defined in *scripts/writing.tcl*).

First we need to do in this function is to call some initialization procedures:
To open the file for writing:

```
customlib::InitWriteFile $filename
```

To initialize the material's database, indicating which 'conditions' have materials assigned.

```
customlib::InitMaterials [list "Shells"] active
```

Then we write some headers and to **write the number of elements and nodes**, we call some GiD_Info Functions:

```
customlib::WriteString "[GiD_Info Mesh NumElements] [GiD_Info Mesh
NumNodes]"
```

To **write the nodes and their coordinates** we need to prepare the format for an integer node id and two reals x, y (z is omitted)

```
customlib::WriteCoordinates "%5d %14.5e %14.5e%.0s\n"
```

As we can see, the format is prepared to write 2D coordinates (X & Y).

Next we need to **write are the connectivities of the elements**. For each element, we want to write it's id, it's nodes, and the material id that we assigned. In order to do this, again we prepare the parameters for the function WriteConnectivities:

```
set elements_conditions [list "Shells"]
set element_formats [list {"%10d" "element" "id"} {"%10d" "element"
"connectivities"} {"%10d" "material" "MID"}]
customlib::WriteConnectivities $elements_conditions $element_formats
active
```

Then, the material's block. To get and **write the number of materials**, there is a function, GetNumberOfMaterials:

```
set num_materials [customlib::GetNumberOfMaterials used]
customlib::WriteString "Nº Materials= $num_materials"
```

And, to **write the material information**, again, we need to prepare the parameters to print the material's id and it's density, and call the function WriteMaterials

```
customlib::WriteMaterials [list {"%4d" "material" "MID"} {"%13.5e"
"material" "Density"}] used
```

It is time to write the point weights. To get the number of nodes where we are applying the weights, we need to specify which is the condition we are writing, and call GetNumberOfNodes:

```
set condition_list [list "Point_Weight"]
set number_of_conditions [customlib::GetNumberOfNodes $condition_list]
```

And foreach node with a Point_Weight condition assigned, we need to **print the node id and the assigned weight.**

```
set condition_list [list "Point_Weight"]
set condition_formats [list {"%1d" "node" "id"} {"%13.5e" "property"
"Weight"}]
customlib::WriteNodes $condition_list $condition_formats
```

Finally, all we need to do is to close the writing file

```
customlib::EndWriteFile
```

Solver execution

At this point, we have generated a .dat file, containing the input data for the solver. It's time to launch it. In order to do it, we should have a .bat file (One for MS Windows and another for Linux/macOS based systems).

This script will delete the output files of previous executions (if they exist) and launch the solver executable. Check *cmas2d_customlib.win.bat* and *cmas2d_customlib.unix.bat* for more information.

The solver is located in the exec folder of the problem type. For this example, you can access to de cmas2d.c source file, and there are some compilated versions, for common architectures.

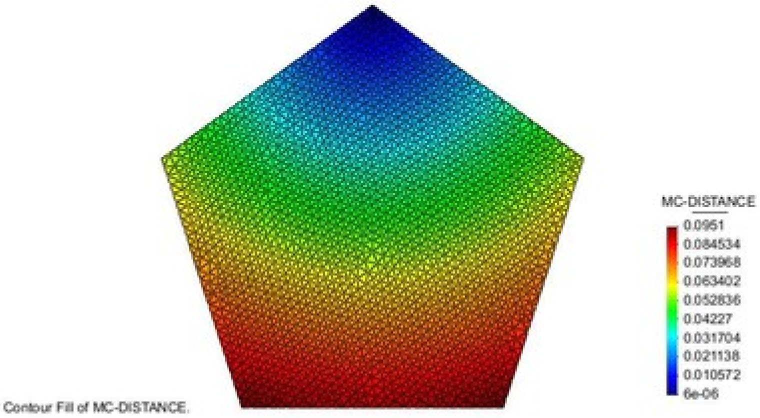
Results

After the execution, the solver has generated some files:

- {projectname}.err : With the error information (if necessary).
- {projectname}.log : With some 'log' information.
- {projectname}.post.res : With the result data.

Access the .post.res file to see the format, and check the documentation about the Postprocess data files in the Customization manual.

It's time to change to postprocess and see our results. Go to **View results** in the menu, **contour fill** > **MC-DISTANCE**
You should see something like this:

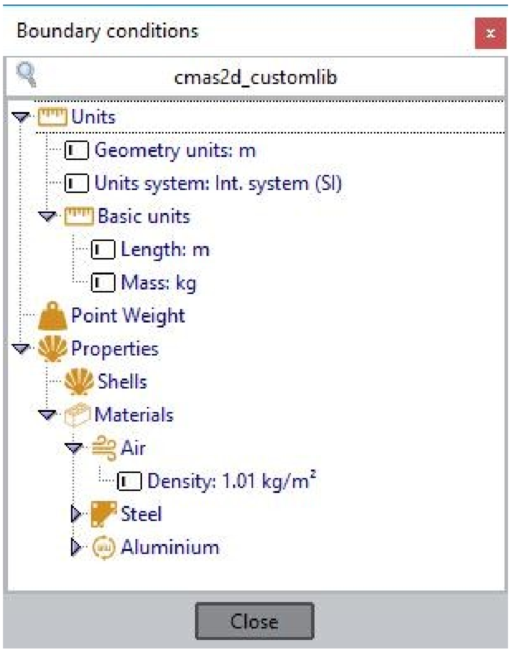


Using the problemtype with an example

We will repeat the same as for the classical use example

- Create or import the geometry (pentagon of classical case)
- Load the problemtype: menu **Data -> Problemtype -> cmas2d_customlib**

And show its tree in case it is hidden, with the menu **Data->Data tree**



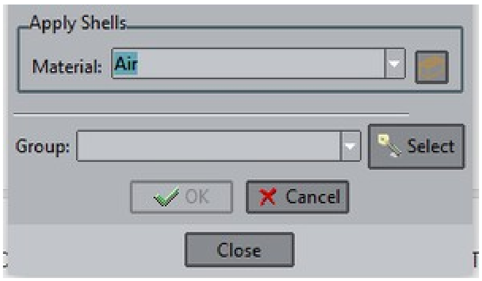
Boundary conditions

cmas2d_customlib

Units
  Geometry units: m
  Units system: Int. system (SI)
  Basic units
    Length: m
    Mass: kg
Point Weight
Properties
  Shells
  Materials
    Air
      Density: 1.01 kg/m^2
    Steel
    Aluminium

