Protocol

Wind Load Calculation in OpenFOAM

Author(s):	Johannes Kneer and Rebecca Wong
Beginning Date:	01. Februar 2019
Last Edited:	31. Juli 2019
Responsible Persons:	Rebecca Wong, Johannes Kneer
Kurzzusammenfassung:	Setup a case to calculate the wind load on a SunOyster. Further automate the case for project-based wind load estimation.

Document history:

- 28.02.2019 Draft of Document
- 30.04.2019 Final Document

Table of Contents

1. Goals of Development

An OpenFOAM case is set up to calculate realistic 3D flow fields around SunOyster8 (SO8). The case will be improved to take into account all relevant physical phenomena. The case will be automated to understand the wind load based on the main parameters of the SO8 set up: opening angle of the mirror, angle of attack, and various terrains affecting flow varying from open ground to urban/forested setting. The results will be used for development of the SO8. For example, the flow field yields will provide important information on the best placement of the wind sensor. The results can also be used to estimate excitation frequencies due to fluid-structure interaction.

1.1. Objectives

There are three main objectives:

1) to accurately model the atmospheric boundary layer profile on a typical day and normal wind load conditions at a variety of positions/angles of attack

2) to accurately model the maximum wind loads on the SO8 for fully open at 55 $kmhr^{-1}$ (15.28 ms^{-1}) and closed at 150 $kmhr^{-1}$ (41.67 ms^{-1}) at a variety of positions/angles of attack

3) to accurately model the vortex shedding and determine resonance frequencies of the SO8 over time at a variety of positions/angles of attack

1.2. Procedure

Open-source Field Operations And Manipulations (OpenFOAM) is a programming library with a significant amount of command line tools. Using a scripting environment, OpenFOAM enables the setup of complex, automated flow calculations. The only inputs required are parameter adjustments. Limited fluid dynamics knowledge is required to gain useful results for research and project development.

1. Set up a “rough” steady-state case using atmospheric boundary conditions (BCs) and simpleFoam, an incompressible turbulent solver
2. Improve discretization using snappyHexMesh and establish mesh independence
3. Use function objects such as forceCoeffs to calculate drag and lift
4. Run a transient simulation with the same setup
5. Write Python (pyFoam) wrapper to automate case setup (orientation, opening angle, and terrain)
6. Define standard set of calculations to run for project development

2. Background Knowledge

In this section, an overview of computational fluid dynamic programs, in particular OpenFOAM as well as the atmospheric boundary conditions, is provided.

2.1. Computational Fluid Dynamics

Computational fluid dynamics (CFD) has been extensively used to model flow phenomena realistically and relatively inexpensively when compared to actual experiments. Though there are a variety of commercial programs that perform CFD measurements with regards to wind phenomena, including ANSYS Fluent and SimScale, OpenFOAM was chosen for this project as it provides tools and accuracy comparable to these programs while also allowing for the flexibility of open-source programming and editing.

2.1.1 OpenFOAM

OpenFOAM is a collection of open-source, C++ libraries capable of solving complex physical problems. The program contains many tools and frameworks useful for creating solvers, flow types, and boundary conditions for modeling fluid dynamics in a variety of environments. In this study, we are particularly interested in using its capabilities to model atmospheric boundary layer conditions and extreme conditions.

Previous studies have confirmed that although large-eddy simulations (LES) tend to be the most accurate as they better predict vertical velocity and Reynolds shear stress, the computational requirements are at least one order of magnitude greater than for Reynolds-Averaged Navier-Stokes (RANS) equations. Due to limited computational resources as well as the RANS model’s reasonable modelling abilities and performance, the RANS $k - \varepsilon$ model is adopted, which has been shown to model the recirculation behind bluff bodies accurately. In the future, for more accurate pressure and drag coefficient calculations, the $k - w$ shear stress transport (SST) model may be adopted.

2.2. Atmospheric Boundary Layer

The atmospheric boundary layer (ABL) constitutes roughly the atmospheric height from zero to 200 m. The ABL’s velocity in neutral conditions is best modeled using a logarithmic profile relative to height (the log wind profile).

