DexOF: DEXter OpenFOAM interface

Release 0.1

Revised: December 30, 2021

Stevens Institute of Technology
Questions/Comments: Kishore Pochiraju (kpochira@stevens.edu)
Emil Pitz (epitz1@stevens.edu)

What is DexOF?

DEXter (Design Exploration and Testing For Engineers) is a Stevens-built platform that allows system design space representation (using the .dex language), exploration (DoE, Sampling), and Optimization (variety of Multi-Objective GA and other algorithms). Design space abstractions, operations on design spaces (e.g., Unite*: a regularized design space union), parameter dependencies (uses NetworkX graph) , simulation execution paths (with multi-fidelity) are part of the DEXter platform core. They have been developed at Stevens Institute of technology. DEXter uses several open-source packages. The exploration and optimization tools are powered by pymoo (for Optimization), SMT (for exploration and surrogation), and Altair (for limited visualization).

DexOF is the DEXter's simulation interface to OpenFOAM (CFD) software. Currently, it is set up to automate lift, drag, and moment coefficient estimation for submerged bodies in laminar and turbulent flow regimes.

How to access and use DexOF?

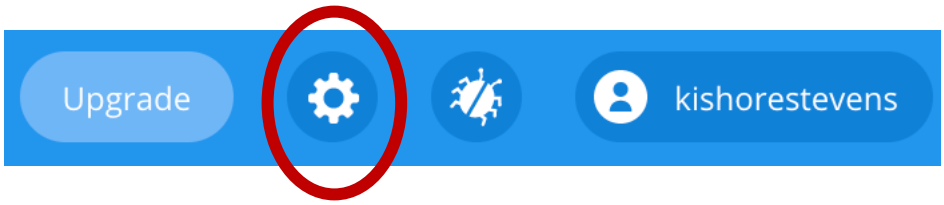
Set up Requirements:

1. (For Windows/Win64 Users Only) Install Windows Subsystem for Linux – Install Ubuntu 20.04 **(Requires: Windows 10 version 16237.0 or higher)**
   Install WSL2: https://ubuntu.com/wsl
   Use this link to install:
   https://www.microsoft.com/en-us/p/ubuntu/9nblggh4msv6?activetab=pivot:overviewtab
   (Installation will ask for a username and a password – please remember these you will need them)

   (For Win64 **and** Linux Users) After installation of WSL/Ubuntu 20.04 LTS, Look for the Ubuntu Shell in the windows menu (or search for it in the command box) and open a new shell.
   You have to permit docker to run within that shell: Tyep:
   - `sudo groupadd docker`
   - `sudo usermod -aG docker ${USER}`

     Close that terminal and restart and new one. (One time set up only.)

2. (Mac, Linux, Win64)
   Install Docker Desktop: https://www.docker.com/products/docker-desktop
   a. Verify Docker installation – look for the docker icon in the application tray
   b. Let docker know you have Ubuntu on your windows as a subsystem
   c. **[Win 64 only]** In Docker-Desktop settings (Gear icon on top left)



   Go to **Resources – WSL Integration** (left side bar)
   Turn on **Ubuntu 20.04** switch
   (you will see Ubuntu on this screen if step 1 was a success.)
   Apply and Restart

3. There are only 3 commands to run set up problem and run OpenFoam
   See section below for step-by-step

```
I)  docker run --rm -v ${PWD}:/home/aimed_user/dexof_work kishorestevens/dexof
    /home/aimed_user/dex_of/setup_dexof.sh
II) cd ./test_casestudy
III)./run_dexof.sh rough_mesh_8cores.dex seaglider.stl 1
```

In more detail (all the nitty-gritties**)**

- Open a Linux/MAC/WSL-Ubuntu terminal.
- Make a new working directory – say, *mkdir test_dexof*
- ***On Linux Hosts: Avoid the uid/gid mapping issues by simply setting permissions so everyone can write into that folder: chmod 777 test_dexof***
- Cd into that directory - *cd test_dexof*
- Run the following commands (in red) to get the required scripts and a test case.

i) Get the docker image and the script for running DexOF (copy the red command below and paste into the terminal)

```
docker run --rm -v ${PWD}:/home/aimed_user/dexof_work kishorestevens/dexof
/home/aimed_user/dex_of/setup_dexof.sh

   ****************************************
   **     DEXTER-OPENFOAM INTERFACE      **
   **  Stevens Institute of Technology   **
   **  No warranties: use at your own risk **
   ****************************************
   cd to test_casestudy folder and run the following command to run 1 degree aoa on seaglider
   Modify the dex files as necessary to make changes to computational grids or other options
     ----
    ./run_dexof.sh <dexfile> <stlfile> <aoa_in_degrees>
    FOR EXAMPLE:
   ./run_dexof.sh rough_mesh_8cores.dex seaglider.stl 1
    RUNS seaglider with a coarse mesh and at 1 degree aoa.
     ----
   *** ALL DONE  ***
```

This command will create an example case folder with one .stl file (seaglider.stl), four dex files (corresponding to four mesh densities – rough, coarse, medium,and fine and 3 different CPU core resources), and the script that will run open foam: run_dexof.sh.

WARNING 1: Default mesh files are set for 8 cores. If your system has less, you will need to adjust that in the .dex file. Look for these two parameters below:

subdomains,input,discrete,8
computegrid,input,string,(2 2 2)

These parameters are setting 8 cores in a 2x2x2 (X x Y x Z) grid.