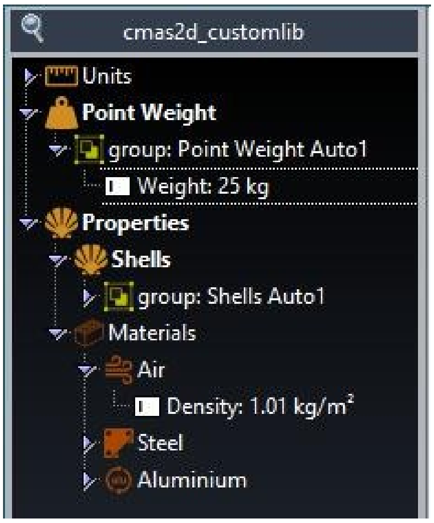
- Choose in the tree **Properties -> Shells**

A frame on the bottom of the tree will allow to select the material: Air and expect to be assigned to groups ofentities.
In general a recommendable workflow is to create first the groups of entities and to apply in a second step the calculation properties to these groups, but for simple cases it is possible to create automatically the group during a selection, pressing the Select button


Apply Shells

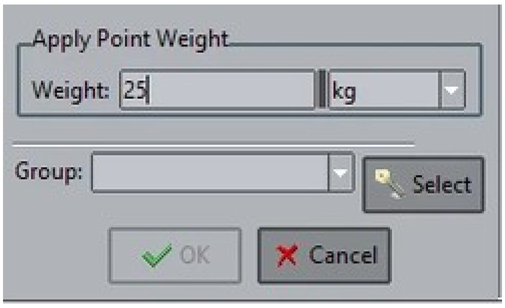
Material: Air

Group:


cmas2d_customlib

- Units
- Point Weight
  - group: Point Weight Auto1
    - Weight: 25 kg
- Properties
  - Shells
    - group: Shells Auto1
  - Materials
    - Air
      - Density: 1.01 kg/m^2
    - Steel
    - Aluminium

- Click **Select** and select the surface. Press **<Esc>** when this step is finished, and then press **Ok** to validate it (the automatic group name could be changed before its validation)
- Choose then Point Weight, and the frame to assign it to points will be showed

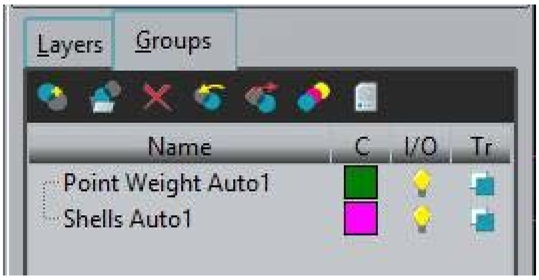

Apply Point Weight

Weight: 25 kg

Group:

- Enter the value 1e3 in the Weight box. Click **Select** and select the upper corner point. Press **<Esc>** when this step is finished, and then Ok to accept it.

with the menu: **Utilities -> Layers and groups** you can open the groups window, and see that the selection has created a couple of new groups, with automatic names based on the data tree items.



Name	C	I/O	Tr
Point Weight Auto1	🟩		
Shells Auto1	🟪		

- Choose the **Mesh -> Generate** option, use the suggested mesh size value
- Save the model with a name, to allow the calculation program write its result files in the case folder.
- Choose the **Calculate** option from the Calculate menu to start the calculation module.
- Wait until a box appears indicating the calculation has finished.
- Select the option **Files-> Postprocess**.
- Visualize the desired result like the classical example.

Additional information

NOTE: In this example, a code for the program will be developed in C. Nevertheless, any programming language may be used.
The code of the program that calculates the center of mass (cmas2d.c) is as follows:

The cmas2d.c file

```
#include <stdio.h>
#include <stdlib.h>
#include <malloc.h>
#include <math.h>

#define MAXMAT 1000
#define MAXCND 1000
char projname[1024];
int i, ielem, inod, icnd;
double *x, *y;
int *N, *imat;
int  nodc[MAXCND];
double rho[MAXMAT], wval[MAXCND];
int Nelem, Nnod, Nmat, Ncnd;
double x_CG, y_CG;

void input(void);
void calculate(void);
void output(void);

void main( int argc, char *argv[] ) {
    strcpy( projname, argv[1] );
    input();
    calculate();
    output();
}
```

The main program

The main program is called from the cmas2d.win.bat file and has as parameter the name of the project. This name is stored in the variable projname.
The main program calls the input (), calculate () and output () functions.
The input function reads the .dat file generated by GiD. The .dat file contains information about the mesh. The calculate function read and processes the data and generates the results. The output function creates the results file.

```
void input () {
char filename[1024], fileerr[1024], sau1[1024], sau2[1024];
FILE *fp, *ferr;
int aux,j, error=0;
void jumpline (FILE*);
strcpy(filename, projname);
strcat(filename,".dat");
fp=fopen(filename,"r");
```

The first part of the input function concatenate the project name with the .dat extension, thus obtaining the name of the file that is to be read. This file is opened in order to be read.
The jumpline(FILE*) function is declared. This function simply reads a line from the file that it receives as a parameter, It is used to jump lines of the text when reading the .dat file.

```
for (i=0; i<6; i++) jumpline (fp);
fscanf(fp, "%d %d", &Nelem, &Nnod);
```

The first six lines of the .dat file are jumped over since these are lines of information for the user (see .bas file). Then the total number of elements and nodes of the project are read and stored in the variables Nelem and Nnod respectively.

```
x=(double *) malloc((Nnod+1)*sizeof(double)); if (x==NULL) {error=1;}
y=(double *) malloc((Nnod+1)*sizeof(double)); if (y==NULL) {error=1;}
N=(int *) malloc((Nelem+1)*3*sizeof(int)); if (N==NULL) {error=1;}
imat=(int *) malloc((Nelem+1)*sizeof(int));   if (N==NULL) {error=1;}
if (error) {
  strcpy(fileerr, projname);
  strcat(fileerr,".err");
  ferr = fopen(fileerr,"w");
  fprintf(ferr, "*****  ERROR: Not enough memory. *****\n");
  fprintf(ferr, "(Try to calculate with less elements)\n");
  fclose(ferr);

  exit(1);
}
for (i=0;  i<6; i++) jumpline (fp);
```

Space is reserved for storing the coordinates of the nodes (pointers x, y), the connectivities (pointer N), and the materials corresponding to each element (pointer imat).
In case of error (insufficient memory), a file is created with the extension .err. This file contains information about the error and the program is aborted.
The next six lines are jumped over.

```
/* reading connectivities */
for (ielem=1; ielem<=Nelem; ielem++){
  fscanf (fp, "%d", &aux);
  for(j=0;j<3;j++) fscanf (fp, "%d", &N[(ielem-1)*3+j]);
  fscanf (fp, "%d", &imat[ielem]);
  if (imat[ielem]==0){
    strcpy(fileerr, projname);
    strcat(fileerr,".err");
    ferr = fopen(fileerr,"w");
    fprintf(ferr, "**ERROR: Elements with no material!!**\n");
    fclose(ferr);
    exit(1);
  }
}
```