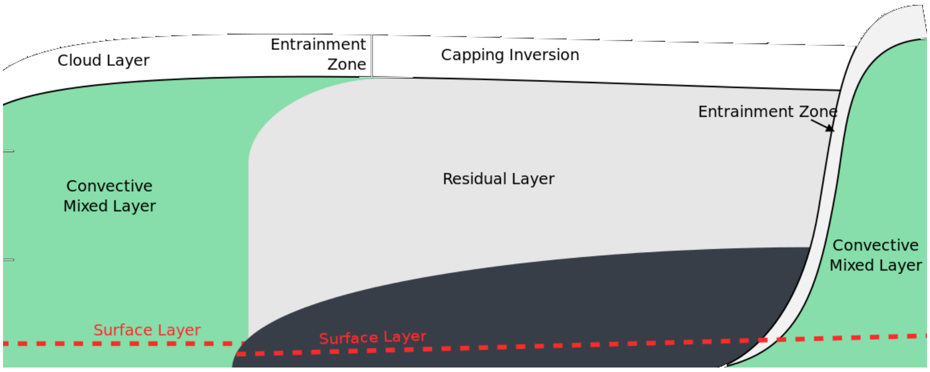

Fig. 1: ABL profile including surface layer, where neutral conditions are an appropriate assumption.

Currently the ABL neutral boundary conditions provided in OpenFOAM v1812 provide inlet conditions based off of Richards and Hoxey's (1993) "Appropriate boundary conditions for computational wind engineering models using the k-epsilon turbulence model" with modified constants provided by Hargreaves and Wright's (2007) "The use of commercial CFD software to model the atmospheric boundary layer". The following inlet conditions provide fully developed, equilibrium profiles for mean wind velocity (U), turbulent kinetic energy (k), turbulent dissipation rate (ε), and turbulent dissipation Prandtl number (σ_ε):

$$U = \frac{u_*}{\kappa} ln\left(\frac{z + z_0}{z_0}\right) \tag{1}$$

$$k = \frac{u_*^2}{\sqrt{C_\mu}} \tag{2}$$

$$\varepsilon = \frac{u_*^3}{\kappa(z + z_0)} \tag{3}$$

$$\sigma_\varepsilon = \frac{\kappa^2}{(C_{\varepsilon 2} - C_{\varepsilon 1})\sqrt{C_\mu}} \tag{4}$$

where u_* is the friction velocity, κ is the von Karman constant (normally takes a value between 0.40 – 0.42; in this case 0.41), z_0 the aerodynamic roughness length, and C_μ, $C_{\varepsilon 1}$, and $C_{\varepsilon 2}$ (in this case 0.09, 1.92, and 1.13, respectively) are experimentally-determined constants.

The following assumptions are made with these BCs:

1) zero vertical velocity
2) pressure is constant in vertical and streamwise directions
3) constant shear stress in the boundary layer
4) the turbulent kinetic energy and dissipation rate satisfy their transport equations

Though these equations are complete and provide reasonable accuracy, it has been shown that this set of equations creates 1) an inconsistency between the fully developed ABL inlet profiles and the rough wall function formulation and 2) the inlet profile for k assumes a constant value with height, which contradicts wind tunnel measurements. Further investigations into these boundary conditions are made by Yang et al.'s (2008) "Influences of equilibrium atmosphere boundary layer and turbulence parameter on wind loads of low-rise buildings" and Parente et al.'s (2011) "Improved k–epsilon model and wall function formulation for the RANS simulation of ABL flows".

2.3. Terrain Considerations

The velocity profile is significantly impacted by roughness as shown by a variety of papers. As explained previously, z_0 can be set to model various terrain conditions; for example, an open flat terrain of grass with a few, isolated obstacles has a z_0 of 0.3 m. In the case of a suburb or a forest, this value increases to 1 m, and within a city center with many low- and high-rise buildings, this value increases to 2 m or more.

