Memorandum

Xavier Mosquera Campos

December 2021

License

Introduction

This new tool called *vtkConverter* has been developed in order to standardize a methodology across F4E/A&C and suppliers for analyzing, modifying and converting VTK responses to suitable formats for subsequent analyses (for example with Alya, ANSYS APDL/Mechanical, ANSYS Fluent or Abaqus). The tool is a python 3.9 based scripts able to read any VTK file produced by *Mesh2Vtk*, a tool depeloped by UNED and F4E, which converts the meshes produced by MCNP and D1S-UNED into a VTK format.

The tool is used through a text based interactive menu, and it can be run under Windows or Linux systems.

What the tool does

Capabilities of vtkConverter are divided into six categories:

- Mesh information

  The tool can display either information of mesh or information about an array of data corresponding to points or cells.

- Mesh conversion to other formats

  The tool can write the information of a mesh selected by the user. Conversion formats are: Point Cloud, IP Fluent and CSV. See them at the Annex I.

- Operation on meshes

  The tool allows performing various operations with meshes. Operations are:

  - Translate: Translate the mesh in x,y,z axes.
  - Rotate: Rotate the mesh around x,y,z axes.
  - Joint: Joint two different meshes.

- Mesh export to VTK

  The tool can export a mesh to VTK again, either in *binary* or *ASCII* format. It's useful after applying some operation on a mesh.

- Scale factor

  The tool has integrated a scale factor, to multiply the coordinate points of the mesh by a factor when it is converted to other formats.

- Safety factor

  The tool has integrated a safety factor, to multiply the values of any array data by a factor when it is converted to other formats.

Installation

The vtkConverter is written in python 3.9. It is compound of two modules, the principal module "vtkConverter.py" and a module dependency "vtkConv_functions.py". In addition, the script uses two other package dependencies which are "numpy" and "pyvista". The last one must be the 0.32.1 version or higher.

The VtkConverter_program folder contains a "setup.py" file, a README and another folder called VtkConverter with "vtkConverter.py" and "vtkConv_functions.py".

In order to install all the packages needed directly, execute this command line within the VtkConverter_program folder:

```
> pip install .
```

How to use it

The vtkConverter script operates through an interactive text menu interface. To be launched the user has to type, in the principal folder directory, the following command:

```
> vtkconverter
```

Once the program is launched, the principal menu is displayed and it has seven options (plus "Exit" option), see figure 1. The keywords to be used to perform actions are the words between parentheses displayed in the menu.

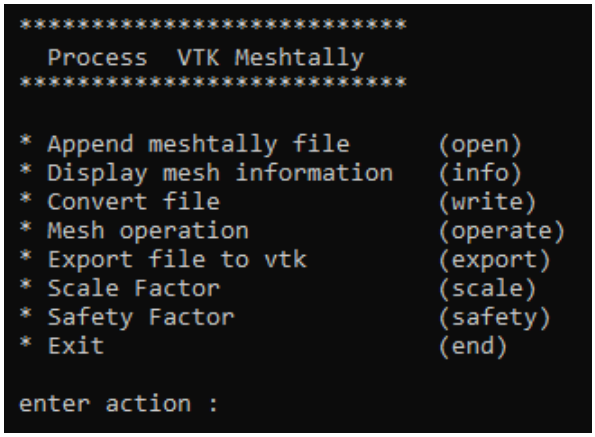

Figure 1: Principal menu

- **Append meshtally file:** This option is used to open a VTK file to be processed. Once you write the keyword *open*, the program displays a new window to select the input file. See figure 2.

  Several files can be opened at the same time. For this, the action *open* should be repeated as many times as there are files to be added.