The connectivities are read and the N variable is saved. This variable is a Nelem x 3- size table with two fields. The nodes (assumed triangles of 3 nodes) forming the element are saved in the first field. The element identifiers are saved in the second one.
All the elements are checked, ensuring that they have been assigned a material. If the identifier of the material is 0 (meaning that no material has been assigned to the element), an .err file is created containing information about the error and the program is aborted.

```
for (i=0;  i<5; i++) jumpline (fp);
fscanf(fp, "%s %s  %d",sau1, sau2, &Nmat );
for (i=0;  i<3; i++) jumpline (fp);
/* reading density of each material */
for (i=1; i<=Nmat; i++)
fscanf (fp, "%d %lf", &aux, &rho[i]);
/* reading conditions*/
for (i=0;  i<4; i++) jumpline (fp);
fscanf(fp, "%d", &Ncnd);
for (i=0;  i<6; i++) jumpline (fp);
for (icnd=1; icnd<=Ncnd; icnd++) {
  fscanf (fp, "%d %lf", &nodc[icnd], &wval[icnd]);
  jumpline (fp);
}
fclose (fp);
```

Reading the remaining information in the .dat file.
The total number of materials is read and stored in the Nmat variable.
The density of each material are read and stored in the rho table. The material identifier indexes the densities.
The total number of conditions is read and stored in the Ncnd variable.
The nodes associated with a condition are read and stored in the nodc table indexed by the condition identifier.
The value of the condition is stored in wval, another table indexed by the condition identifier.

```
void calculate ()
{
double v,aux1,aux2,aux3;
int n1, n2, n3;
int mat;
double x_CGi, y_CGi;
double x_num=0, y_num=0, den=0;
for(ielem=1;ielem<=Nelem;ielem++) {
  n1=  N[0+(ielem-1)*3];
  n2=  N[1+(ielem-1)*3];
  n3= N[2+(ielem-1)*3];
  /* Calculating the volume (volume is the area for surfaces) */
  v=fabs(x[n1]*y[n2]+x[n2]*y[n3]+x[n3]*y[n1]-x[n1]*y[n3]-x[n2]*y[n1]-x
[n3]*y[n2])/2;
  x_CGi= (x[n1]+x[n2]+x[n3])/3;
  y_CGi= (y[n1]+y[n2]+y[n3])/3;
  mat= imat[ielem];
  x_num+=  rho[mat]*v*x_CGi;
  y_num+=  rho[mat]*v*y_CGi;
  den+= rho[mat]*v;
}
/* puntual weights */
for(icnd=1;icnd<=Ncnd;icnd++) {
  inod= nodc[icnd];
  x_num+=  wval[icnd]*x[inod];
  y_num+=  wval[icnd]*y[inod];
  den+= wval[icnd];
}
x_CG= (x_num/den);
y_CG= (y_num/den);
```

This is the function that calculates the center of mass.

The identifiers of the nodes of the present element are saved in n1, n2, n3.
This loop makes a rundown of all the elements in the mesh. The volume is calculated for each element. (Here, the volume is the area, provided we are dealing with 3D surfaces). The volume calculations are stored in the v variable.
The geometric center of the element is calculated (coinciding with the center of gravity) and the coordinates are stored in the x_Cgi and y_Cgi variables.

The numerator sums are calculated. When the loop is finished, the following sums are stored in the x_num and y_num variables. Finally, the result of dividing the x_num and y_num variables by the den variable is stored in the x_CG and y_CG variables.

```
void output() {
  char filename[1024];
  FILE *fp, *fplog;
  double v;
```

The output() function creates two files: .post.res, and .log.
The results to be visualized in GiD Post-process are stored in the .post.res file. It is this file that stores the data which enables GiD to represent the distance of each point from the corresponding center of mass.
The numerical value of the center of mass is saved in the .log file. The accuracy of this value is directly proportional to the element size.

```
/* writing log information file */
strcpy(filename, projname);
strcat(filename,".log");
fplog=fopen(filename,"w");
fprintf(fplog, "CMAS2D routine to calculate the mass center\n");
fprintf(fplog, "project: %s\n", projname);
fprintf(fplog, "mass center: %lf %lf\n", x_CG, y_CG);
fclose(fplog);
```

Creating the .log file: the .log extension is added to the project name and a file is created that will contain the numerical value of the position of the center of mass, which in turn is stored in the x_CG and y y_CG variables of the program.

Creating the .post.res file. The output data (results) are stored in this file.
The format of the .post.res file is explained in the GiD help, see section
Postprocess data files ->Postprocess results format.

```
/* writing .post.res */
strcpy(filename,projname);
strcat(filename,".post.res");
fp=fopen(filename,"w");
fprintf(fp,"GiD Post Results File 1.0\n");
fprintf(fp,"Result MC-DISTANCE \"LOAD ANALYSIS\" 1 Scalar OnNodes\n");
fprintf(fp,"ComponentNames MC-DISTANCE\n");
fprintf(fp,"Values\n");
for(inod=1;inod<=Nnod;inod++) {
  /* distance or each node to the center of masses */
  v=sqrt((x_CG-x[inod])(x_CG-x[inod])+(y_CG-y[inod])(y_CG-y[inod]));
  fprintf(fp,"%d %lf\n",inod,v);
}
fprintf(fp,"End values\n");
fclose(fp);
```

In this example only a scalar result , with a single time step, is written in the .res file.