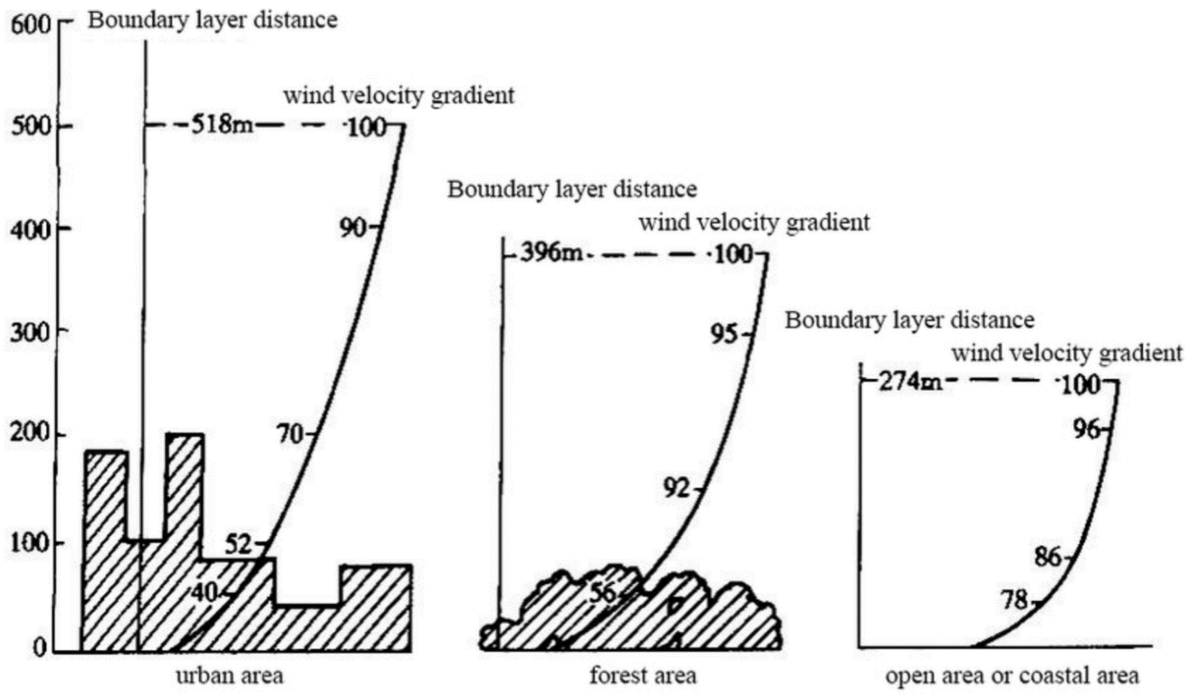

Fig. 2: Impact of various terrain roughnesses on the wind velocity profile

To establish the values for U_{ref} and z_{ref}, wind stations collect wind speed data at various locations. Generally, this data is collected at a reference high of 10 m over some years. This data is then extrapolated using the log wind profile described earlier to model the velocity at a variety of heights within the ABL. For the case of the SO, any reference height would be plausible for a typical ABL profile but it may be more beneficial to use a reference height closer to the SO, particularly if the interest is in returning accurate drag and lift forces at the SO's actual position. Additionally, if the mast (3 m) is ever used to collect data, this would then give us the ability to set z_{ref} and U_{ref} with experimentally-determined data for each area of interest.

Fig. 3: Setting the reference height to the wind mast would allow for appropriate tests on the opening and closing of the SunOyster at typical day-to-day velocities.

3. Implementation

3.1. Assumptions

The current model assumes that the main source of drag is due to the mirrors themselves; thus, other components of the SO are neglected. The log law of the wind is also a simplification of the relationship between height and wind speed as is the use of z_0 for modelling various types of terrain roughnesses.

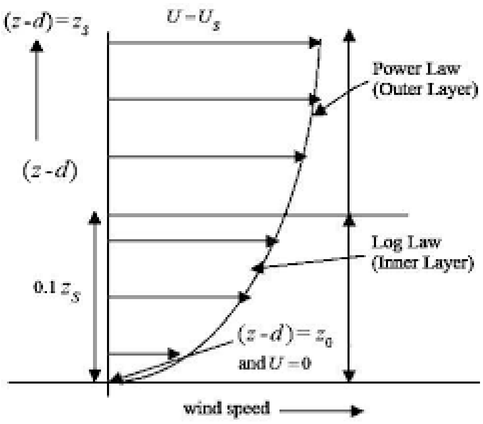

Fig. 4: The log law of the wind for heights around 0 to 200 m

3.2. Computational Mesh

The computational mesh is 24 m by 14 m by 42 m in x-, y-, and z-direction respectively. The box is set such that the SO8's torque tube is at origin (0,0,0) and such that the inlet is placed $5H$ (where H is taken to be the height of the SO8, namely 1.7 m) before, above, and on either side of the SO8 and $15H$ in the wake of the SO8. The size of the mesh was taken from recommendations by several papers such that the boundary conditions do not influence the results of the experiment. This arrangement also keeps the blockage ratio to less than 3 percent (results from tests where the blockage ratios are significantly larger than 6 percent can be influenced by the boundary conditions). In order to use the snappyHexMesh command, the .stl file needs to be placed in the constant/triSurface folder and should be converted from mm (the original SolidWorks file) to m using the command surfaceConvert -scale. The angle of the SO8 can then be manipulated by the Python script as mentioned below.

3.3. Boundary Conditions

The U profile is modified so that the top boundary has a fixedShearStress boundary condition following the equation for wall shear stress:

$$\tau = u_*^2 \varrho \qquad (5)$$

where ϱ is the density of the medium (air, taken to be 1.225 kgm^{-3} at sea level, 1 atm, and 15°C). The fixed shear stress should be in the direction opposite of the free stream velocity. This BC is only important if the top of the domain is of interest; otherwise, the top can also simply be modeled with slip. All other BC profiles remain the same as the turbineSiting tutorial (the sides remain as slip boundary conditions with the inlet profiles all based off of atmBoundaryLayerInlet). Finally, the viscosity of air is specified in constants/transportProperties and is correspondingly 1.81×10^{-5} $kgm^{-1}s^{-1}$ at 15°C.

