

On the efficiency of bird-inspired dual-wing generator

Felix Yanwei Wang

Department of Physics, Middlebury College, Middlebury, Vermont 05753

Project report for PHYS0704

Abstract

Bird flight, with its simplicity, elegance and efficiency, has garnered much interest in scientific and engineering projects. In 2014, Festo, a German robotics company inspired by how efficient a bird wing can produce lift, built a dual-wing generator (DWG) that mimics the movement of wings in flight to extract wind energy and produce electricity. It claims its prototype achieves higher efficiency than current wind turbines at low wind speeds. My project uses COMSOL's computational fluid dynamics module (CFD), a finite-element solver, to model several different configurations of the dual-wing system immersed in the air. The project evaluates the efficiency of the system to extract wind energy by calculating the lift on the airfoils, and verifies the DWG's higher energy output than traditional wind turbines from a quasi-static perspective.

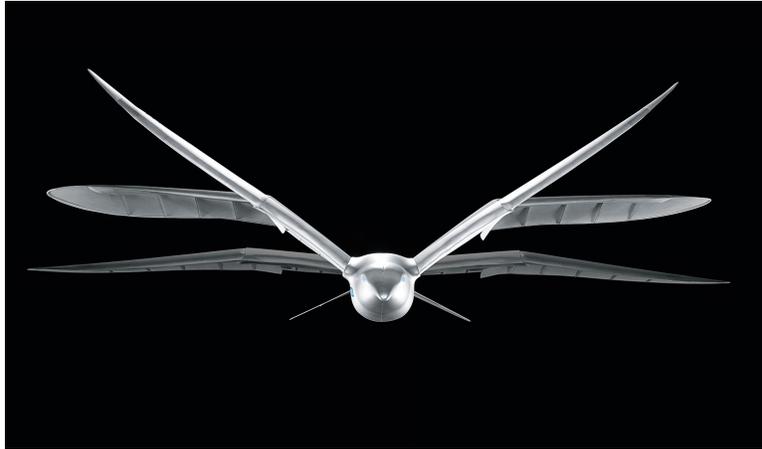
Signatures

Advisor: _____

2nd Reader: _____

Date accepted: _____

To the beauty of flight



Contents

| | | |
|-------------|---|----|
| I. | Introduction: What is a bird-inspired dual-wing generator (DWG)? | |
| 1.1 | An alternative to wind turbines: dual-wing generator | 5 |
| 1.2 | Outline and methodology | 8 |
| II. | Physics: How does an airfoil generate lift? | |
| 2.1 | Definition of a fluid and flow | 9 |
| 2.2 | Definition of an airfoil | 12 |
| 2.3 | Theories of lift generation | 13 |
| 2.4 | The conservation of mass and continuity equation | 18 |
| 2.5 | The conservation of energy and Bernoulli's equation | 19 |
| 2.6 | The conservation of momentum, Euler equations and Navier-Stokes equations | 21 |
| 2.7 | The calculation of lift | 24 |
| 2.8 | COMSOL and finite-element method | 25 |
| III. | Modeling: How to simulate fluid flows in COMSOL CFD? | |
| 3.1 | Model and parameter specification for dual-wing generator | 28 |
| 3.2 | Geometry of DWG | 30 |
| 3.3 | Boundary conditions | 33 |
| 3.4 | Meshing of DWG | 35 |
| 3.5 | Validity of the model | 36 |
| IV. | Results: Is DWG more efficient than wind turbines? | |
| 4.1 | DWG lift calculation in a static simulation..... | 41 |
| 4.2 | DWG lift results and energy calculation | 43 |
| 4.3 | Concluding remarks | 51 |

I. Introduction: What is a bird-inspired dual-wing generator (DWG)?

1.1 An alternative to wind turbines: dual-wing generator

In 2014, Festo, a German robotics company built a dual-wing generator (DWG) that mimics the movement of bird wings in flight to extract wind energy and produce electricity. The system is based on reversing the natural wing-beating principle: Whilst birds flap their wings to generate power to move forward, a stationary system like the DWG can extract kinetic energy from the wind [1]. Figure 1 shows the setup of the DWG, which uses airfoils to approximate bird wings. The wind power creates a linear lifting movement of the airfoils, which is converted into a rotary movement at the upper and lower ends of the DWG, so that both airfoils can return to their initial position to complete a cycle. Having a cycle, the DWG can be connected to an electric generator to convert the kinetic energy of the airfoils into electricity.