This is the full source code of this program:

```
#include <stdio.h>
#include <stdlib.h>
#include <malloc.h>
#include <math.h>

#define MAXMAT 1000
#define MAXCND 1000
char projname[1024];
int i,ielem,inod,icnd;
double* x,* y;
int* N,* imat;
int  nodc[MAXCND];
double rho[MAXMAT],wval[MAXCND];
int Nelem,Nnod,Nmat,Ncnd;
double x_CG,y_CG;

void input(void);
void calculate(void);
void output(void);

void main(int argc,char* argv[]) {
  strcpy(projname,argv[1]);
  input();
  calculate();
  output();
}

void input() {
  char filename[1024],fileerr[1024],sau1[1024],sau2[1024];
  FILE* fp,* ferr;
  int aux,j,error=0;
  void jumpline(FILE*);
  strcpy(filename,projname);
  strcat(filename,".dat");
  fp=fopen(filename,"r");
  for(i=0; i<6; i++) jumpline(fp);
  fscanf(fp,"%d %d",&Nelem,&Nnod);
  x=(double*)malloc((Nnod+1)*sizeof(double)); if(x==NULL) { error=1; }
  y=(double*)malloc((Nnod+1)*sizeof(double)); if(y==NULL) { error=1; }
  N=(int*)malloc((Nelem+1)*3*sizeof(int)); if(N==NULL) { error=1; }
  imat=(int*)malloc((Nelem+1)*sizeof(int));   if(N==NULL) { error=1; }
  if(error) {
    strcpy(fileerr,projname);
    strcat(fileerr,".err");
    ferr=fopen(fileerr,"w");
    fprintf(ferr,"*****  ERROR: Not enough memory. *****\n");
    fprintf(ferr,"(Try to calculate with less elements)\n");
    fclose(ferr);
    exit(1);
  }
  for(i=0; i<6; i++) jumpline(fp);
  /* reading the coordinates */
  for(inod=1; inod<=Nnod; inod++)
    fscanf(fp,"%d %lf %lf",&aux,&x[inod],&y[inod]);
  for(i=0; i<6; i++) jumpline(fp);
  /* reading connectivities */
  for(ielem=1; ielem<=Nelem; ielem++){
    fscanf(fp,"%d",&aux);
    for(j=0;j<3;j++) fscanf(fp,"%d",&N[(ielem-1)*3+j]);
    fscanf(fp,"%d",&imat[ielem]);
    if(imat[ielem]==0){
      strcpy(fileerr,projname);
      strcat(fileerr,".err");
      ferr=fopen(fileerr,"w");
      fprintf(ferr,"**ERROR: Elements with no material!!**\n");
      fclose(ferr);
      exit(1);
    }
  }
  for(i=0; i<5; i++) jumpline(fp);
  fscanf(fp,"%s %s  %d",sau1,sau2,&Nmat);
  for(i=0; i<3; i++) jumpline(fp);
  /* reading density of each material */
  for(i=1; i<=Nmat; i++)
    fscanf(fp,"%d %lf",&aux,&rho[i]);
  /* reading conditions*/
  for(i=0; i<4; i++) jumpline(fp);
  fscanf(fp,"%d",&Ncnd);
  for(i=0; i<6; i++) jumpline(fp);
  for(icnd=1; icnd<=Ncnd; icnd++) {
    fscanf(fp,"%d %lf",&nodc[icnd],&wval[icnd]);
    jumpline(fp);
  }
  fclose(fp);
}

void calculate() {
  double v;
  int n1,n2,n3;
  int mat;
  double x_CGi,y_CGi;
  double x_num=0,y_num=0,den=0;
  for(ielem=1;ielem<=Nelem;ielem++) {
    n1=N[0+(ielem-1)*3];
    n2=N[1+(ielem-1)*3];
    n3=N[2+(ielem-1)*3];
    /* Calculating the volume (volume is the area for surfaces) */
    v=fabs(x[n1]*y[n2]+x[n2]*y[n3]+x[n3]*y[n1]-x[n1]*y[n3]-x[n2]*y[n1]-x
[n3]*y[n2])/2;
    x_CGi=(x[n1]+x[n2]+x[n3])/3;
    y_CGi=(y[n1]+y[n2]+y[n3])/3;
    mat=imat[ielem];
    x_num+=rho[mat]*v*x_CGi;
    y_num+=rho[mat]*v*y_CGi;
    den+=rho[mat]*v;
  }
  /* puntual weights */
  for(icnd=1;icnd<=Ncnd;icnd++) {
    inod=nodc[icnd];
    x_num+=wval[icnd]*x[inod];
    y_num+=wval[icnd]*y[inod];
    den+=wval[icnd];
  }
  x_CG=(x_num/den);
  y_CG=(y_num/den);
}

void output() {
  char filename[1024];
  FILE* fp,* fplog;
  double v;
  /* writing log information file */
  strcpy(filename,projname);
  strcat(filename,".log");
  fplog=fopen(filename,"w");
  fprintf(fplog,"CMAS2D routine to calculate the mass center\n");
  fprintf(fplog,"project: %s\n",projname);
  fprintf(fplog,"mass center: %lf %lf\n",x_CG,y_CG);
  fclose(fplog);
  /* writing .post.res */
  strcpy(filename,projname);
  strcat(filename,".post.res");
  fp=fopen(filename,"w");
  fprintf(fp,"GiD Post Results File 1.0\n");
  fprintf(fp,"Result MC-DISTANCE \"LOAD ANALYSIS\" 1 Scalar OnNodes\n");
  fprintf(fp,"ComponentNames MC-DISTANCE\n");
  fprintf(fp,"Values\n");
  for(inod=1;inod<=Nnod;inod++) {
    /* distance or each node to the center of masses */
    v=sqrt((x_CG-x[inod])*(x_CG-x[inod])+(y_CG-y[inod])*(y_CG-y[inod]));
    fprintf(fp,"%d %lf\n",inod,v);
  }
  fprintf(fp,"End values\n");
  fclose(fp);
  free(x);
  free(y);
  free(N);
  free(imat);
}

void jumpline(FILE* filep) {
  char buffer[1024];
  fgets(buffer,1024,filep);
}
```

Wizard extension

The current course focuses on a 'tree distribution' of the information. There is another example that implements a '**wizard distribution**', based on this tree one.
The problem-type is located in problemtypes/Examples/cmas2d_customlib_wizard.gid
You can find the wizard one on our Github site.

Appendix

In this appendix, the example on how to implement a problemtypes with the classic problem-type system is presented. This system of connecting a calculation module to GiD is still supported, but it is deprecated.

The 'classic' problem-type is located in problemtypes/Examples/cmas2d.gid
A 'classic'-like problemtype is composed by these files:
.prb: configuration of the general parameter (not associated to entities)
.mat: configuration of materials and their properties
.cnd: configuration of the conditions imposed on the calculation
.bas: (template file) the file for configuring the format of the interchange file that mediates between GiD data and the calculating module. The file for interchanging the data exported by GiD has the extension .dat. This file stores the geometric and physical data of the problem.
.bat: the file that can be executed called from GiD. This file initiates the calculating module.
The calculating module (in this example cmas2d.exe) solves the equations in the problem and saves the results in the results file. This module may be programmed in the language of your choice, 'C' in used in this example.