For tests where we are interested primarily in the drag coefficient and high loading on the SO (which would not be representative of normal ABL conditions), a fixedValue inlet for the U profile is used such that the velocity of interest is the free stream velocity for the entire domain. Additionally, a slip boundary condition is used for the top boundary condition since the top of the domain is not of interest.

3.4. Initial Conditions

The current case is modified off of the turbineSiting tutorial provided in $FOAM_TUTORIALS/incompressible/simpleFoam. Current initial conditions are based off of Hargreaves & Wright's (2008) paper. These files can be found in 0/include/ABLConditions and 0/include/initialConditions. z_{ref} is set to 60 m with corresponding u_{ref} of 10 ms^{-1}, z_0 to 0.01 m (though these three values can be modified to accurately reflect the data collected as mentioned above), p to 82714.2857 $m^{-2}s^{-2}$ (note that this indeed corresponds to typical atmospheric pressure of 101,325 Pa since OpenFOAM normalizes by density), k to 1.36 m^2s^{-2}, and ε to 0.625 m^2s^{-3}. Note that the *zDir* must be correctly oriented to align with the standard axes provided by the .stl file created in SolidWorks. Updating the flowVelocity (free stream) initial condition with the free stream value is unnecessary when the ABL profile inlet is used (when using the fixed inlet, update the flowVelocity to accurately reflect the desired free stream velocity). By using these initial conditions, convergence with solver simpleFoam can be achieved.

The system is very sensitive to changes in the initial conditions (particularly k and ε); many trials show non-convergence. Interestingly, even when more accurate k and ε values are used from the converged case, the trial does not achieve convergence. Instead, it is recommended that the command mapFields -consistent is used stabilize and start a case rather than the adjustment of these initial values.

3.5. Force Coefficients

The forceCoeffs file is taken from the motorBike tutorial ($FOAM_TUTORIALS/incompressible/simpleFoam) and is modified such that the $liftDir$ (upwards and normal to flow direction), $dragDir$ (in same direction and parallel to flow direction), and $pitchAxis$ (axis around which the SO8 opens and closes) match those of the modified flow direction. These reference axes should be adjusted according to the direction of the flow velocity. The *lRef* is taken to be *H* and *Aref* is taken to be the total area of the mirror surfaces of 6.8 m^2 (this area should be the same as the projected area of the surface on the plane orthogonal to the wind direction). The $CofR$ represents the center of rotation which is taken to be the point about which the mirror rotates around (set at the origin). This $CofR$ accurately reflects the actual motion of the SO8 about its pivot point (care should be taken that when the SO8 is fully open that it is still within the bounding box). The $CofR$ in relation to the ground is 1.1 m up and 0.88 m from the base of the mirror. This point will not need to be redefined by calculating the corresponding y- and z-components using sine and cosine respectively when the origin is set to the position about which the SO8 opens. Note that to accurately calculate the force and force coefficients, the *magUInf* must be modified to match the free stream velocity, which is either the velocity near the top of the domain or, in the case of a fixed inlet, simply the free stream velocity. Additionally, rho needs to be specified as $rhoInf$, which tells the solver to treat the medium as incompressible. The coefficients for drag (C_d), lift (C_d), and moment (C_m) are calculated as follows:

$$C_d = \frac{2F_z}{\varrho U^2 A} \tag{6}$$

$$C_l = \frac{2F_y}{\varrho U^2 A} \tag{7}$$

$$C_m = \frac{2F}{\varrho U^2 AL} \tag{8}$$

where ϱ is the density, U the velocity, L the characteristic length, A the area, and F the force in the corresponding direction. Note that to get the actual C_d, C_l, and C_m values, one must multiply the value by ϱ.

To calculate the force coefficients at different positions, the STL file will need to be properly rotated about different axes. This process can be accomplished by the use of surfaceTransformPoints. Additionally, a Python script using the numpy-stl library has been created to automatically rotate the STL file either about the pitch axis (x-axis) to simulate opening and closing of the SO8 at 30° increments and about the yaw axis (y-axis) to simulate different angles of attack. The files are automatically produced and saved to the working directory.