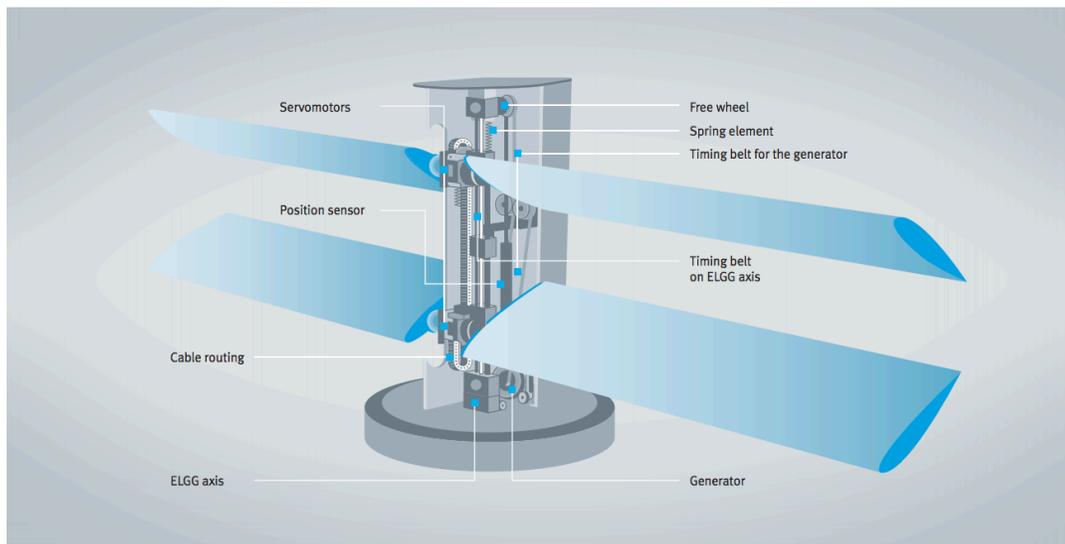


Fig. 1. The setup of Festo's dual-wing generator [1]. Notice the wings in this setup can only have a small displacement in the vertical direction compared to their length.

The essence of how the DWG extracts wind power is similar to that of a wind turbine. The DWG utilizes its airfoils to produce lift while a wind turbine uses its turbine blades. Both machines rely on airflows to drive the motion of their movable parts, through which the wind energy is transferred to the kinetic energy of the airfoils or the

turbine blades. A turbine blade also has a similar shape to an airfoil, and they share a similar lift generation mechanism. In fact, one can even regard the rotary motion of the turbine blades as a linear motion of successive parallel turbine blades if one observes a turbine from its side and watches the blades go by.

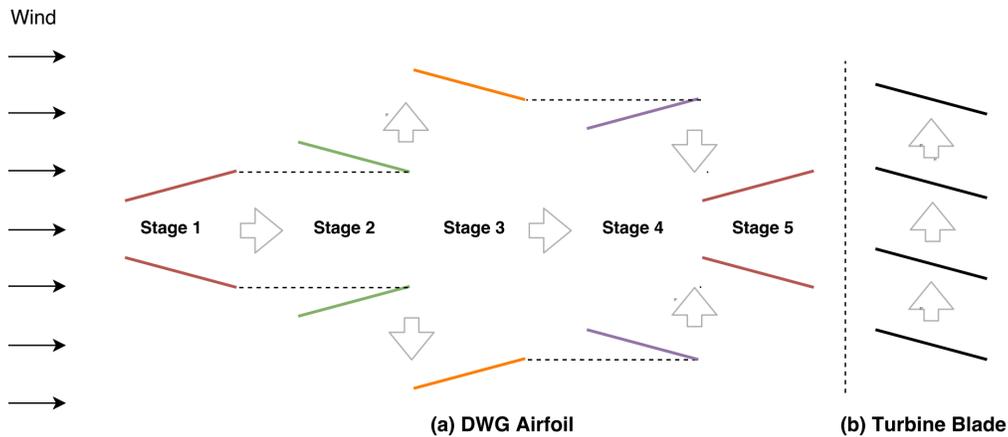


Fig. 2. Illustrations of how airfoils and turbines blades, represented by line segments, move in wind. Part (a) shows a cycle of the DWG’s energy generation process where the dual-wing configuration moves from stage 1 to stage 5 to complete a cycle. Airfoils in different colors are in different stages, and the dotted line shows the part of the airfoil that maintains its position in a transition. Part (b) shows the movement of turbine blades observed from side. A turbine usually has three blades, but four blades are shown here because the first blade follows the third blade in a rotation after completing one cycle.

Figure 2 shows the movement of airfoils and wind turbine blades, which are represented by line segments. If a line segment is tilted upwards, the lift generated will push up the line segment as indicated from “stage 2” to “stage 3”, and vice versa as indicated from “stage 4” to “stage 5”. In both processes, wind does work and transfers energy to the machines. The wind turbine blades move in a similar fashion as do the airfoils from “stage 2” to “stage 3”, despite that the turbine blades move upwards continuously if one views from the side of a turbine, as shown in Fig. 2(b). Notice in Fig. 2(a), the DWG does need to do work to reverse the motion and tilting angle of airfoils during transitions from “stage 1” to “stage 2” and from “stage 3” to “stage 4”. However, one can argue that the work used to tilt the downward facing airfoil in “stage 1” first to a

horizontal position is compensated by the energy added in the next transition, driven by the wind, from that horizontal position to the upward facing position in “stage 2”, thus making the entire transition from “stage 1” to “stage 2” a process of zero net energy change approximately. To simplify the calculation in my analysis part, I ignore the “reversing angle” process and only focus on the linear lifting movement of the airfoils from “stage 2” to “stage 3” and from “stage 4” to “stage 5”.

Despite their similarities, the wind turbine blades’ motion differs from that of the airfoils in the DWG in one crucial respect: The separation between the turbine blades does not change, whereas the separation between the two airfoils does change. This is why Festo designs a dual-wing generator rather than a single wing generator. Apart from that, the opposing tandem wings neutralize the bearing torques from the upwards and downwards motion, having two airfoils moving towards each other creates a suction effect when they get close [1]. Similar to the situation where two boats get pulled together if they move in parallel and close to each other, the suction effect introduces additional lift and increases the energy output. This is one of the reasons why Festo claims its DWG achieves higher efficiency than current wind turbines especially at low wind speeds, as shown in Fig. 3. My project is thereby motivated to investigate whether the suction effect contributes to a higher energy output, and results in an energy-output-to-wind-speed relationship for the DWG similar to the one shown in Fig. 3. The specific process is outlined in section 1.2.