'Classic' problemtype implementation

A 'classic'-like problemtype is composed by these files (most are not compulsory):

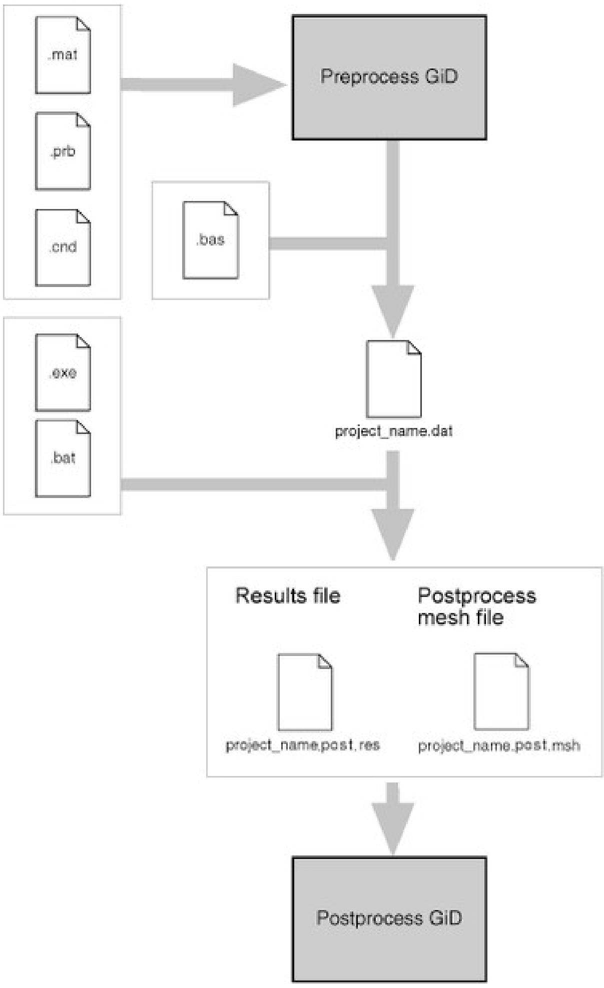
Configuration files

- problem_type_name.cnd Conditions definitions
- problem_type_name.mat Materials properties
- problem_type_name.prb Problem and intervals data
- problem_type_name.uni Units Systems
- problem_type_name.sim Conditions symbols
- ***.geo Symbols geometrical definitions
- ***.geo Symbols geometrical definitions ...

- **Template files**
- problem_type_name.bas Information for the data input file
- ***.bas Information for additional files
- ***.bas Information for additional files ...

- **Tcl extension files**
- problem_type_name.tcl Extensions to GiD written in the Tcl/Tk programming language

- **Command execution files**
- problem_type_name.bat Operating system shell that executes the analysis process



Implementation

Creating the Subdirectory for the Problem Type
Create the subdirectory "cmas2d.gid". This subdirectory has a .gid extension and will contain all the configuration files and calculating module files (.prb, .mat, .cnd, .bas, .bat, .exe).

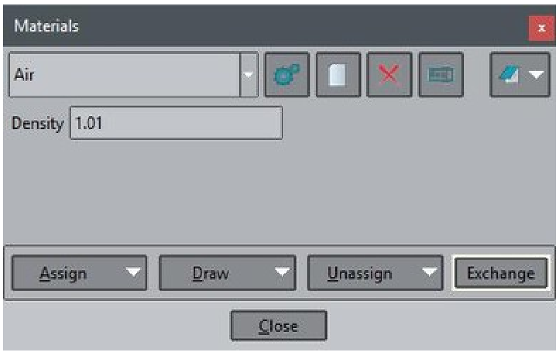
NOTE: If you want the problem type to appear in the GiD **Data->Problem type** menu, create the subdirectory within "problemtypes", located in the GiD installation folder, for example C:\GiD\Problemtypes\cmas2d.gid

Materials file

Create the materials file "cmas2d.mat". This file stores the physical properties of the material under study for the problem type. In this case, defining the density will be enough.
Enter the materials in the "cmas2d.mat" file using the following format:

MATERIAL: Name of the material (without spaces)
QUESTION: Property of the material. For this example, we are interested in the density of the material.
VALUE: Value of the property
HELP: A help text (optional field)
…
END MATERIAL

In GiD, the information in "cmas2d.mat" file is managed in the materials window, located in **Data->Materials.**



Air

Density 1.01

```
MATERIAL: Air
QUESTION: Density
VALUE: 1.01
HELP: material density
END MATERIAL
MATERIAL: Steel
QUESTION: Density
VALUE: 7850
HELP: material density
END MATERIAL
MATERIAL: Aluminium
QUESTION: Density
VALUE: 2650
HELP: material density
END MATERIAL
```

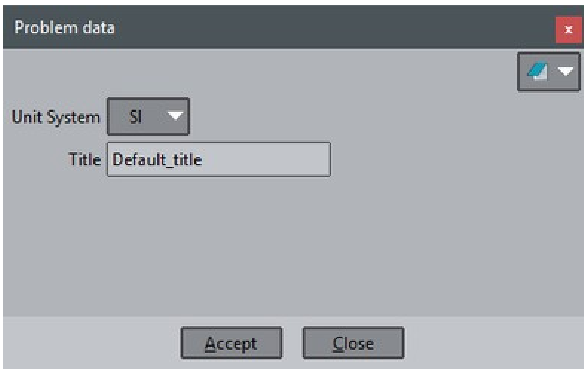
General file

Create the "cmas2d.prb" file. This file contains general information for the calculating module, such as the units system for the problem, or the type of resolution algorithm chosen.
Enter the parameters of the general conditions in "cmas2d.prb" using the following format:

PROBLEM DATA
QUESTION: Name of the parameter. If the name is followed by the #CB# instruction, the parameter is displayed as a combo box. The options in the menu must then be entered between parentheses and separated by commas.

For example, Unit_System#CB#(SI,CGS,User).
VALUE: The default value of the parameter.
…
END GENERAL DATA

In GiD, the information in the "cmas2d.prb" file is managed in the problem data window, which is located in **Data->Problem Data**.