3.6. Discritization Schemes and Run Time Control

The fvSolution file is located in system and is roughly the same as those from most tutorials. Here one can adjust the residualControl, which is currently set to p, U, k, and ε at 1.0^{-4}. One can also adjust the nNonOrthogonalCorrectors. It is recommended that meshes with non-orthogonality between 70 and 80 use 3 correctors, between 60 and 70 2, and between 40 and 60 1 (we use 1).

The schemes can be edited in system/fvSchemes. The ddtScheme for steady state is steadyState. A mix of first-/second-order schemes are used. The convective terms are set such that gradient schemes (gradScheme) are Gauss linear and cellLimited Gauss linear for U; for the divergence schemes (divScheme), the bounded Gauss upwind scheme is used for k and ε with bounded Gauss linearUpwindV for U (the V indicates that the limiter is calculated based on the direction of the most rapidly changing gradient, which is more stable but less accurate). linearUpwind represents a blended first-/second-order scheme. As Gauss upwind is a first-order scheme, it is robust but potentially too diffusive. However, attempting to implement a blended first-/second-order turbulence model using

Gauss linearUpwind schemes for k and ε, even with underrelaxation, eventually led to divergence at most angles. Reported results thus use the scheme mentioned above.

To run transient simulations, the system/controlDict should be adjusted from the steady state case. Namely, the adjustTimeStep should be used in combination with maxCo (typically should be less than 1) so that the Courant number is kept below 1 using automatically adjusted time steps. Additionally, make sure that runTimeModifiable is yes. Potentially, a future test of time step independence could be performed to determine the acceptable limit in time step to speed up tests. A Fast Fourier Transform (FFT) analysis could then be performed on this data to determine the exact vortex shedding frequency.

The simulation must first be initialized with a first-order Euler scheme (gradScheme is Gauss linear and divScheme Gauss limitedLinear 1 except for U, which uses Gauss linearUpwind) before switching to higher order schemes. It is possible also to mapFields from the corresponding steady state case. After stabilization, Crank-Nicholson 0.9 ddtScheme is used for increased accuracy, where 0 represents Euler method and 1 represents pure Crank-Nicholson scheme. For even greater accuracy, cellMDLimited Gauss linear 0.5 is used as the gradScheme to decrease diffusivity. Though the leastSquares method may be more accurate, it is also more oscillatory. For the divScheme, Gauss linearUpwind is used for all solved variables. For the diffusive terms, laplacianSchemes is set to Gauss linear limited 0.777 where 0/uncorrected offers greater accuracy and 1/corrected more stability (uncorrected should only be used with meshes with non-orthogonality lower than 5°). snGradSchemes (component of the gradient normal to a cell face) is set as limited 0.777. interpolationSchemes (cell to face interpolations of values) is left as linear. It is additionally recommended that the nOuterCorrectors is set to a value of around 50 to 300 depending on the number of iterations necessary to find a steady state/stable solution within each time step (in this case, around 40) with nCorrectors set to 1 (1 to 3 depending on stability), which is the number of iterations of pressure correction. All other elements, including relaxation factors and schemes, are taken from transient tutorials provided by OpenFOAM. Surprisingly, unlike in the steady state case, the transient case is relatively stable with this combination of schemes after stabilization is achieved.

Though steady state simulations can be run with only a single processor, it is recommended to use the developed Allrun script for transient cases due to the substantially increased computation time. The decomposition is done automatically using the scotch method assuming the user knows how many processors are available (in our case 4). The script automatically produces logs for all processes sequentially for monitoring.

4. Results

4.1. Mesh Independence Study

It has been shown that when the computational mesh is not fine enough, orders of magnitude of error for the lift and drag coefficients are obtained. At the same time, a mesh that is too refined may be so computationally expensive that running simulations becomes too time-consuming. Therefore, it is important to achieve a balance between the number of elements and the desired accuracy of the result. A variety of snappyHexMeshes with different refinement levels for the ground surface and mirror surface were tested and used to determine the impact of the mesh resolution on the system. $y+$ values did change with the refinement level of the mesh as expected, but with greater refinement, both minimum and maximum $y+$ values decreased (lowest range was about $7 \leq y+ \leq 700$ for the mirror surface). The range of $y+$ values never reached the recommended $30 \leq y+ \leq 300$ range for wall functions (sometimes they were close initially for the mirror surface at t = 0 but quickly diverged from the optimum during the simulation). They were usually a magnitude of order greater for the ground. Due to the use of refinement levels in snappyHexMesh, it was difficult to finely control the $y+$ values and the mesh generation as a whole. Adding layers to the ground and mirror surface using snappyHexMesh resulted in $y+$values within the buffer sublayer of $5 \leq y+ \leq 30$ and often resulted in meshes with increased non-orthogonality and skewness. The chosen mesh has $70 \leq y+ \leq 3000$. Despite the non-optimal $y+$ values, the results appear to be reasonable.