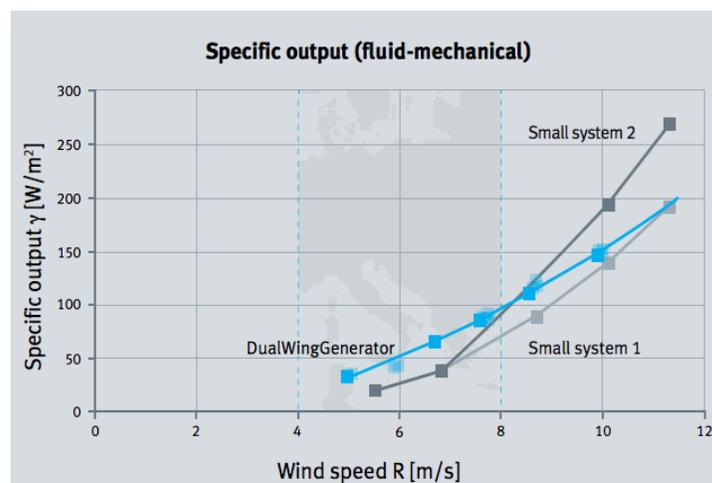


Fig. 3. DWG’s specific output compared to small systems, which are conventional wind turbines of similar size [1].

1.2 Outline and methodology

In this project, I first motivate the connection between bird wings and the DWG as well as the comparison between the DWG and wind turbines in chapter I. In chapter II, I give an overview of fluid dynamics, establishing essential concepts to my analysis. The goal is to understand how the airfoil generates lift, why I need to solve Navier-Stokes equations, and how COMSOL solves those equations. In chapter III, I focus on the detailed implementation of my DWG simulation in COMSOL, specifically on four parts: model parameters, model geometry, model boundary conditions and model meshing. The very last section of this chapter discusses the validity of my previous assumptions and justifies that the model I eventually use is suitable for my DWG simulation. Lastly, in chapter IV, I present my results from simulation and discuss in terms of energy output whether those results support Festo's claim of DWG as a superior alternative to traditional wind turbines at low wind speeds.

The methodology I use in this project is an iterative process: Simplify my model and ask if the simplified model works? → If yes: add back some complication to make the model more realistic; if no: simplify the model more → Repeat the process. This is the golden rule that leads to many of my unnatural assumptions originally in an attempt to find a working model. When I begin to understand better the physics as well as modeling in COMSOL through the simplified models, I gradually update them to match our reality.

II. Physics: How does an airfoil generate lift?

2.1 Definition of a fluid and flow

A **fluid** is a continuous medium, a *continuum*, which deforms *continuously* in the presence of a shear stress, regardless of how small the shear stress may be [2]. A **shear force** F_s is the force component tangent to a surface, and the **shear stress** τ is defined over an associated area A , as $\frac{F_s}{A}$ [2]. The definition of a fluid is underpinned by its characteristic of a continuous deformation. Unlike a solid, for which momentum transfer occurs at the point of collision and completes within the instant of contact, a moving fluid deforms when it meets any resistance and thus the momentum transfer is more complicated. On the one hand, the deformation is spatially continuous so that neighboring points in the fluid change speed and pressure gradually. On the other hand, the deformation is temporally continuous in the sense that it does not complete instantly, and a disturbance propagates at a finite speed, the speed of sound in forms of pressure waves.

There are two important properties of a fluid: compressibility and viscosity. **Compressibility** describes the “springiness” of the fluid, while **viscosity** describes the “stickiness” of the fluid [3]. These properties are usually discussed in the context of a **flow**, which is a moving fluid. In terms of compressibility, a flow is categorized into either a **compressible flow** or an **incompressible flow**. The difference lies in whether the density of the fluid remains constant under external forces [3]. For an incompressible flow, the fluid density does not vary with pressure. Technically, all fluid can be compressed as long as the external force is large enough. To determine when a flow can be regarded incompressible, we use **Mach number** M , the ratio of flow velocity to the speed of sound. If $M < 0.3$, the fluid speed is small enough compared to the speed of disturbance propagation, which is the speed of sound. As a result, the density of the fluid does not vary much and the flow can be approximated as an incompressible flow [4].

In terms of viscosity, a fluid is either viscous or inviscid. An **inviscid flow** is a flow of an ideal fluid of no viscosity, but in real life all fluids are viscous and form a **viscous flow**. **Viscosity** measures how gradual the spatial change in velocity is under a unit shear stress, and is defined as the proportionality factor μ in the equation