Unit System SI

Title Default_title

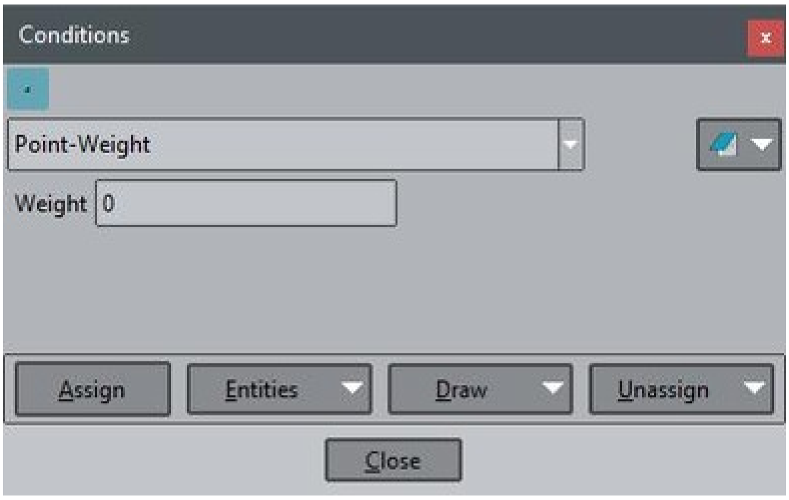
```
PROBLEM DATA
QUESTION: Unit_System#CB#(SI,CGS,User)
VALUE: SI
QUESTION: Title
VALUE: Default_title
END PROBLEM DATA
```

Conditions file

Create the "cmas2d.cnd" file, which specifies the boundary and/or load conditions of the problem type in question. In the present case, this file is where the concentrated weights on specific points of the geometry are indicated.
Enter the boundary conditions using the following format:

CONDITION: Name of the condition
CONDTYPE: Type of entity which the condition is to be applied to. This includes the parameters "over points", "over lines", "over surfaces", "over volumes", "over layers" or "over groups". In this example the condition is applied "over points".
CONDMESHTYPE: Type of entity of the mesh where the condition is to be applied. The possible parameters are "over nodes", "over body elements" or "over face elements". In this example, the condition is applied on nodes.
QUESTION: Name of the parameter of the condition
VALUE: Default value of the parameter
…
END CONDITION
…

In GiD, the information in the "cmas2d.cnd" file is managed in the conditions window, which is found in **Data->Conditions**.



```
CONDITION: Point-Weight
CONDTYPE: over points
CONDMESHTYPE: over nodes
QUESTION: Weight
VALUE: 0.0
HELP: Concentrated mass
END CONDITION
```

Writing the calculation files

Create the "cmas2d.bas" file. This file will define the format of the .dat text file created by GiD. It will store the geometric and physical data of the problem. The .dat file will be the input to the calculating module.

NOTE: It is not necessary to have all the information registered in only one .bas file. Each .bas file has a corresponding .dat file.
Write the "cmas2d.bas" file as follows:
The format of the .bas file is based on commands. Text not preceded by an asterisk is reproduced exactly the same in the .dat file created by GiD. A text preceded by an asterisk is interpreted as a command.

Example:

.bas file

```
%%%% Problem Size %%%%
Number of Elements & Nodes:
*nelem *npoin
```

.dat file

```
%%%% Problem Size %%%%
Number of Elements & Nodes:
5379 4678
```

The contents of the "cmas2d.bas" file must be the following:
.bas file

```
==================================================================
General Data File
==================================================================
Title: *GenData(Title)
%%%%%%%%%%%%%%%%%% Problem Size %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Number of Elements & Nodes:
*nelem *npoin
```

In this first part of "cmas2d.bas" file, general information on the project is obtained.
***nelem:** returns the total number of elements of the mesh.
***npoin:** returns the total number of nodes of the mesh.

```
Coordinates:
Node X Y
*loop nodes
*format "%5i%14.5e%14.5e"
*NodesNum *NodesCoord(1,real) *NodesCoord(2,real)
*end nodes
```

This command provides a rundown of all the nodes of the mesh, listing their identifiers and coordinates.

***loop, *end:** commands used to indicate the beginning and the end of the loop. The command ***loop** receives a parameter.
***loop nodes:** the loop iterates on nodes
***loop elems:** the loop iterates on elements
***loop materials:** the loop iterates on assigned materials

***format:** the command to define the printing format. This command must be followed by a numerical format expressed in C syntax.

***NodesNum:** returns the identifier of the present node
***NodesCoord:** returns the coordinates of the present node
***NodesCoord (n, real):** returns the x, y or z coordinate in terms of the value n:
n=1 returns the **x** coordinate
n=2 returns the **y** coordinate
n=3 returns the **z** coordinate

```
Connectivities:
Element Node(1) Node(2) Node(3) Material
*set elems(all)
*loop elems
*format "%10i%10i%10i%10i%10i"
*ElemsNum *ElemsConec *ElemsMat
*end elems
```

This provides a rundown of all the elements of the mesh and a list of their identifiers, the nodes that form them, and their assigned material.
***set elems(all):** the command to include all element types of the mesh when making the loop.
***ElemsNum:** returns the identifier of the present element
***ElemsConec:** returns the nodes of an element in a counterclockwise order
***ElemsMat:** returns the number of the assigned material of the present element

```
Begin Materials
Nº Materials= *nmats
```

This gives the total number of materials in the project
***nmats:** returns the total number of materials

```
Mat.          Density

*loop materials
*format "%4i%13.5e"
*set var PROP1(real)=Operation(MatProp(Density, real))
*MatNum *PROP1
*end
```

This provides a rundown of all the materials in the project and a list of the identifiers and densities for each one.
***MatProp (density, real):** returns the value of the property "density" of the material in a "real" format.
***Operation (expression):** returns the result of an arithmetic expression. This operation must be expressed in C.

***Set var PROP1(real)=Operation(MatProp(Density, real)):** assigns the value returned by MatProp (which is the value of the density of the material) to the variable PROP1 (a "real" variable).
***PROP1:** returns the value of the variable PROP1.
***MatNum:** returns the identifier of the present material.

```
Point conditions

*Set Cond Point-Weight *nodes
*set var NFIX(int)=CondNumEntities(int)
Concentrate Weights
*NFIX
```

This provides the number of entities with a particular condition.
***Set Cond Point-Weight *nodes:** this command enables you to select the condition to work with from that moment on. For the present example, select the condition "Point-Weight".
***CondNumEntities(int):** returns the number of entities with a certain condition.
***Set var NFIX(int)= CondNumEntities(int):** assigns the value returned by the command CondNumEntities to the NFIX variable (an "int" variable).
***NFIX:** returns the value of the NFIX variable.

```
Potentials Prescripts:

Node Tipus
Valor/Etiqueta
*loop nodes *OnlyInCond
*NodesNum *cond(1)
*end
```

