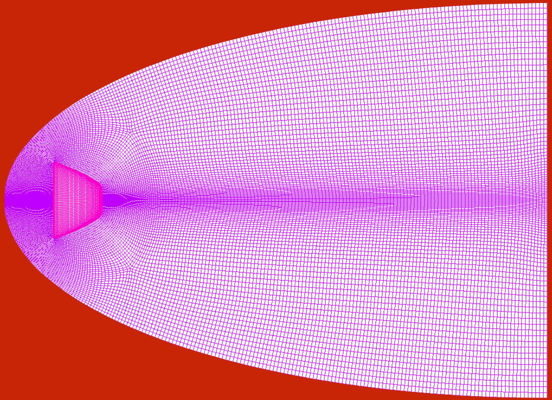
Grid*Pro* API – Variable Geometry Automation of Re-entry Capsule

API Tutorial - 5



Scope of the tutorial

- In this tutorial, we will discuss about how to generate multiple grids for different geometries with minor design variations.
- Initially a 2D profile representing the re-entry capsule is created with several design variables.
- The 2D profile was then revolved to generate the 3D model. The design variables are then tweaked to generate 10 different designs.
- A template topology is created and saved as topology.fra.
- We will be generating the grid using the template topology for all the 10 designs with the help of API.

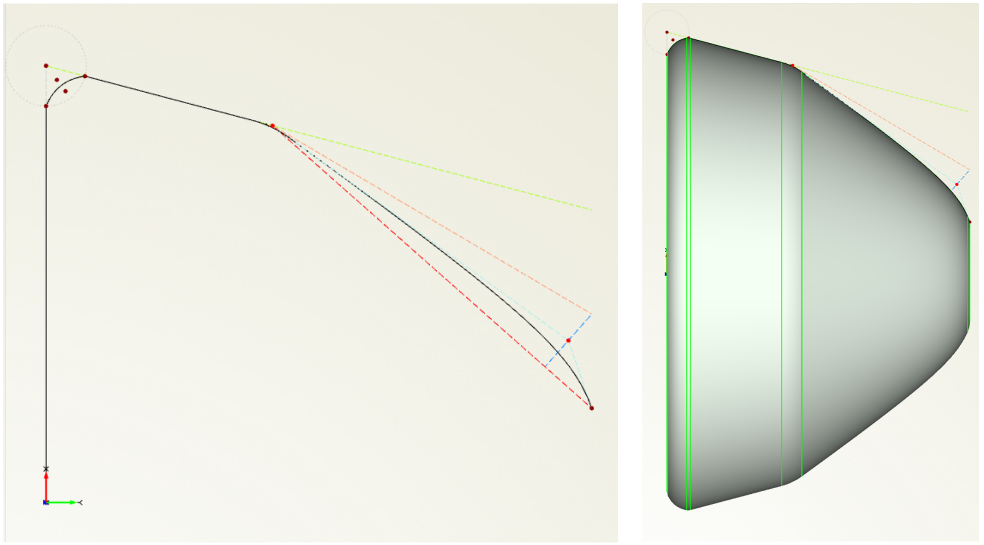
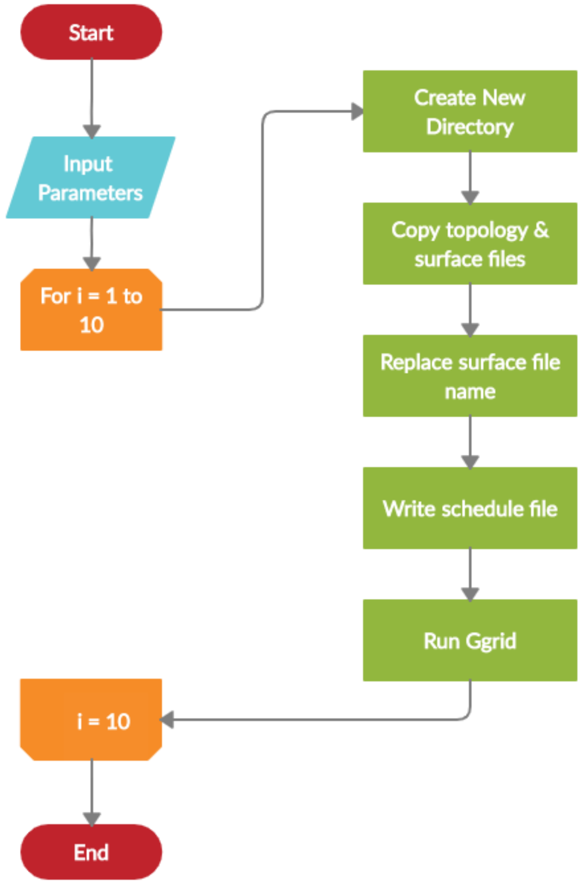