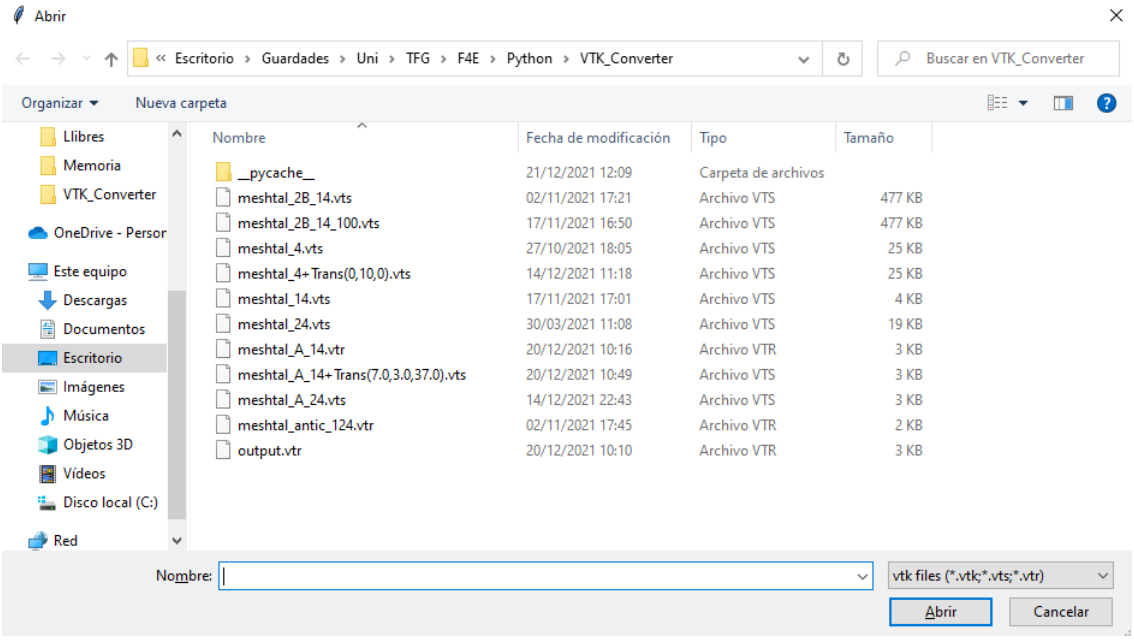

Figure 2: Window to select the file

- **Display mesh information:** This option is used to either display mesh or array information. See figure 3.

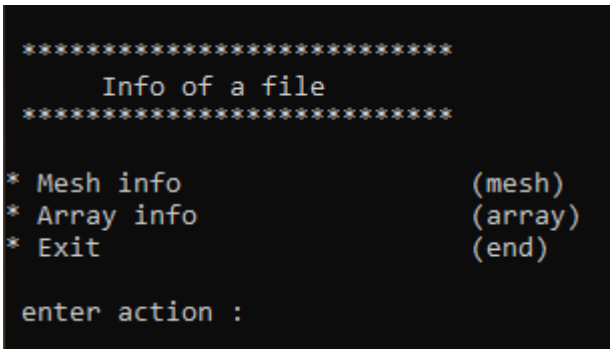

Figure 3: Information menu

  - **Mesh info:** Displays general information of the mesh of the file selected (figure 4).
  - **Array info:** Displays information of the array selected corresponding to the mesh of the file selected (figure 5).

    Integral and Average without volume calculates, respectively, the sum and the average of all values in the array selected. With volume values are calculated according to every cell volume. Each value is multiplied for its corresponding volume cell.

```
Name: meshtal_A_24.vts
Author: ['xavie']
Path: ['C:\\Users\\xavie\\Desktop\\Guardades\\Uni\\TFG\\F4E\\Python
 \\Mesh2vtk']


Number of cells: 24
Number of points: 60
Cells arrays: ['Value - Total', 'Error - Total']
Points arrays: []
Number of coordinates: 3
Mesh bounds: [2.0, 8.0, -8.0, 8.0, -8.0, 8.0]
Mesh dimensions: (6.0, 16.0, 16.0)
Mesh type: StructuredGrid
```

Figure 4: Mesh info

```
Minimum value: 2.54e-04
Maximum value: 3.88e-03

Integral without volume: 2.94e-02
Integral with volume: 6.69e-01

Average without volume: 1.23e-03
Average with volume: 8.71e-04
```

Figure 5: Array info

- **Convert file:** This option is used to convert the VTK file to other suitable format. It uses the safety and scale factor during the conversion. The formats available are shown in figure 6 and figures 12, 13 and 14 are examples of each one, explained in the Annex I. Once the format is selected, the program displays the different files that can be written and, after this, it displays the possible arrays of data that can be chosen during the conversion.

  For the *point_cloud* and the *ip_fluent* formats, the tool writes a different file for each array of data selected. For *csv* format, it writes all data in the same file. If the user wants to write several arrays at once, they must be written with a "," between they (and no spaces).

  Example, for *"Value - Total" and "Error - Total"* arrays:

  > Value - Total,Error - Total

  The name of the files is *"VTKfilename_arrayname_format.txt"* for Point Cloud and IP Fluent formats, and *"VTKfilename_[arrayname1,arrayname2,...].csv"* for CSV format.