A mesh independence test was conducted to determine the mesh refinement required to achieve appropriate results. All meshing was performed using snappyHexMesh tool. Tests were conducted to include no refinement (except directly around object) (mesh 1), refinement directly about SO8 (meshes 2 and 3), a refinement sphere about SO8 (mesh 4), a refinement sphere and band running from front to back of domain parallel to wind direction (mesh 5) and a more refined version of this mesh (mesh 6), and refinement bands spanning both the front and back of the object and side to side as recommended by many studies (mesh 7). Images of the representative meshes is given in Fig. 5. The C_d and C_l values were primarily used to determine acceptability of mesh as well as a visual analysis of the flow vortices behind the mirror. The following tests are summarized in Table 1 below:

Mesh Number	Number of Elements	C_d	Percent Difference	C_l	Percent Difference
1	16,000	1.86	8.1	0.0098	97.8
2	21,000	1.77	2.9	0.43	4.4
3	93,000	1.84	7.0	0.46	2.2
4	401,000	1.79	4.1	0.45	0
5	493,000	1.68	-2.3	0.44	-2.2
6	1,030,000	1.69	-1.7	0.44	-2.2
7	3,250,000	1.72	0	0.45	0

Table 1: Mesh independence study comparing C_d and C_l values for various meshes

Mesh 5 was chosen based on its accuracy for both coefficients as well as through visual examination of vorticity structures for the smallest number of elements. There is a gap in the number of elements due to the large increase in number of elements required when refining a region of mesh rather than only the SO8 as well as the difficulty in controlling refinement using snappyHexMesh.

This mesh has the mirror edges refined to level 5 with the rest of the mirror refined to levels between 4 and 5. The refinement sphere is level 3, and the refinement band is level 2. nCellsBetweenLevels is set to 5 for more graduation between the different levels. Other parameters of snappyHexMesh are otherwise unaltered.