$$\tau = \mu \frac{du}{dy}, \quad (1)$$

where τ is the shear stress at a point, and $\frac{du}{dy}$ is the velocity gradient component perpendicular to the shear force at that point. Equation (1) explains the velocity profile in Fig. 4, where the lower plate is at rest and the upper plate under a shear stress τ moves at velocity V . The “**no slip condition**” at both plates commands the fluid in direct contact with the plate surface to acquire the velocity of the plate, thus giving a velocity difference V in the direction perpendicular to the flow. If the fluid is divided into infinitely many thin layers parallel to the flow direction, where each layer has a uniform velocity, then each layer will move faster than the layer directly below it and slower than the layer directly above it. Because the fluid is viscous, the viscous force arises when neighboring layers have different velocities, and causes shear stress that slows the faster layer and speeds up the slower layer. A simple model for the fluid between two parallel plates to bridge the velocity difference, as shown in Fig. 4, is a linear velocity profile: $\frac{du}{dy} = \frac{\tau_{upperplate}}{\mu} = \frac{V}{h}$ (constant), where $\tau_{upperplate}$ is the shear stress at the upper plate, and h is the distance between the two plates. The velocity profile for an unbounded flow over a single surface, as shown in Fig. 5 where there is no second plate, however, is not a linear one. Therefore, Eq. (1) is written in differential form to generalize the relationship between the shear stress and the velocity gradient at any point in the fluid.

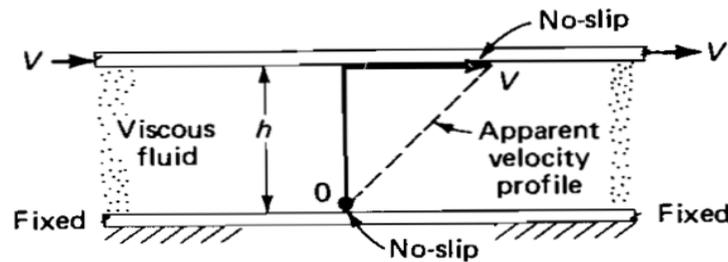


Fig. 4. A viscous fluid sheared between two plates of separation h . The lower plate is fixed while the upper plate moves horizontally to the right at speed V . No slip condition specifies that fluid in immediate contact with the plate shares the velocity of the plate. This is Fig. 1-15 from *Viscous Fluid Flow* (White 1991) [4].

Although a fluid slows down at the surface of a plate, in other words a boundary, under the “no slip condition”, it is intuitive to think that the fluid should maintain its free-stream velocity far from the boundary. The **free-stream velocity** is the velocity of a fluid in a region far upstream of an object where the velocity is uniform in space [5]. In contrast, the aforementioned varying velocity profile describes a region close to the boundary, called a **boundary layer**, which exists between the 0-velocity surface and the free-stream velocity area. Figure 5 shows the transition of a laminar boundary layer into a turbulent boundary layer along a surface. A **laminar flow** characterizes a flow regime of parallel layers without any disruption between the layers, while a **turbulent flow** characterizes a flow regime of unsteady swirling flows [6]. Notice in Fig. 5 a gradual transition from the parallel laminar layers into the swirling turbulent layers within the boundary. If the schematic shows more of the turbulent region, we would expect all the parallel layers to turn into swirls and even a separation of flow from the surface. Also, the laminar boundary layer continues a little before transitioning into the turbulent boundary layer, which implies the transition is dependent on the length of the boundary. Specifically, it is called the characteristic length L in the formula of Reynolds number,

$$Re = \frac{\rho V L}{\mu}, \quad (2)$$

where other variables include ρ , the density of the fluid, V , the free-stream velocity of the fluid, and μ , the viscosity of the fluid. **Reynolds number** expresses the ratio of a fluid parcel’s inertial forces to viscous forces. A **fluid parcel** is an infinitesimal amount of fluid, which has a constant mass yet a deformable shape. The fluid parcel’s inertial force is its resistance to change in motion. The fluid parcel’s viscous force is the net shear force caused by its surrounding fluid to change its motion [7]. If Reynolds number is small, the viscous force is large compared to the inertial force and parallel layers of fluid are maintained, leading to a laminar flow. On the contrary, if Reynolds number is large, the viscous force is not sufficient enough to bound flowing fluid parcels into orderly layers and thus the fluid becomes turbulent.

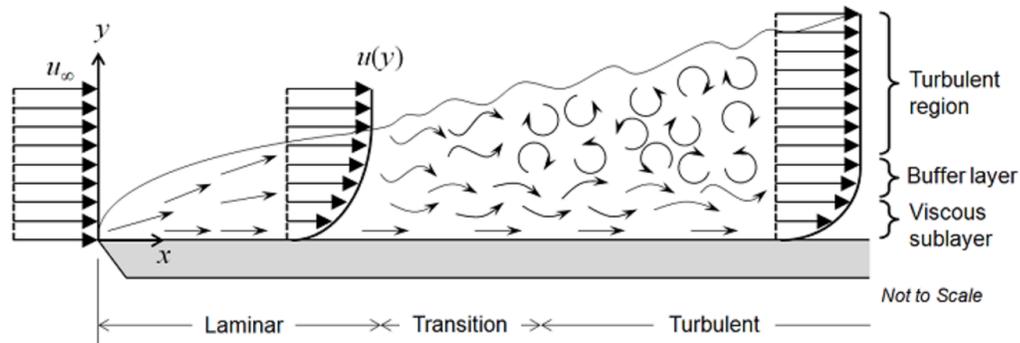


Fig. 5. Boundary layer development from a laminar region into a turbulent region. The uniform arrows to the left of the y -axis represent a free-stream velocity u_∞ , and the arrows of varying length to the right show a velocity profile $u(y)$. As the boundary layer develops, it becomes thicker and the original ordered straight arrows evolve into swirled and curved arrows, indicating turbulence [6].

