**EasyCFD Instruction Manual**

EasyCFD is an automated open-source workflow for computational fluid dynamics simulations.

This folder contains the following:

* Python Code
  + ezFOAM.py
    - The graphics window the user interacts with to supply input necessary for building an OpenFOAM case
  + main.py
    - The master function that ezFOAM.py calls. It contains a function that calls all the functions below
  + make0.py
    - Builds the 0 folder
  + makeConstant.py
    - Builds the constant folder
  + makeSystem.py
    - Builds the system folder
  + su2ToSTL.py
    - Converts the enGrid SU2 file to STL files for each named boundary
  + groups.py
    - Function called by su2ToSTL.py that extracts the boundary names
* Sample STL Files
  + Provided geometry files to be processed through enGrid
* Necessary Software
  + enGrid is the software used for preparing STL geometry files for OpenFOAM use. An installer is included in this folder, but it can also be downloaded from <https://sourceforge.net/projects/engrid/>
  + Python 3 is the programming language of EasyCFD. The Python 3.7.2 installer is included in this folder, but other Python 3 versions can be downloaded from <https://www.python.org/downloads/>
  + blueCFD-Core is OpenFOAM for Microsoft Windows and runs the CFD simulations. The latest version can be downloaded from <http://bluecfd.github.io/Core/Downloads/>

**Installation Instructions**

1. In the Necessary Software folder, download and run the Python 3.7.2 installer. Select Install Now.
2. Download the latest version of blueCFD-Core at <http://bluecfd.github.io/Core/Downloads/>. Follow the prompts and accept the defaults.
   1. Choose the full installation
   2. On the Completing the blueCFD-Core Setup Wizard window, uncheck all boxes. The user may leave the blueCFD-Core Online User Guides box checked
3. In the Necessary Software folder, download and run the enGrid installer.
4. Download the contents of the Python Code folder. This folder can be downloaded in any location.

Your machine is now able to use EasyCFD

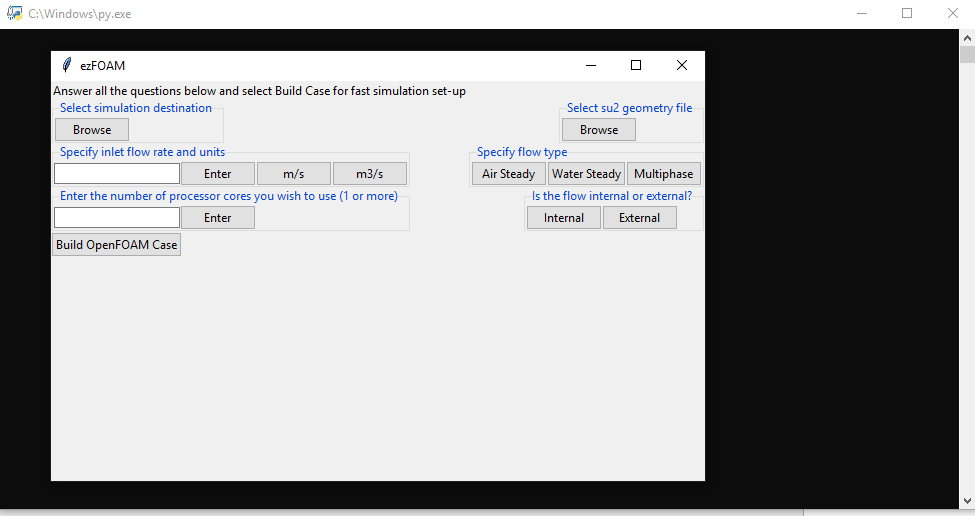
**EasyCFD Operating Instructions (needs work, more description, missed some things)**

**Stage 1: Geometry Preparation through enGrid**

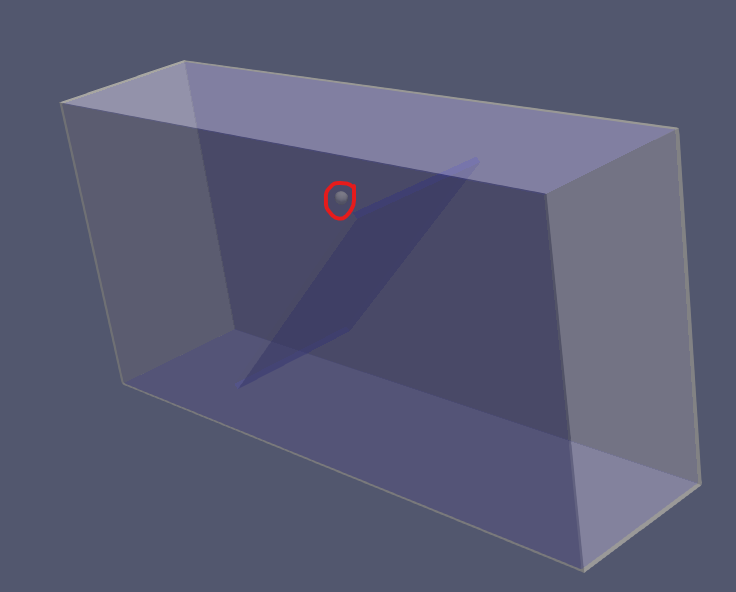
Now that we have everything downloaded and installed, we are ready to start EasyCFD. The first stage is geometry prepatation through enGrid.