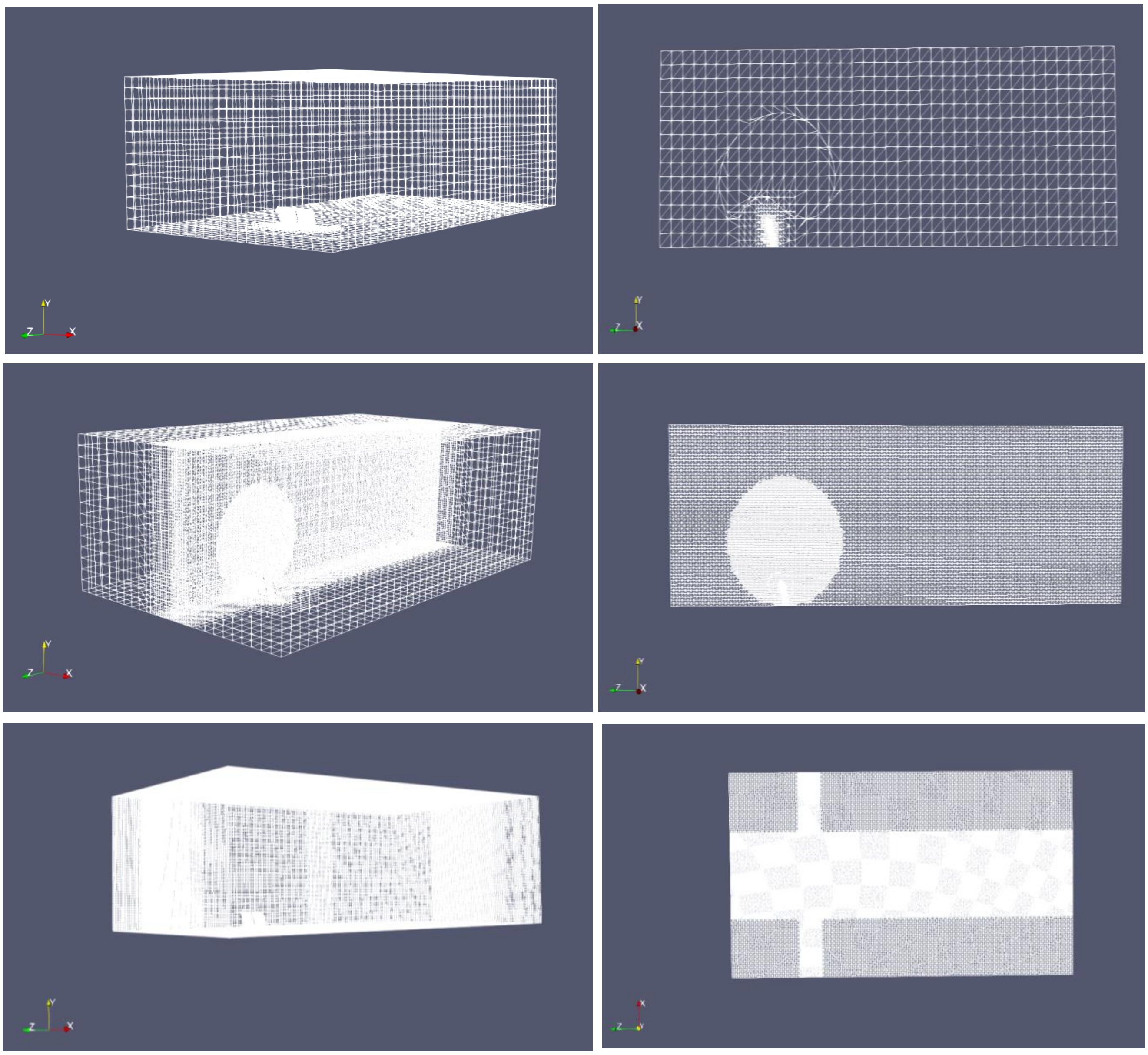

Fig. 5: Representative meshes and mesh slices to show grid refinement are shown. From top to bottom: Mesh 3, Mesh 5, and Mesh 7

4.2. Steady State Simulations

The Reynolds number is a dimensionless number that dictates the ratio of inertial to viscous forces:

$$Re = \frac{\varrho U L}{\mu} \quad (9)$$

where ϱ is the density, U the velocity, L the characteristic length, and μ the dynamic viscosity. A low Reynolds number results in laminar flow with dominant viscous forces. In our case, we have a high Reynolds number (on the order of 10^5 or 10^6), where inertial forces dominate and turbulent flow occurs resulting in vortices. Literature suggests that at Re above 5×10^4, load coefficients are independent of Re for parabolic solar collectors, and the primary contribution to the C_d is due to the pressure force.

Tests were run for the cases of maximum velocity on open SO8 of 55 $kmhr^{-1}$ (15.28 ms^{-1}) as well as 150 $kmhr^{-1}$ (41.67 ms^{-1}). Table summarizes the coefficients at various opening angles of the SO8.

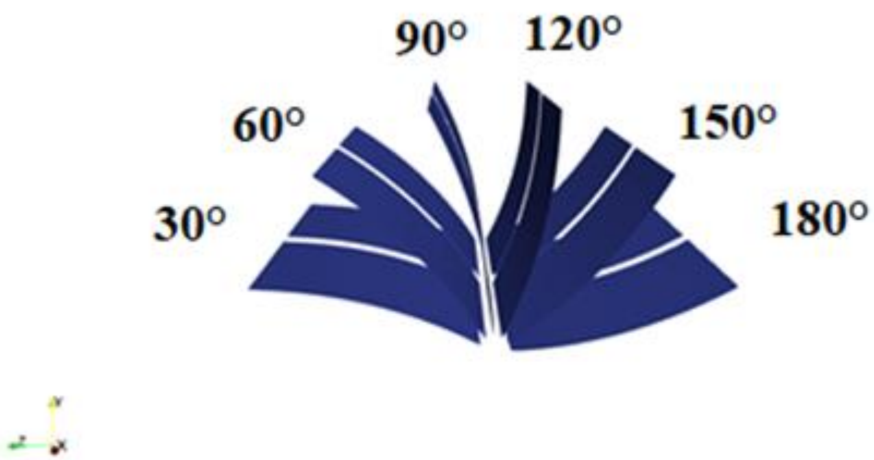

Fig. 6: Various opening and closing angles of the SO8. Note that the SO8 does not rotate about the base but rather around the torque tube. The image is made as such for clarity

Angle (°)	Velocity (ms^{-1})	C_d	C_l	C_m
30	41.67	0.31	1.32	-0.19
30	15..28	0.31	1.31	-0.20
60	15.28	1.12	1.13	-0.13
90	15.28	1.64	0.43	-0.10
105	15.28	1.75	-0.022	-0.063
120	15.28	1.70	-0.51	-0.0046
150*	15.28	1.35	-1.46	0.071
180	15.28	0.30	-1.37	0.097
210	15.28	0.29	0.71	-0.15