2.2 Definition of an airfoil

An **airfoil** is a physical body that generates more lift than drag when positioned at a proper angle to airflow [5]. A fluid moving past any physical body exerts a force on the surface of the body, and **lift** is the force component perpendicular to the flow direction, while **drag** is the parallel force component [5]. Figure 6 gives the 2D cross-section of a sample airfoil. In Fig. 6, the leftmost tip of the airfoil is called the **leading edge** while the rightmost tip is called the **trailing edge**. The leading edge is less pointed than the trailing edge. Airflow meets an airfoil at its leading edge first and leaves at its trailing edge. The line that connects the two edges is the **chord**, which is the longest line that fits into the 2D cross-section of the airfoil. The **mean camber line** divides the airfoil into the upper camber and the lower camber. Each point on the mean camber line is equally distant from the upper surface and lower surface. Although Fig. 6 shows an almost straight camber line, it is normally curved. The deviation of the mean camber line from the chord determines the shape of the airfoil, and the overlap of these two lines gives a symmetrically shaped airfoil. Lastly, the **angle of attack** is the angle between the direction of the airflow and the chord, and it determines how tilted the airfoil is positioned. Although the airfoil in Fig. 6 has more curvature to the upper surface than the

lower surface, this is not necessary for lift generation according to the definition of the airfoil. In fact, an airfoil can be symmetrical with respect to the chord, or even just a sheet of any material whose cross-section is a line segment. The lack of specificity of the shape explains why airplane wings can be flown upside down. The more curvature to the upper surface only improves the efficiency of lift generation but is not essential [9].

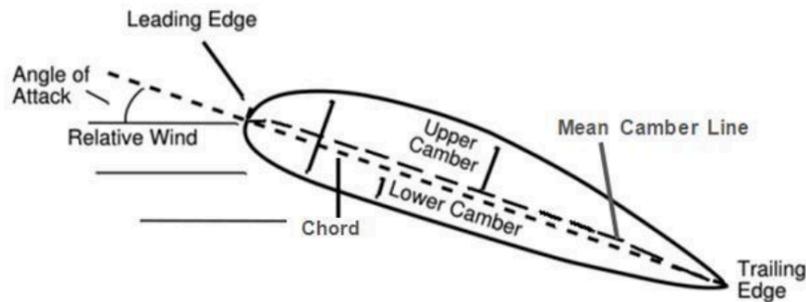


Fig. 6. Cross-section of an asymmetric airfoil [8].

2.3 Theories of lift generation

As we establish that the shape of the airfoil is not essential to lift generation, it is tempting to ask what explains lift generation. There are many seemingly reasonable yet incorrect theories of lift. One of the *incorrect* explanations is the “**skipping stone**” **theory** [10]. Figure 7 shows an inflow of air coming from below the airfoil and “bumps” into the airfoil’s lower surface, during which the air creates a lift. This theory is based on the Newton’s third law, and pictures air molecules as skipping stones that bounce back from a surface. The theory is partially right because air molecules do get deflected downwards by the airfoil, but it fails at recognizing the airflow as a fluid rather than a solid. As a fluid, the airflow *deforms* as it is obstructed by the lower surface of the airfoil, and the resulting shear stress propagates in all directions to other regions, including the region above the airfoil. The “skipping stone theory” misses the lift contribution from the upper surface of the airfoil, which in fact accounts for the majority of the lift generation of an airfoil. Thus we learn from this incorrect theory that one has to consider wind as a continuum in which the airfoil is immersed, and thus the lift calculation should derive from applying the Newton’s third law to the entire surface of the airfoil.

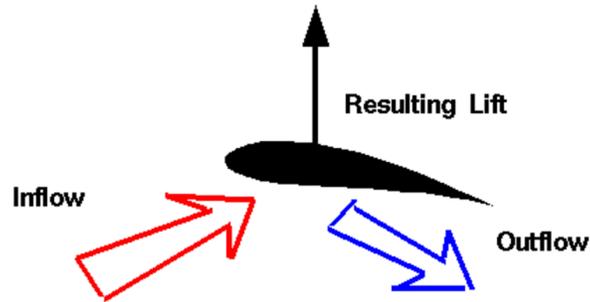


Fig. 7. “Skipping stone” theory of lift generation based on Newton’s third law [10].

A second incorrect theory that gives insight into lift generation is the “**longer path**” theory [11]. To explain the theory, let me first introduce the concept of streamlines. A **streamline** is an imaginary line drawn in the field of flow such that the velocity vector at any point on the line is always tangential to the line [5]. The dots of the same color, except for the black ones, in Fig. 8 form streamlines and separate the domain into parallel layers of fluid. These layers deform and vary in their widths as they flow past the airfoil. Because no point in the field can have more than one velocity, it follows that no two streamlines can intersect each other, except for at a point called stagnation point. A **stagnation point** is a point in the flow field where the local velocity of the fluid is zero. There are two stagnation points in Fig. 8, where the streamlines first separate right below the leading edge and where the streamlines meet again at the trailing edge. A clearer view of the two stagnation points can be found in Fig. 21. The fluid traveling around the airfoil has two different paths from the first stagnation point to the second stagnation point: the upper surface and the lower surface of the airfoil. Because the first stagnation point is below the leading edge and that the upper surface has more curvature than the lower surface, the upper path is longer than the lower path. The “longer path” theory claims that the fluid parcels in the upper path must travel faster than do the fluid parcels in the lower path in order for them to meet again at the second stagnation point at the trailing edge. As a result, a pressure difference is created according to the Bernoulli’s equation, which will be explained in detail in section 2.5.