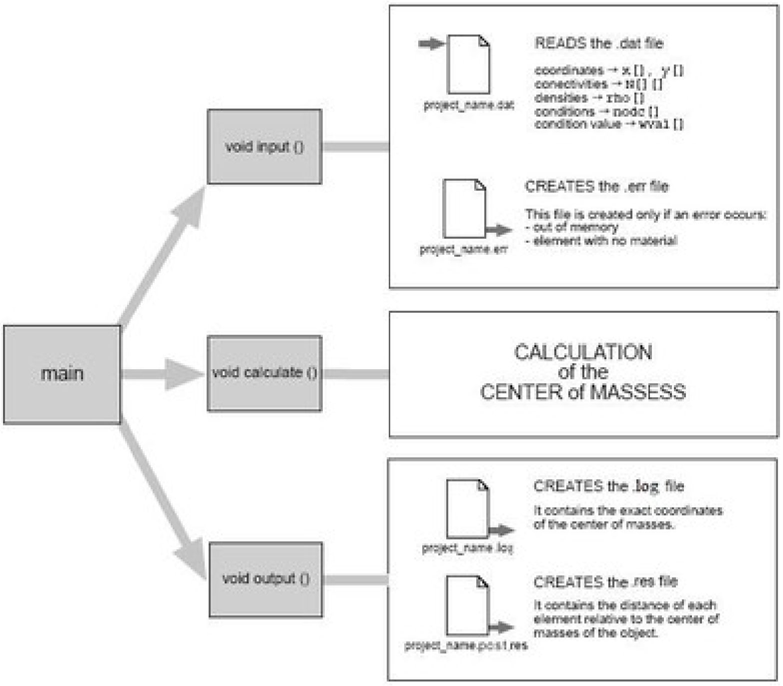
This provides a rundown of all the nodes with the condition "Point-Weight" with a list of their identifiers and the first "weight" field of the condition in each case.
***loop nodes *OnlyInCond:** executes a loop that will provide a rundown of only the nodes with this condition.
***cond(1):** returns the number 1 field of a condition previously selected with the *set cond command. The field of the condition may also be selected using the name of the condition, for example cond(weight).

cmas2d.bas

```
Node        X                Y
*set elems(all)
*loop nodes
*format "%5i%14.5e%14.5e"
*NodesNum *NodesCoord(1,real) *NodesCoord(2,real)
*end nodes

.................................................................

Connectivities:
Element    Node(1)   Node(2)   Node(3)     Material
*loop elems
*format "%10i%10i%10i%10i%10i"
*ElemsNum *ElemsConec *ElemsMat
*end elems

.................................................................

Begin Materials
Nº Materials=  *nmats
Mat.          Density
.................................................................
*loop materials
*format "%4i%13.5e"
*set var PROP1(real)=Operation(MatProp(Density, real))
*MatNum *PROP1
*end
.................................................................
Point conditions
*Set Cond Point-Weight *nodes
*set var NFIX(int)=CondNumEntities(int)
Concentrated Weights
*NFIX
.................................................................

Potentials Prescripts:
Node  Tipus
Valor/Etiqueta
*Set Cond Point-Weight *nodes
*loop nodes *OnlyInCond
*NodesNum     *cond(1)
*end

.................................................................
```

Solver

Create the file "cmas2d.c". This file contains the code for the execution program of the calculating module. This execution program reads the problem data provided by GiD, calculates the coordinates of the center of mass of the object and the distance between each element and this point. These results are saved in a text file with the extension .post.res.
Compile and link the "cmas2d.c" file in order to obtain the executable cmas2d.exe file.
The calculating module (cmas2d.exe) reads and generates the files described below.



NOTE: The "cmas2d.c" code is explained in the appendix.

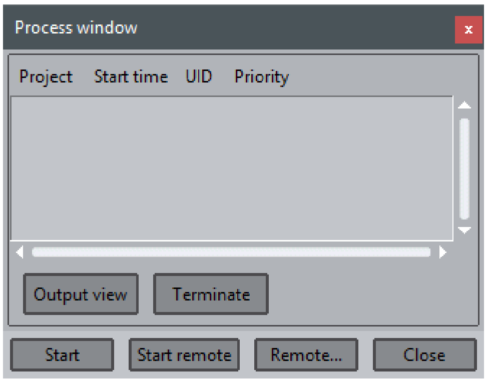
Run the solver

Create the "cmas2d.win.bat" file. This file connects the data file(s) (.dat) to the calculating module (the cmas2d.exe program). When the GiD Calculate option is selected, it executes the .bat file for the problem type selected.

When GiD executes the .bat file, it transfers three parameters in the following way:
(parameter 3) / *.bat (parameter 2) / (parameter 1)
parameter 1: project name
parameter 2: project directory
parameter 3: Problem type location directory

NOTE: The .win.bat fiile as used in Windows is explained below; the shell script for UNIX systems is also included with the documentation of this tutorial.

```
rem OutputFile: %2%1.log
```



```
rem ErrorFile: %2%1.err
```

A comment line such as "rem OutputFile: file_name.log" means that the contents of the file indicated will be shown if the user clicks Output View in **Calculate->Calculate window**.
In this example the .log file is shown. This file contains the coordinates of the center of mass.

A comment line such as "rem ErrorFile: file_name.err" means that the indicated file will contain the errors (if any). If the .err file is present at the end of the execution, a window comes up showing the error. The absence of the .err file indicates that the calculation is considered satisfactory.
GiD automatically deletes the .err files before initiating a calculation to avoid confusion.

```
del %2%1.log
del %2%1.post.res
```

This deletes results files from any previous calculations to avoid confusion.

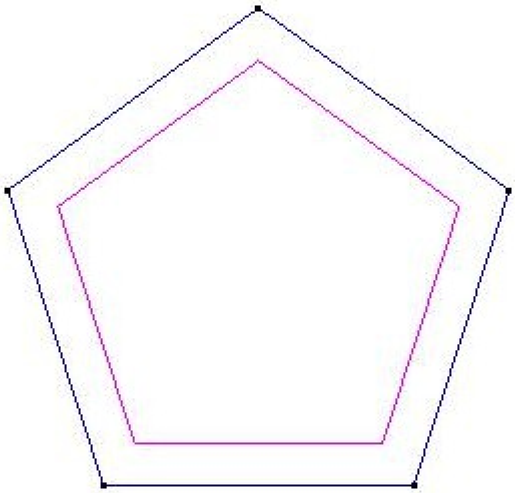
```
%3\cmas2d.exe %2%1
```

This execute the cmas2d.exe and provide the .dat as input file file.

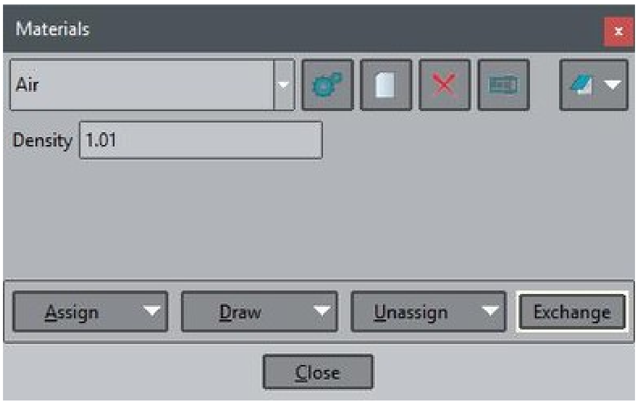
Using the problemtype classic with an example

In order to understand the way the calculating module works, simple problems with limited practical use have been chosen. Although these problems do not exemplify the full potential of the GiD program, the user may intuit their answers and, therefore, compare the predicted results with those obtained in the simulations.