If you have only 4 cores,edit these lines in the dex file:

subdomains,input,discrete,4
computegrid,input,string,(2 2 1)

If you have only 2 cores,edit these lines in the dex file:

subdomains,input,discrete,2
computegrid,input,string,(2 1 1)

If you have only one care, please buy a new computer. You deserve it!

II) Use the following command to run OF using the same docker container pulled before.

```
cd ./test_casestudy
```

```
./run_dexof.sh <dexfile> <stlfile> <aoa_in_degrees>
 FOR EXAMPLE:
./run_dexof.sh rough_mesh_8cores.dex seaglider.stl 1
```

Notes:

- In this setup, only the stlfile and the aoa can be overwritten from the script's command line. All other variables must be edited in the .dex file.
- Other scripts and OF templates are in the docker container in the /home/aimed_user/dex_of/ directory. Feel free to edit them as you need them.
- Warnings during pre-processing are ok. Ignore.

```
<frozen importlib._bootstrap>:219: FutureWarning: Passing (type, 1) or '1type' as
a synonym of type is deprecated; in a future version of numpy, it will be
understood as (type, (1,)) / '(1,)type'.
Your mesh is not closed, the mass methods will not function
            correctly on this mesh. For more info:
            https://github.com/WoLpH/numpy-stl/issues/69
Your mesh is not closed, the mass methods will not function
            correctly on this mesh. For more info:
            https://github.com/WoLpH/numpy-stl/issues/69
<frozen importlib._bootstrap>:219: FutureWarning: Passing (type, 1) or '1type' as
a synonym of type is deprecated; in a future version of numpy, it will be
understood as (type, (1,)) / '(1,)type'.
```

- **You may replace STL file with your own file.** Computation box will be automatically set up as defined in the configuration dex file. (See Warning #1 and Appendix-A)
- **WARNING #2:** Note that the flow is along +x axis and lift along +Y axis. Please make sure that this is the case. Rotation of the STL File into this coordinate frame has not been implemented (Future Release). We assume that you already have the STL has +x oriented along the flow direction.
- **You may edit the configuration file (.dex) files as you like.** The parameters in the configuration files (.dex) files are described in Appendix-A. These are propagated into OpenFOAM input files at various places by the dex_of.py script located in the container.

Appendix-A

DexOF configuration file structure.

Syntax:
*** Comment*
** DEXter command – dex_of ignores these*
Parameter, …, Value

*All lines that do not begin with a * are parameter definitions. They are comma-separated filed. First field is the parameter name, and the last field is the parameter value. All the other fields are for DEXter use and dex_of.py users can safely ignore them.*
(All parameters are bolded-red in the listing below)

```
*** AIMED Project
*** Stevens Institute of Technology,2021
***
** UUV CL CD COMPUTATION WITH OPEN Foam
***
*** Syntax: ParameterName,,,ParameterValue
*** All lines without * in front are parameters
*** Example:
*** casefoldername,input,string,UUV_aoa0_core32
*** line.split(',')[0] --> parameter Name
*** line.split(',')[-1] --> Parameter Value
****
**** NOTES: 1) CHANGE ONLY THE VALUES (after the last comma)—Names are fixed.
**** 2)runopenfoam, backgroundof are not implemented in release 0.1
**** 3) Values in blue are overwritten by command line from shell scripts.
**** 4)dex_of.py can overwrite them all.
****
*DesignSpace,Name=UUV_Coarse
*Parameters
***
*** RUN CONTROL PARAMETERS
***
***
casefoldername,input,string,UUV_aoa0_core32
runopenfoam,input,string,yes
backgroundof,input,string,no
kOmegaTol,input,continuous,1e-8
upTol,input,continuous,1e-8
maxiter,input,discrete,500
***
*** Geometry Details
***
infile,input,string,UUV_Orig.stl
outfile,input,string,UUV0.stl
aoa,input,continuous,1.0
***
*** if the stl file needs to be scaled.
scalex,input,continuous,0.001
```

```
scaley,input,continuous,0.001
scalez,input,continuous,0.001
***
*** Computational Grid -- meshing/solution is run on 32 cores
***
subdomains,input,discrete,8
computegrid,input,string,(2 2 2)
***
*** Flow Characteristics
***
Uinlet,input,continuous,1.22
kinematic_viscosity,input,continuous,1.736124635e-6
density,input,continous,1027.0
kInlet,input,continous,0.01
omegaInlet,input,continuous,57.
***
***
** Meshing details (cell sizes in meters)
** Domain Size scaling -- the domain size will be N times the UUV size in
front of the domain
** X is the flow direction, Y is the lift direction and Z is the span
direction.
** Inlet is to the front.
***
DomainSizeXFront,input,continuous,2
DomainSizeXBack,input,continuous,2
DomainSizeYTop,input,continuous,2
DomainSizeYBot,input,continuous,2
DomainSizeZLeft,input,continuous,2
DomainSizeZRight,input,continuous,2

** Block mesh will create domain/cellSize? cells in the ? direction
cellSizeX,input,continuous,0.1
cellSizeY,input,continuous,0.1
cellSizeZ,input,continuous,0.1
***
*** SNAPPY HEX MESH REFINEMENT
***
maxLocalCells,input,discrete,1000000
maxGlobalCells,input,discrete,8000000
nsurfacelayers,input,discrete,10
***
```