# Generate a cluster from a CAD part for Windows using SphereTree:

**Necessary software:**

* *SphereTree*: <http://isg.cs.tcd.ie/spheretree/>
  + Under Downloads, get the file: “Pre-compiled Binaries (Win32)” and extract it into a folder. Standard extraction creates this folder: “spheretree-1.0-win32”
* SolidWorks or similar CAD Software in order to generate our CAD part
* *Spin 3D Mesh Converter:* <https://www.nchsoftware.com/3dconverter/index.html>. (In order to transform CAD parts into OBJ file)
  + *Alternatively use MeshLab to combine the creation of a CAD part with the direct export to .obj file 🡪* <http://www.meshlab.net/>
* *MINGW or CYGWIN:* necessary the use of a C compiler command window.
  + <https://sourceforge.net/projects/mingw-w64/>
  + <https://cygwin.com/install.html>
  + Download, extract, install and do not forget to add to path (Under environment variables).
* *Visual Studio*: In our case, VS17 is used. Make sure that the features “.NET desktop development” and “Desktop development with C++” are clicked inside the Visual studio Installer: <https://www.microsoft.com/de-de/techwiese/aktionen/visual-studio-2017.aspx>
  + This software is needed in order to obtain the *“****x64 Native Tools Command Prompt for VS 2017****”* command window, which will be used when generating the SPH file.
  + Add Visual Studio to the path (environment variables)
* *Visual Studio Code* (Optional): <https://code.visualstudio.com/download>. *In order to be able to edit the code. Very recommended. Otherwise, one can edit the code using the notepad or VS17.*

**Method:**

Create Object we want to transform into cluster:

* Create desired geometry in CAD Software. Make sure that the center of mass of the piece is located in the origin of the coordinate system (0,0,0) 🡪 IMP
  + Using SolidWorks + Spin 3D Mesh Converter: (recommended)
    - Export in SolidWorks as .STL mesh, import in Spin 3D and export again as .OBJ
  + Using directly MeshLab:
    - Export geometry directly to .OBJ file
* Save the .OBJ file to the same folder where SphereTree is located, in our case: “spheretree-1.0-win32” after extraction.

Generate SPH file:

* For more information regarding which software and the specifications of each of them, refer to the README file inside the extracted folder. Here, it is recommended not to use the GUI since the customization possibilities are reduced.
  + Using Windows GUI:
    - To use GUI, execute SphereTree.exe
    - Open the .OBJ file inside using File\Open and select the desired .OBJ file
    - To generate .SPH file go into Generate\Tree and select the desired algorithm (Octree or Hubbard)
    - File will be saved with the same name as the .OBJ file but with a SPH format
  + Using cmd algorithms (Visual Studio’s cmd is necessary):
    - Recommended algorithm to use is: “makeTreeMedial.exe”
    - Open the command window from Visual Studio mentioned before “x64 Native Tools…”. If Visual Studio has been added to the path, simply by looking for “cmd” in the Windows search bar, one should be able to open it.
    - Copy the path to the “spheretree-1.0-win32” folder, where the algorithms and the .obj file are located.
    - In the command window, write “**cd**” and paste the path to our folder. The folder will be opened inside the terminal. (*See image below*).
    - Then, run the following command:
      * *makeTreeMedial -branch* ***NS*** *-depth 1 -testerLevels 2 -numCover 10000 -minCover 5 -initSpheres 1000 -minSpheres 200 -erFact 2 -verify -nopause -eval -expand -merge -burst -optimise balance -balExcess 0.001 -maxOptLevel 100* ***file\_name.obj***

Where **NS** is the max number of spheres we want in the cluster- We also substitute “**file\_name**” for the name of the file we want to transform, the OBJ file. In order to know what each subcommand means, please refer to the README file.

It should look similar to this:

After running the command, a “**file\_name**-medial.SPH” file is created inside the folder. The format of the file looks like this:

2 10

0.004280 -0.013003 0.008341 0.783414 1.000000

**-0.568996 0.018189 -0.051217 0.244317** 1.000000

**-0.545343 0.021028 0.130538 0.223218** 1.000000

**-0.350713 0.022849 -0.223548 0.276149** 1.000000

**-0.231132 0.007127 0.186624 0.293184** 1.000000

**-0.076302 0.013149 -0.184722 0.294902** 1.000000

**0.055253 0.007866 0.184866 0.300992** 1.000000

**0.212799 0.020166 -0.153577 0.232894** 1.000000

**0.415381 0.003981 0.101654 0.263739** 1.000000

**0.595659 0.013944 -0.063549 0.190173** 1.000000

**0.459636 0.017563 -0.196114 0.145843** 1.000000

0.000000 0.000000 0.000000 -0.008474 1.000000

Information of the cluster

…

**Recommended to use VS Code to edit the file. Notepad or VS also works.**  
  
The first line describes the number of clusters and the number of spheres that the cluster with the most spheres, has. 🡪 Delete line

The second line contains the center and radius of a cluster of one sphere (the 1.000000 is not important). 🡪 Line not important. Delete

The next lines are the center and radius of the spheres representing the cluster, followed by some information about it which is of no interest to us.

Inside the *SPH* file we need to erase all of this useless information: We edit the file and we leave only the previously marked in bold. (This means, we delete the first two lines and all the 1.000000. Also make sure that all radii are positive. Elements like: 0.000000 0.000000 0.000000 -0.008474 1.000000, should also be deleted. Lastly, the additional cluster information after the spheres should also be deleted).