The “longer path” theory is partially correct in that there is a velocity difference and consequently a pressure difference between the upper and lower surface, which is essentially how lift is generated. But this theory fails to explain how the velocity

difference arises, because the fluid parcels that separate at the first stagnation point never meet again at the second stagnation point. Figure 8 shows several sets of black dots forming vertical lines. Each vertical line formed by the black dots is a **timeline**, where the black dots of the same timeline have traveled for the same period of time [12]. Every two neighboring timelines share a same time difference T . In Fig. 8, the first set of black dots forms a straight timeline far to the left of the airfoil as the fluid parcels represented by the dots have so far traveled at the same velocity and for a same time span. As the fluid parcels acquire different velocities approaching the airfoil, the black dots representing them begin to travel past different distances, thus deforming the straight timeline into a curve as the timeline moves towards the airfoil. When the timeline meets the leading edge of the airfoil, it splits into two parts, with the faster moving part moving ahead, or in this case to the right, of the other part. However, contrary to what the “longer path” theory predicts, the two separated parts of the same time line never rejoin at the trailing edge of the airfoil. When the fluid parcel traveling along the upper surface reaches the trailing edge, the fluid parcel traveling along the lower surface from the same timeline is only half way through and is almost left behind by a T , as shown in Fig. 8.

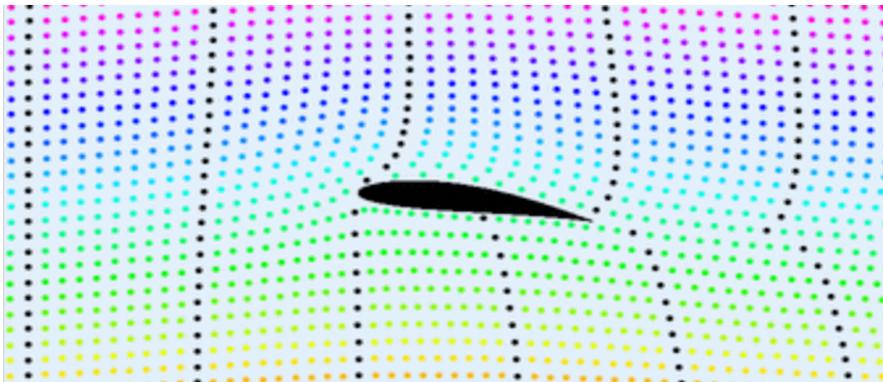


Fig. 8. A flow around an airfoil. Each dot represents a fluid parcel. The dots of the same color constitute a streamline, while the dots of different color move with the flow at different velocities. The black dots are from different streamlines, and yet they join the same timeline, marking an equal traveling time for these dots [13].

A correct explanation of lift generation is based on Newton’s third law. As the airflow is deflected downwards by the airfoil in Fig. 8, it experiences a force from the

airfoil pointing downwards. According to Newton’s third law, the airflow also exerts a force to the airfoil in the opposite direction, which is essentially the lift. In contrast to the incorrect “skipping stone” theory, we apply the Newton’s third law to both the upper and lower surface of the airfoil, as the airflows moving along both surfaces are deflected downwards. In the momentum term, the airfoil transfers a downward momentum to the airflow and thus acquires an upward momentum and moves upward [14].

The lift can also be seen as from the pressure difference between the upper and lower surfaces of the airfoil. Because the pressure far above and below the airfoil should have the same magnitude, there must be a pressure gradient around the airfoil, so that a difference between the pressure on the upper and lower surface is achieved. Such a pressure gradient is achieved through the turning of the streamlines.

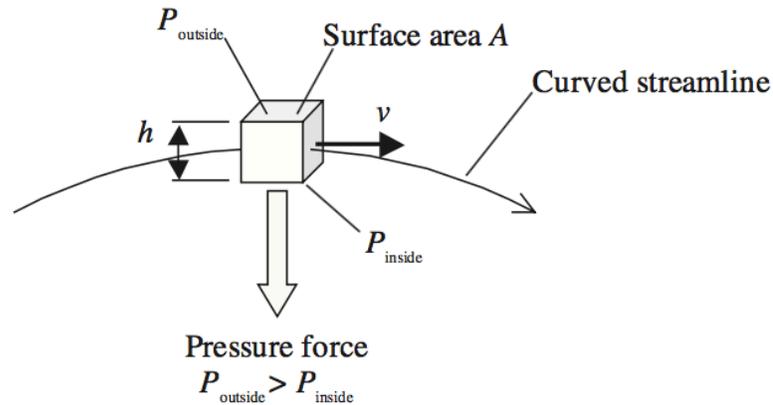


Fig. 9. Pressure gradient across curved streamlines. A fluid parcel of cubic shape has edge h , velocity v along the streamline, and a pressure $P_{outside}$ on the upper surface whose normal direction points away from the center of the streamline curvature, and a pressure P_{inside} on the lower surface whose normal direction points to the center of streamline curvature [15].