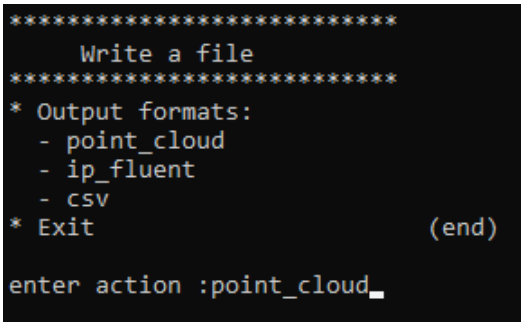

Figure 6: Write menu

- **Mesh operation:** This option allows the user to operate with a mesh. In figure 7 is shown the operate menu.

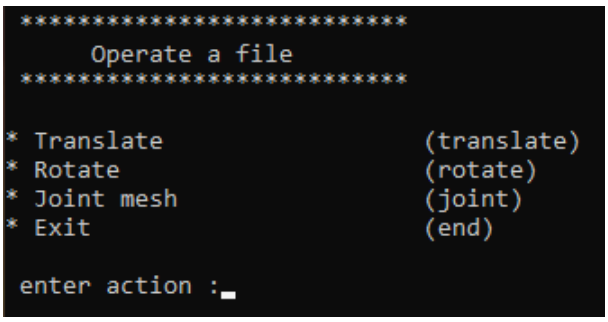

Figure 7: Operate menu

This menu has three options (plus the "Exit"):

  - **Translate:** This option allows the user to translate a mesh. The user must select a file and the next input required indicates the translation in x,y,z axes. The three numbers are required and the units are the same as in the original file (the scale factor is not used yet). They have to be written with "," separation (and no spaces). For example:

    > 100,-50,0

    When the operation is finished, the program saves the result in a file with name *"VTKfilename_+Trans(x,y,z).extension"*. The extension will be *vts* for *Structured Grids* and *vtu* for *Unstructured Grids* meshes.

    As the **Translate** operation cannot be applied to a *RectilinearGrid*, the program itself converts this type of mesh into a *StructuredGrid* mesh.

  - **Rotate:** This option allows the user to rotate a mesh. The user must select a file and the next input required indicates the rotation around x,y,z axes. The three numbers are required and the units of the angle rotated is in *degrees* [º]. They have to be written with "," separation (and no spaces). For example:

    > 180,-90,0

    When the operation is finished, the program saves the result in a file with name

"VTKfilename_+Rot(x,y,z).extension". The extension will be *vts* for *Structured Grids* and *vtu* for *Unstructured Grids* meshes.

As the **Rotate** operation cannot be applied to a *RectilinearGrid*, the program itself converts this type of mesh into a *StructuredGrid* mesh.

– **Joint:** This option allows the user to joint two different meshes. After selecting a first file, the next input is the file to perform the joint with.

When the operation is finished, the program saves the result in a file with name *"VTKfilename1_+VTKfilename2_.vtu"*. The extension is *vtu* because the result mesh is an *Unstructured Grid*, regardless the initial meshes.

- **Export file to vtk:** This option is used to export a mesh to VTK again. It is useful to save a mesh in this format after performing some operations on it, like *translate* or *rotate*. First of all, as shown in figure 8, the format must be selected. It determines how the file information will be written, either in *binary* or *ASCII*. *Binary* files write much faster than *ASCII* and have smaller file size.

  Once the format is selected, the user has to choose the file that will be exported. The name of the new file is "VTKfilename.extension". The extension will be *vts* for *Structured Grids*, *vtr* for *Rectilinear Grids* and *vtu* for *Unstructured Grids* meshes.

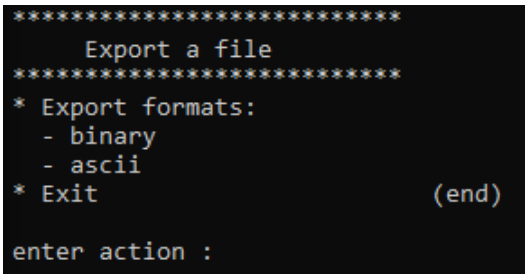

Figure 8: Export menu

- *Scale Factor:* This option is used to change the value of the *Scale Factor*, see figure 9. Its default value is 1 and it multiplies the values of the coordinate points of the mesh.

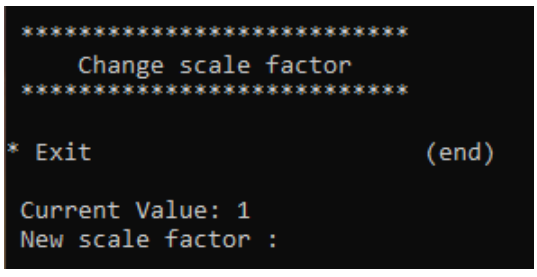

Figure 9: Scale menu

- *Safety Factor:* This option is used to change the value of the *Safety Factor*, see figure 10. Its default value is 1 and it multiplies the values of the fields arrays of the mesh in the *write* conversion.