*note: values for 150 are not well converged

Table 2: C_d, C_l, and C_m values for maximum velocity given SO8 position

We see that, as expected, the drag coefficient increases as the exposed area surface perpendicular to the wind direction increases. The maximum value of C_d is at 105°, representing a fully open position, while the minimum is at values of fully open and closed, such as 30°, 180°, and 210°. We also see the expected trend for C_l, where the lift is in the upwards direction and greatest for fully closed angles before decreasing. As the mirror bends backwards, the C_l becomes negative, indicating it is forced towards the ground. Interestingly, the downward lift value is highest at an angle of 150° and decreases thereafter. The C_m remains relatively small in all cases and is negative, indicating that the SO8 is being pushed back into closed position, except for 150° and 180°, where it is being forced into a more open position. Though the moment is greatest along the x-axis, from angles between 60° and 150°, the moment in the y-axis and z-axis are not completely negligible, meaning that the SO8 may be shaking from side to side, and, to a lesser extent, up and down slightly.

These results can also be compared to previous studies such as those as shown in Table 3. Though these studies can be used for general comparison, it should be noted that due to differences particularly in the models and parabolic troughs used, the results cannot be perfectly compared. However, they do provide a good benchmark for the reliability of these results until actual wind tunnel or experimental data is available.

Study	Model	Shape	Theoretical C_d	Experimental C_d	Notes
Torrecilla et al.	LES	Shallow trough, no gap	0.8	0.8	Re 8.5×10^4, ABL, 19.44 ms^{-1} at 10 m
Dundage et al.	SST k-ω	Shallow trough, no gap	1.72	--	18 ms^{-1}
Hachicha et al.	LES	Eurotrough	~2.2	~1.7	Re 1×10^6, 3 ms^{-1}
Paetzold et al.	SAS-SST	Shallow to deep troughs	Agreement with experimental data	~1.7 (deep) – 2.1 (shallow)	Re 3.2×10^6, 1 ms^{-1}

Table 3: Previous similar studies comparing C_d and C_l values at reference angle 90°

4.2.1 Wind Mast Probe Locations

Using the results of the above test at various opening and closing angles, various probe locations were tested using probesDicts to find a position that, regardless of opening position, would accurately estimate or slightly overestimate wind speed for safely shutting the SO8. The probesDict file is located in the system directory. The base probe location is assumed to be 1.2 m up and 0.25 m back from the torque tube. However, this location underestimates the wind speeds when the SO8 is at 90°

(underestimated by ~3 ms^{-1}) and 150° (~1 ms^{-1}). The angles that causes the greatest underestimation in windspeed are 150° and 180°. Interestingly, 180° is underestimated regardless of probe location. The ideal location would be 1.6 m up and 0.4 m behind the torque tube, where the wind speed is not underestimated except at 180°. However, if we want to include 180°, then we should include a safety factor of 1 ms^{-1}, in which case a wind mast located 1.3 m up and 0.25 m behind the torque tube should work well. Otherwise, other wind mast positions should have some wind safety factors applied depending on severity of underestimation.

4.3. Transient Simulations

The Strouhal number (St) is a dimensionless parameter that measures the ratio of the characteristic length to the distance travelled during an oscillation period.

$$St = \frac{fL}{U} \tag{10}$$

where f is the frequency of vortex shedding, L the bluff body characteristic length, and U the velocity. The frequency is found by finding the oscillation period of C_l or double the period of C_d.

Representative Length	Characteristic Length (m)	Re number	St number	Vortex shedding frequency (Hz)	Sampling time (s)
Mirror Thickness	0.004	4×10^3	0.213	812.28	0.00123
Gap Width	0.26	2.7×10^5	0.197	11.59	0.0862
Mirror Height	1.7	1.8×10^6	0.235	2.12	0.472

Table 4: Re and St numbers for various characteristic lengths and their corresponding vortex shedding frequencies

Though preliminary tests have been set up for transient cases, results are yet to be determined.

5. Conclusions and Future Outlook

In this report, it has been determined that a mesh consisting of a refinement sphere and band running along the length of the SO8 is sufficient in capturing the details of the flow and providing reasonable results for the C_d coefficient comparable to literature while minimizing the number of elements. A base case (baseCaseSO8) has been created that can easily be manipulated for various scenarios of extreme cases as well as an ABL case. We have determined values for C_d, C_l, and C_m at various opening and closing angles of the SO8 and verified that these values do follow expected trends. Additionally, we found locations for the wind mast probe that may be most suitable. Finally, the beginnings of a transient case have been set up, which would provide interesting results in vorticity as well as vortex shedding frequencies of the SO8.

In the future, depending on the accuracy of coefficients desired, a meshing program other than snappyHexMesh might be used for greater meshing control, as well as using a k-ω SST model and higher order schemes. Additionally, future experiments could include more environmental factors, including the effects of wind fences, multiple SOs in rows, and within different environments, such as on urban and residential rooftops.