In Fig. 9 a fluid parcel is traveling along a streamline curved downwards. As the fluid parcel acquires a downward velocity component, it must experience acceleration and thus a force pointing downward. This force derives from the pressure difference between the fluid parcel’s upper and lower surface, and the direction of the force suggests

the upper surface experiences a higher pressure. Let the pressure on the upper surface “outside” the curve be $P_{outside} = p + dp$ and the pressure on the lower surface “inside” the curve be $P_{inside} = p$. Thus $dp = h (dp / dy)$, where dy is in the direction normal to the streamline and points away from the center of curvature [15]. The force due to the pressure difference is $F = A \cdot dp$, where A is the surface area of the cubic fluid parcel. This force must equal the centripetal force $F = mv^2 / R$, where v is the flow velocity and R is the local radius of curvature. The mass of the fluid parcel is $m = \rho Ah$. Combining all of the above equations yields

$$F = A \cdot dp = Ah \frac{dp}{dy} = \rho Ah \frac{v^2}{R},$$

which is simplified to

$$\frac{dp}{dy} = \rho \frac{v^2}{R}, \quad (3)$$

which is the pressure gradient *across* streamlines. Equation (3) shows that there will be a pressure difference if the streamline is curved. If $R \rightarrow \infty$, then $\frac{dp}{dy} \rightarrow 0$. So there is no pressure gradient across straight streamlines [15]. In Fig. 8, both the streamlines above the airfoil and the streamlines below the airfoil are curved downwards like the streamline shown in Fig. 9. Therefore, there is a continuous pressure dropping from above the airfoil to below the airfoil, creating a net force on the airfoil point upwards. Thus for an airfoil to generate lift, it must turn the streamlines in the direction opposite to the desired direction of the lift. Notice the lift is not necessarily an upward pointing force and it can point downwards when the airfoil has negative angle of attack and deflects the flow upwards.

Equation (3) also gives us insight into the different portion of lift contributed by the upper and lower surface of the airfoil. As shown in Fig. 8, the streamlines above the airfoil have larger curvature than the streamlines below the airfoil, which is $R_{upper} > R_{lower}$. According to Eq. (3), this means a larger pressure difference above the airfoil than below the airfoil, indicating a larger lift contribution from the upper surface than from the lower surface.

2.4 The conservation of mass and continuity equation

Apart from the pressure information, the distribution of the streamlines also gives us information about the velocity change of fluid parcels in Fig. 8. A careful reader will notice the streamlines above the airfoil are more densely packed than they are below the airfoil, which means fluid parcels above the airfoil will flow into a narrower streamtube than will the fluid parcels below the airfoil as they approach the airfoil. A **streamtube** is a tubular region of fluid surrounded by streamlines [16]. At the boundary of the streamtube formed by streamlines, there should be no flux crossing. A fluid parcel crossing the boundary indicates an intersection of two streamlines of different direction, which is forbidden by definition. Therefore, fluid parcels in a streamtube in Fig. 10 can only enter in the flow direction from its inlet, which has size A_1 , and exits in the flow direction from its outlet, which has size A_2 . **The conservation of mass** states that mass is neither created nor destroyed [17]. So the amount of mass inside the streamtube remains constant, which means the amount of mass that flows in through area A_1 must equal that flows out through area A_2 in every instant. For an infinitesimal straight steady streamtube we have

$$\rho_1 A_1 v_1 = \rho_2 A_2 v_2 , \quad (4)$$

which for an incompressible flow simplifies to $A_1 v_1 = A_2 v_2$, as the fluid density does not change. Equation (4) relates the velocity of the flow to the geometry of streamlines based on the conservation of mass. In light of this, the higher density of streamlines at the upper surface of the airfoil in Fig. 8 than at the lower surface implies a higher velocity at the upper surface than at the lower surface.

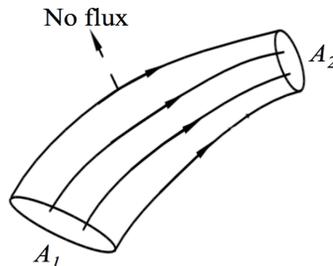


Fig. 10. Streamtube formed by streamlines. There is no flux across the boundary of the streamtube, and mass flows in from inlet of area A_1 and flows out of outlet of area A_2 .

Simple and intuitive as Eq. (4) is, it is only suitable for a steady flow. A **steady flow** is one where the flow conditions such as velocity and pressure do not change with time [2]. To amend for the time-dependence, we use a general differential form of the conservation of mass equation, or the **continuity equation**,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \quad (5)$$

where the term $\frac{\partial \rho}{\partial t}$ accounts for the fluid density dependence on time, the divergence term accounts for the net mass flow out of the boundaries, and the “0” term means there is no sink or source of mass and accounts for mass conservation. Notice the velocity field \mathbf{u} is a vector containing u, v, w , each of which is a one-dimensional velocity.

2.5 The conservation of energy and Bernoulli’s equation

In section 2.2 we confirm a pressure difference between the two surfaces of the airfoil based on the curvature of the streamlines, while in section 2.3 we confirm a velocity difference between the two surfaces of the airfoil based on the density of the streamlines. One may naturally ask if there is a relationship between the pressure and velocity. The answer is yes, and in fact, I have alluded to the Bernoulli’s equation in section 2.3 to refer to the relationship. In this section I derive the pressure-velocity relationship from the conservation of energy. As I have looked at the pressure gradient across streamlines, I start with looking at the pressure gradient along streamlines this time.