Figure 1: Reentry Capsule 2D profile and 3D model

Problem Definition

- The work flow of the script goes as follows:
  1. Get the required input parameters.
  2. Create a new directory w.r.t the geometry file name prefix.
  3. Copy the template topology and the corresponding geometry.
  4. Replace the old file name with the new file name in the fra file.
  5. Write the schedule file as required using the input parameters.
  6. Run ggrid on the topology file.
  7. Repeat step 2 to 7 for all the 10 designs.
  8. End

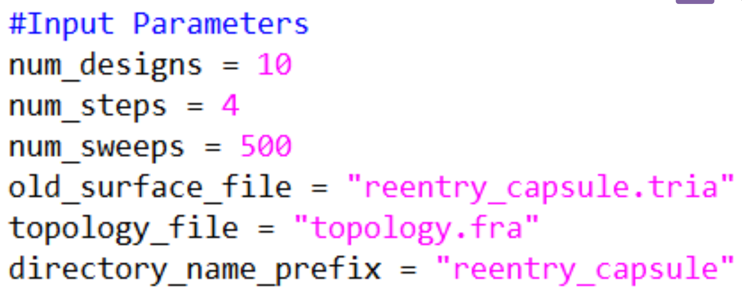


Input Parameters

- To start with we need to input the template topology file name and the number of geometries available.
- Then the old surface file name to replace it with the new file name and the directory name to create a new directory for every design.
- To write the schedule file, we need to input the number of steps and number of sweeps per step to run the grid generator.

Input

- ☐ num_designs – 10
- ☐ num_steps - 5
- ☐ num_sweeps - 500
- ☐ old_surface_file – "reentry_capsule.tria"
- ☐ topology_file – "topology.fra"
- ☐ directory_name_prefix – "reentry_capsule"



```
#Input Parameters
num_designs = 10
num_steps = 4
num_sweeps = 500
old_surface_file = "reentry_capsule.tria"
topology_file = "topology.fra"
directory_name_prefix = "reentry_capsule"
```

Figure 4: Input Parameters

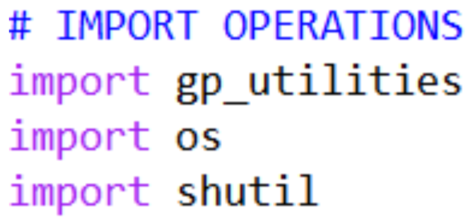
Code Snippet – Reentry_Capsule.py

```python
# IMPORT OPERATIONS
import gp_utilities
import os
import shutil


#Function to copy the given list of files to new directory
def copy_files_to_destination_directory(source, destination,
files_list):
    if os.path.isdir(destination):
       shutil.rmtree(destination)
    os.mkdir(destination)
    for j in files_list:
       shutil.copy(source + "/" + j, destination)


#Function to find an old string and replace with new string
def find_and_replace_string(file, find, replace):
    with open(file, 'r') as f:
       string = f.read()
       f.close()
    with open(file, 'w') as f:
       string = string.replace(find, replace)
       f.write(string)


# Main Function
if (__name__ == '__main__'):
    topo = gp_utilities.Topology()
```

```python
#Input Parameters
num_designs = 10
num_steps = 4
num_sweeps = 500
old_surface_file = "reentry_capsule.tria"
topology_file = "topology.fra"
directory_name_prefix = "reentry_capsule"

for i in range(1, num_designs + 1):
    new_surface_file = "{}_{}.tria".format(os.path.splitext(old_surface_file)[0],
    i)
    source_dir = os.getcwd()
    destination_dir = "{0}_{1}".format(directory_name_prefix, i)

    copy_files_to_destination_directory(source_dir, destination_dir,
    [new_surface_file, topology_file])
    os.chdir(destination)


    find_and_replace_string(topology_file, old_surface_file, new_surface_file)

    topo.write_schedule_file("{}.sch".format(os.path.splitext(topology_file)[0]),
    num_steps, num_sweeps, "{}.grd".format(os.path.splitext(new_surface_file)[0]),
    "dump.tmp")

    Ggrid = "Ggrid {}".format(topology_file)
    os.system(Ggrid)
    os.chdir(source_dir)
```