1. Open enGrid and import your geometry in STL format
   1. Click the **Import** tab and then **STL**
2. Use the default tolerance of 1e-10 and click OK
   1. You should now see your geometry loaded in the enGrid graphics window
3. If your geometry has only one boundary condition (e.g. just a wall), skip to step 7
4. Define your boundary conditions (e.g. inlet, outlet, wall). To define a boundary, start by placing the mouse cursor over a face belonging to the desired boundary and press the P key. This is the hotkey to select a face.
   1. Besides the wall, since it is usually the largest boundary, the order of defining boundaries doesn’t matter.
5. Press the S key to set the boundary code.
   1. Use the default Feature Angle
      1. The lower the feature angle, the less inclusive enGrid is in selecting the surrounding faces that may belong to the same boundary. A completely flat boundary only needs a feature angle of 1, but more rounded or angular boundaries would need a higher feature angle. An angle may be too small and not select every face, but this can be accounted for the next steps.
   2. For the first boundary you set, use a boundary code value of 1. Boundary codes are how enGrid differentiates between boundaries. For each boundary to come, increase the boundary code value by 1 (e.g. inlet = 1, outlet = 2, wall = 3)
   3. Select the bubble for “Process all cells (even invisible ones)
6. Click OK
7. Press the E key to edit the boundary conditions and assign names
   1. If you skipped to here from, step 3, double-click the BC-name entry and change it to a suitable name with no spaces or special characters and proceed to step 8.
   2. The BC-index corresponds to the boundary code you defined earlier. Under the BC-name for that BC-index there will an entry “BC” followed by the BC-index
   3. Double click the BC-name entry for the BC-index you just defined, and change the entry to a suitable name with no spaces or special characters
      1. If the boundary is an inlet, the name must have “inlet” in it. If the boundary is an outlet, the name must have “outlet” in it. If the boundary is a wall, the name must have “wall” in it.
      2. There may be features of interest in the geometry that function as a wall, but would preferably be treated as its own boundary. The user then may use any name as long as it doesn’t include “inlet”, “outlet”, or “wall”
      3. The user may name a boundary as “refine”. Refine functions the same as a wall, but provides more refinement and smoothness when being meshed in OpenFOAM. Refine is recommended for rounded features.
8. Click OK
9. Repeat steps 3-8 for each remaining boundary to be defined
10. Click the **Export** tab and export the prepared geometry as SU2
    1. SU2 format provides point coordinates as well as the boundaries that these coordinates belong in
11. Your geometry is now prepared to be processed through ezFOAM

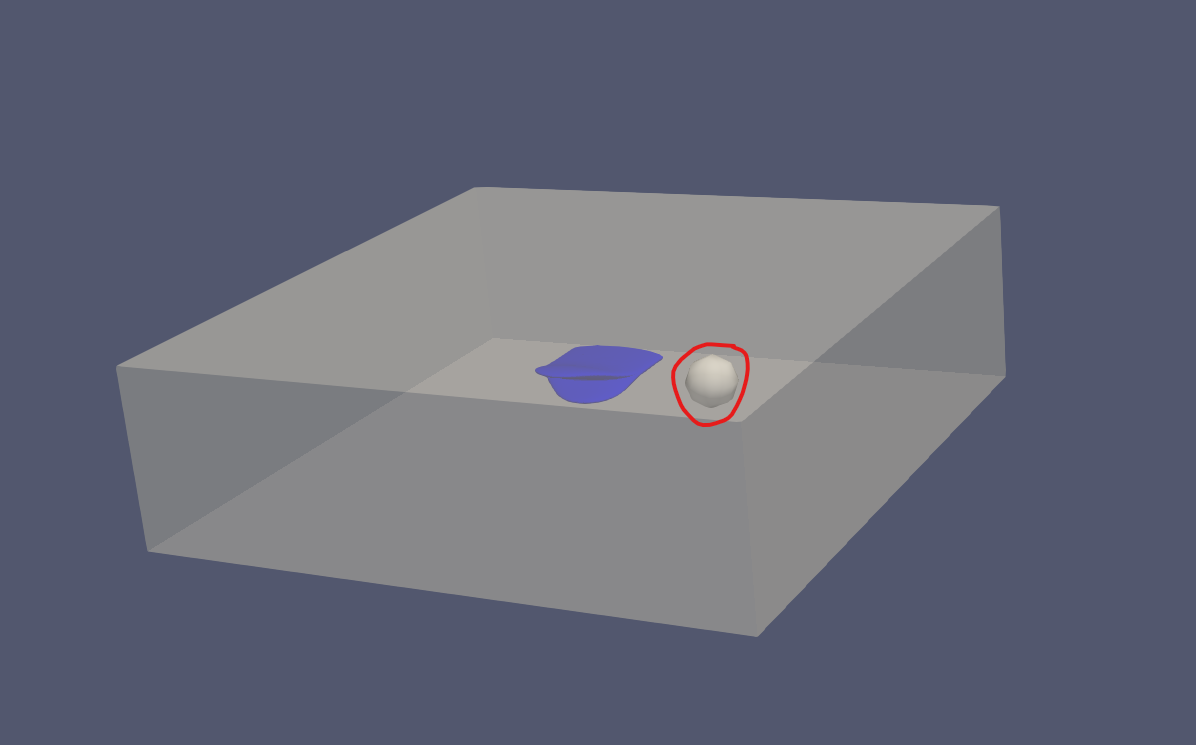
**Stage 2: Case Construction through ezFOAM**