After editing, the file should look like this:

**-0.568996 0.018189 -0.051217 0.244317**

**-0.545343 0.021028 0.130538 0.223218**

**-0.350713 0.022849 -0.223548 0.276149**

**-0.231132 0.007127 0.186624 0.293184**

**-0.076302 0.013149 -0.184722 0.294902**

**0.055253 0.007866 0.184866 0.300992**

**0.212799 0.020166 -0.153577 0.232894**

**0.415381 0.003981 0.101654 0.263739**

**0.595659 0.013944 -0.063549 0.190173**

**0.459636 0.017563 -0.196114 0.145843**

Generate MSH file: (Here we are just generating An MSH file with Tetrahedrons in GiD).

* Once the SPH file has been created, the next step is to create an MSH file. For this, we open GiD.
* We import the CAD part either as an STL mesh (recommended) or as an OBJ wave-front.
* Create a geometry from a mesh by going to: Geometry\Create\Geometry from mesh\One-element one-surface. And select the mesh to convert (our object).
* Repair and collapse model (to avoid inconsistencies in our mesh).
* Create a volume of our model by surface contour.
* Select Mesh\Element type\Tetrahedra.
* Generate mesh.
* Save file in the same folder as the SPH file. (“spheretree-1.0-win32”)

Generate Cluster file:

* Once we have the SPH file and MSH file derived from the original OBJ file, we proceed to generate the cluster file (.clu)
* For this, we look for the file: ***mesh\_to\_clu\_converter.cpp*** from the Kratos folder. We copy this file and paste it in the same folder where the previous files are located.
* We muss edit the file (using VS Code or VS or Notepad) in the following manner:
  + Change the input files names in the lines 13 and 14:

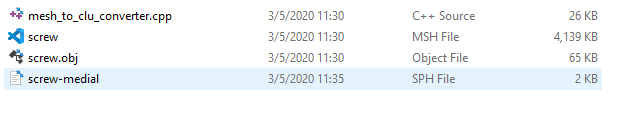
*std::ifstream infile("****file\_name.msh****");*

*std::ifstream infilesph("****file\_name.sph****");*

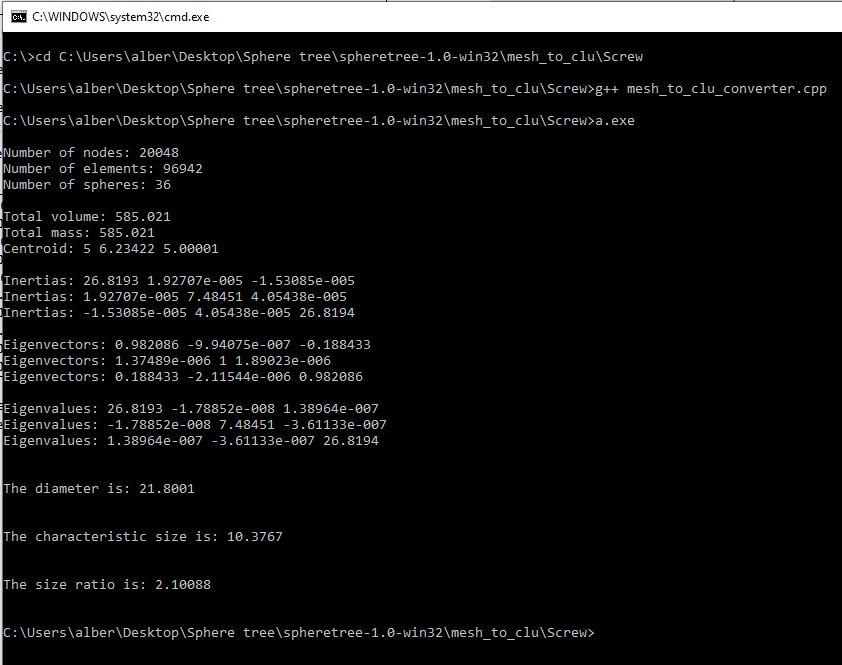
And the output file name in line 388:

*std::ofstream outputfile("****file\_name.clu****", std::ios\_base::out);*

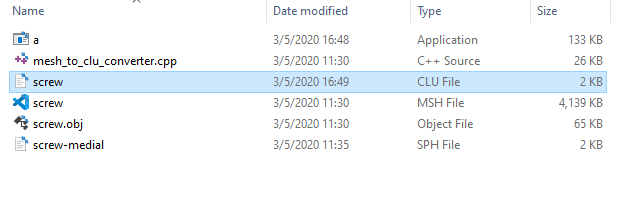
* + The input names must coincide with our input file names (namely SPH and MSH), the output name is up to us to decide.
  + We can also change the lines 389,390 and 391 in order to edit the author and time of cluster generation we require.
* We save the file.
* Our folder should now look something like this:



* The names of the files do not matter, as long as the same exact names are written inside the “mesh\_to\_clu\_converter.cpp” file (as mentioned before).
* Final steps consist in opening the file with either MINGW or CYGWIN. Inside this cmd C compiler, we input the following commands:
  + cd “path to where mesh\_to\_clu\_converter.cpp is located”
  + g++ mesh\_to\_clu\_converter.cpp (a file output named “a” is generated in the same folder) 🡪 here we are compiling.
  + Now we run the “a” file by writing in the cmd: a.exe 🡪Here we execute.
  + After some time, a file name “***file\_name.clu”*** is also created inside our folder.
    - This is the final (.clu) file we can include into Kratos when performing a simulation. This file may be altered to correct miscalculations in the COM for example and/or can be moved to another folder of our preferencer. An example of how the cmd in MINGW should look like is:



The folder after the process looks like this:



“**screw.clu has been generated from screw.obj**”