API Module

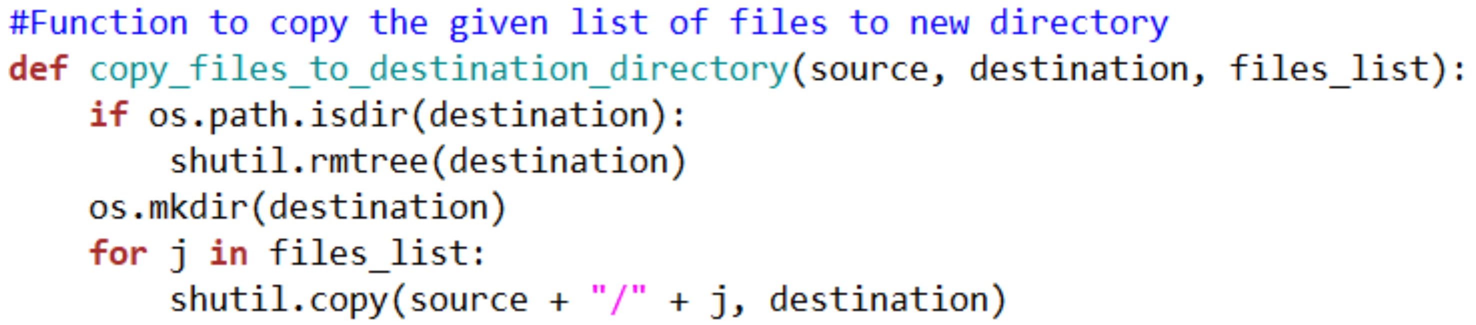
- Import the following libraries:
  1. gp_utilities = GridPro's API
  2. os = To run system command
  3. shutil = To run high level file operations



```python
# IMPORT OPERATIONS
import gp_utilities
import os
import shutil
```

API Module

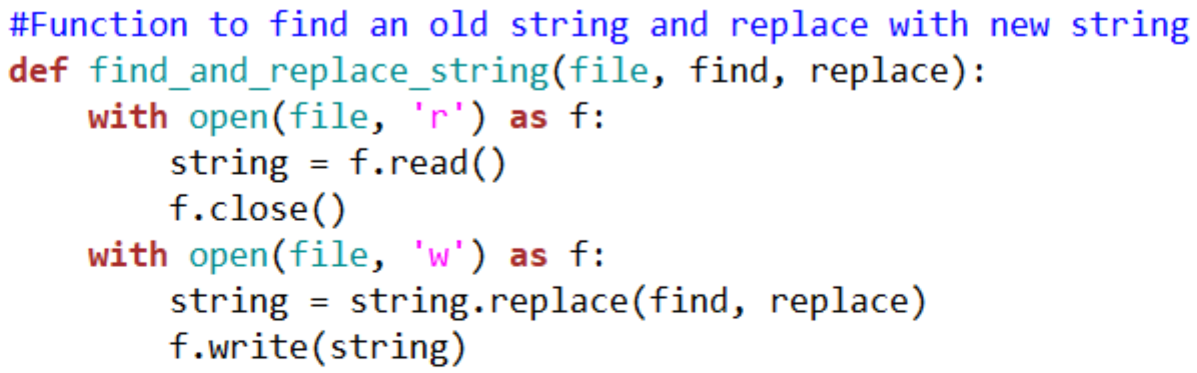
- First Function: To Copy the given files from source directory to destination directory.
  1. If the destination directory is already exists, delete the directory and its contents.
  2. Create a new destination directory.
  3. Copy the given list of files to the destination directory.



```python
#Function to copy the given list of files to new directory
def copy_files_to_destination_directory(source, destination, files_list):
    if os.path.isdir(destination):
        shutil.rmtree(destination)
    os.mkdir(destination)
    for j in files_list:
        shutil.copy(source + "/" + j, destination)
```

API Module

- Second Function: To replace the old surface file name with new surface file name in the given fra file.
    1. Read the contents of the file
    2. Find the given surface file name and replace it with the new surface file name



```python
#Function to find an old string and replace with new string
def find_and_replace_string(file, find, replace):
    with open(file, 'r') as f:
        string = f.read()
        f.close()
    with open(file, 'w') as f:
        string = string.replace(find, replace)
        f.write(string)
```

Note: Both the functions are written in a generic manner so that it can be reused in future.

API Module

- Main Function:

1. Collect the input parameters

```python
#Input Parameters
num_designs = 10
num_steps = 4
num_sweeps = 500
old_surface_file = "reentry_capsule.tria"
topology_file = "topology.fra"
directory_name_prefix = "reentry_capsule"
```

2. Execute both functions, which will copy the topology and surface files to a new directory and replace the old surface file with new surface file.
3. Write the schedule file and run the grid generator.
4. Repeat step 2 and 3 for the total number of designs.

```python
for i in range(1, num_designs + 1):
    new_surface_file = "{}_{}.tria".format(os.path.splitext(old_surface_file)[0], i)
    source_dir = os.getcwd()
    destination_dir = "{0}_{1}".format(directory_name_prefix, i)

    copy_files_to_destination_directory(source_dir, destination_dir, [new_surface_file, topology_file])
    os.chdir(destination_dir)

    find_and_replace_string(topology_file, old_surface_file, new_surface_file)

    topo.write_schedule_file("{}.sch".format(os.path.splitext(topology_file)[0]), num_steps, num_sweeps,
                             "{}.grd".format(os.path.splitext(new_surface_file)[0]), "dump.tmp")
    Ggrid = "Ggrid {}".format(topology_file)
    os.system(Ggrid)

    os.chdir(source_dir)
```

End of the Tutorial