- Create a surface, for example from the menu **Geometry->Create->Object->Polygon**
- Create a polygon with 5 sides, centered in the (0,0,0) and located in the XY plane (normal = 0,0,1) and whit radius=1.0



- Load the problemtype: menu **Data->Problem type->cmas2d**.
- Choose **Data->Materials**.
- The materials window is opened. From the Materials menu in this window, choose the option **Air**.

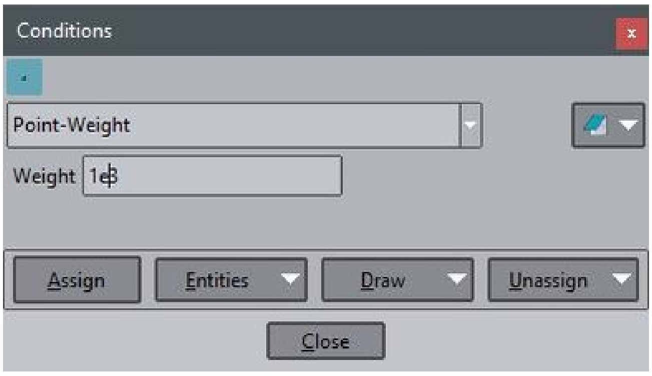


Materials

Air

Density 1.01

- Click **Assign->Surfaces** and select the surface. Press **<Esc>** when this step is finished.
- Choose the **Data->Conditions** option. A window is opened in which the conditions of the problem should be entered.

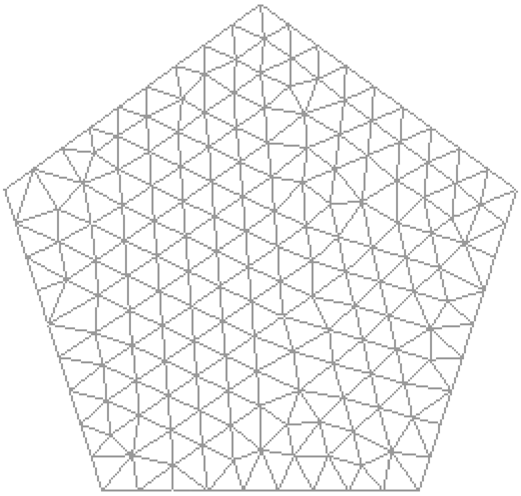


Conditions

Point-Weight

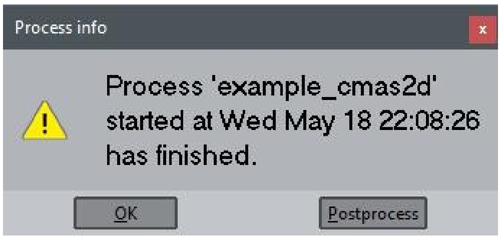
Weight 1e3

- Enter the value 1e3 in the Weight box. Click **Assign** and select the upper corner point. Press **<Esc>** when this step is finished.
- Choose the **Mesh->Generate** option.
- A window appears in which to enter the maximum element size for the mesh to be generated. Accept the default value and click **OK**. The mesh shown will be obtained.



Now the calculation may be initiated, but first the model must be saved (**Files->Save**), use 'example_cmas2d' as name for the model.

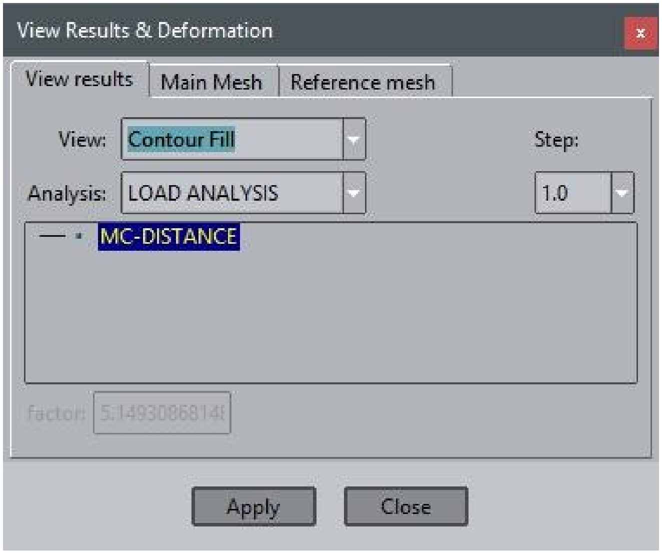
- Choose the **Calculate** option from the Calculate menu to start the calculation module.
- Wait until a box appears indicating the calculation has finished.



Process info

Process 'example_cmas2d' started at Wed May 18 22:08:26 has finished.

- Select the option **Files->Postprocess**.
- Select **Window->View results**.
- A window appears from which to visualize the results. By default when changing to postprocesses mode no results is visualized.
- From the View combo box in the View Results window, choose the **Contour Fill** option. A set of available results (only one for this case) are displayed.



View Results & Deformation

View: Contour Fill

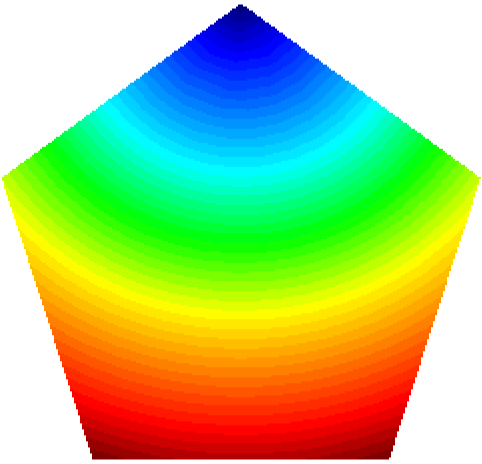
Step: 1.0

Analysis: LOAD ANALYSIS

MC-DISTANCE

Factor: 5.14930868148

- Now choose the MC-DISTANCE result and click **Apply**. A graphic representation of the calculation is obtained.



The results shown on the screen reproduce those we anticipated at the outset of the problem: the center of mass of an object is not in its geometric center because a concentrated mass on the top. The .log file will provide the exact coordinates of the calculated mass center.