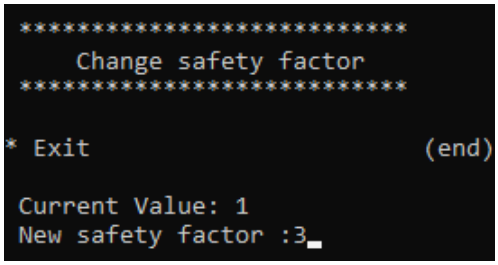

Figure 10: Safety menu

- **Some other details:** The program is made to facilitate the user performance. Thus, there are some details that have to be commented if the user want to use the program.
  - Every time the user needs to select a file, the program displays a list of the files available, as it is shown in figure 11.

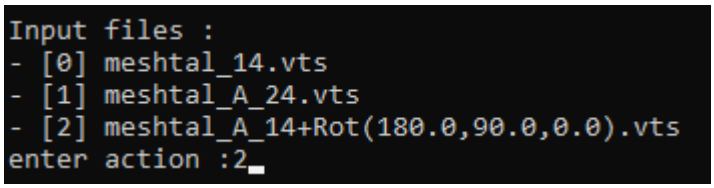

Figure 11: Display of input files

To select a file, the user must just write the index of the file desired. For example, in this case, if the user wants to select *"meshtal_A_+Rot(180.0,0.0,0.0).vts"*, the input action must be *"2"*.

  - If there is only one file opened, every action is performed without the need of selecting a file. It will be applied to the only file available.

Recommendations and warnings

The program gives no indication of the units that are being used. The user should be coherent with the units of every mesh and the operations it performs to the mesh. The same is applied to the *scale* and *safety* factors.

Testing the program first is recommended if different packages version are used.

Also, especially if different packages versions are normally used, the creation of a dedicated python environment is recommended.

Possible developments

Implementing automatic pytest and coverage to execute, analyse and ensure to program still works after applying changes to the actual version.

Also, implementing an automatic generation of a dedicated python enviroment to use the program.

Another possible development would be developing a GUI interface, possibly depeloped with the library *PyQt5*, based on the interactive menu of this program, in order to facilitate the use of the tool.

Annex I

```
x, y, z, value
2.665,-4.000,-4.000,0.001
4.000,-4.000,-4.000,0.001
5.335,-4.000,-4.000,0.001
2.665,4.000,-4.000,0.001
4.000,4.000,-4.000,0.001
5.335,4.000,-4.000,0.001
2.665,12.000,-4.000,0.000
4.000,12.000,-4.000,0.000
5.335,12.000,-4.000,0.000
2.665,-4.000,4.000,0.001
4.000,-4.000,4.000,0.001
5.335,-4.000,4.000,0.001
2.665,4.000,4.000,0.001
4.000,4.000,4.000,0.001
5.335,4.000,4.000,0.001
```

Figure 12: Point Cloud format

The Point Cloud format, see figure 12, consists in the coordinate points values x, y, z and the value of the array corresponding to that point for each line. The points and array values are separated by commas. The first line is always 'x,y,z,value'.

File	Comments
3	-
3	No. of Coordinates
11214	No. of Values
1	-
uds-0	Uds Field to fill
(6.36591796875	X values of the points
6.341943969726563	
…	
)	
(0.02524073839187622	Y values of the points
…	
)	
(4.8198388671875	Z values of the points
…	
4.674461364746094	
)	
(607181.0126304626	Values of the points
619962.9902839661	
..	
)	

Figure 13: IP Fluent format

The IP Fluent format, see figure 13, consists in a list of all x value of the coordinate points, followed by the y values and the z values. After the coordinate points, there is a list of all values of the array selected. Each line corresponds to one value.

The first 5 lines correspond to information about the mesh.

```
1  Xi, Yi, Zi, variable#1, variable#2, ..., variable#n
2  Xi1, Yi+1, Zi+1, variable#1, variable#2, ..., variable#n
3  ...
4  Xn, Yn, Zn, variable#1, variable#2, ..., variable#n
```

Figure 14: CSV format

The CSV format, see figure 14, consists in the coordinate points values x, y, z and the values of one or more arrays corresponding to that point for each line. The points and array values are separated by commas.