1. In the folder where you downloaded the contents of the Python Code folder, select the ezFOAM Python file
2. You should now see a Python terminal window and the ezFOAM window
   1. Do not close the Python terminal window until you are done building an OpenFOAM case
   2. The Python terminal is used for further input when building a multiphase simulation
3. Select simulation destination
   1. Choose where you want your OpenFOAM case to go in your computer by clicking Browse. This destination can vary every time ezFOAM is used.
4. Select su2 geometry file
   1. Click Browse and choose the file of your desired case geometry in SU2 format.
5. Specify inlet flow rate and units
   1. Type the desired flow rate value in the box and click Enter. The flow rate value can be in entered in E-notation (e.g. 1e-5). Then, choose the inlet flow rate unit by clicking either the m/s box for velocity or the m3/s box for volumetric flow rate.
6. Specify flow type
   1. Choose what kind of simulation you want. EzFOAM currently solves air steady, water steady, and multiphase simulations.
7. Enter the number of processor cores you wish to use (1 or more)
   1. Type the number of processor cores you wish to use in the box, then click Enter.
   2. If you specify more than 1, OpenFOAM will run in parallel.
   3. To check how many cores your machine has, check the Performance tab in Task Manager which can be opened through ctrl+alt+del
8. Is the flow internal or external?
   1. Click the “Internal” box for internal flow and the “External” box for external flow
9. Select “Build OpenFOAM Case”
10. Congratulations, you’ve built a complete, ready-to-run OpenFOAM case!

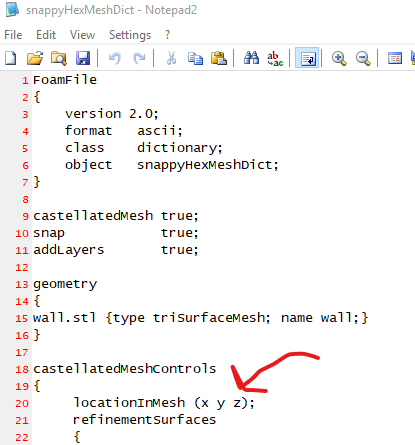
**Stage 3: Simulating through blueCFD-Core**

1. Right-click the resulting test\_clean folder and open with blueCape
   1. If this is the first time your machine is running blueCape, then close the terminal and repeat step 1.
2. In the terminal, enter **./just\_prep**
3. Wait for the prompt, then **enter** **parafoam** in the terminal.
4. In ParaView, **select** **Apply**, which will display the blockMesh.
5. Then, open all the geometry STL files in the same ParaView session, found in constant/trisurface. **Click Apply**
6. Go to the **Sources** menu near the top of the window and select the **Sphere**
7. Decrese the blockMesh and STL opacity in the Properties browser, found in the bottom left of the graphics window
8. Vary the center coordinates as necessary to find a suitable locationInMesh. It may be helpful to decrease the radius of the sphere, and decrease the opacity for visualization purposes.
   1. A suitable locationInMesh must be in empty space, not touching any surface.
   2. If the flow is internal, the center coordinates should be inside the STL geometry
   3. If the flow is external, the center coordinates should be outside the STL geometry
      1. 

For internal flow, the sphere is contained by both the STL and blockMesh



* + 1. For external flow, the sphere is only contained by the blockMesh

1. Go into test\_ran’s snappyHexMeshDict and make the necessary change to (x y z)
   1. 
2. Finally, open test\_ran in blueCape and enter **./snap\_solve**