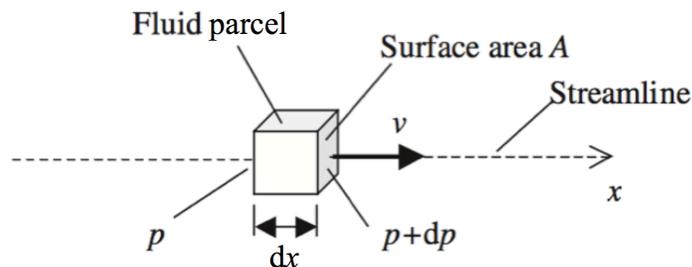


Fig. 11. Pressure gradient along a streamline. Pressure drops accelerate the fluid parcel. In the energy term, the enthalpy of the fluid drops while the kinetic energy of the fluid increases. The energy is conserved [15].

In Fig. 11, a fluid parcel at velocity v and surface area A is accelerated by a pressure difference in the x -direction. The pressure behind it is p and the pressure in front of it is $p + dp$. According to Newton's second law [14],

$$F = ma = m \frac{dv}{dt}, \quad (6)$$

where $m = \rho A dx$. Alternatively, the pressure difference gives $F = -dp A$, which points to the $-x$ direction. Combining both expressions of the force gives us $-dp A = \rho A dx \frac{dv}{dt}$ which can be simplified to an relationship between the pressure gradient in the x -direction and the acceleration of the fluid:

$$-\frac{dp}{dx} = \rho \frac{dv}{dt}. \quad (7)$$

Rearranging Eq. (7) into $dp = -\rho v dv$, where we have substituted $\frac{dx}{dt}$ with v . Now integrating between any two points along the streamline gives us

$$\int_1^2 dp = -\int_1^2 \rho v dv$$

$$p_2 - p_1 = -\rho \left(\frac{v_2^2}{2} - \frac{v_1^2}{2} \right),$$

which can be rearranged as

$$p_1 + \rho \frac{v_1^2}{2} = p_2 + \rho \frac{v_2^2}{2}, \quad (8)$$

which is a simplified form of **Bernoulli's equation** without considering gravity [15].

Notice Eq. (8) comes from integrating over two arbitrary points along the streamline, so it applies to any two points along the streamline. In my derivation I integrate over an infinitesimal dx , so on the macro scale the streamline can be a curve. There are, nevertheless, many limitations to Bernoulli's equation. First, I assume the pressure difference to be the only source of force in the direction of the streamline, excluding any external source of force, such as gravity, or the viscous force of the fluid. So the flow has to be an inviscid flow for Eq. (8) to work. Second, I do not include any time-dependent terms in my derivation, so the flow has to be steady as well. Third, I assume constant fluid density at point 1 and point 2, so the flow has to be incompressible and at a low velocity. There are some other restrictions, but so far I have shown Bernoulli's equation can only solve an inviscid, steady, incompressible flow.

In addition to my Newton's second law's derivation of the Bernoulli's equation, one can also derive it from the first law of thermodynamics: the conservation of energy,

which is nevertheless omitted here [18]. If the internal energy of the fluid does not change, the conservation of term $p + \rho \frac{v^2}{2}$ in Eq. (8) indicates that the sum of enthalpy, defined as the system internal energy plus the product of pressure and volume, and kinetic energy of the fluid conserves [19]. As p decreases and v increases, energy transfers from the fluid's enthalpy into the fluid's kinetic energy, while the total energy of the fluid remains constant.

2.6 The conservation of momentum, Euler equations and Navier-Stokes equations

Since I've discussed the conservation of mass and energy, it will be a little unfair if I leave out linear momentum conservation. The momentum conservation states that the momentum is neither created nor destroyed, and is only changed through force [14]:

$$F \cdot t = m_2 v_2 - m_1 v_1 . \quad (9)$$

Take the time derivative of Eq. (9), we get Eq. (6). So for every instant, the momentum conservation reduces to force balance, which is Newton's second law, or Eq. (6). In fact, when we start with Newton's second law in section 2.5, we are on track to derive the momentum conservation equation, until we integrate Eq. (7) over a path, which then gets us the energy conservation equation. If we do not integrate Eq. (7) and write $\frac{dv}{dt}$ as $\frac{dv}{dx} \cdot \frac{dx}{dt}$, which is essentially $v \frac{dv}{dx}$, we get

$$-\frac{dp}{dx} = \rho v \frac{dv}{dx} , \quad (10)$$

which is the one-dimensional differential form of conservation of momentum for a steady flow [20]. This momentum equation has one benefit over the energy equation, Eq. (8), as it applies to an incompressible flow, because we do not integrate over any path where the fluid density is not uniform. However, Eq. (10) is still not applicable to a viscous flow as I leave out the consideration of viscous force in my derivation from Newton's second law.

A more general form of the momentum conservation equation that solves for an inviscid yet unsteady, compressible flow is included in the **Euler equations**, which is a set of partial differential equations (PDE) that govern fluid motions. Because momentum

is a vector, we need three equations to describe the momentum conservation in addition to our time-dependent mass conservation equation [21]:

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} + \frac{\partial(\rho w)}{\partial z} = 0 \text{ (Continuity Eq.)} \quad (11)$$

$$\frac{\partial(\rho u)}{\partial t} + \frac{\partial(\rho u^2)}{\partial x} + \frac{\partial(\rho uv)}{\partial y} + \frac{\partial(\rho uw)}{\partial z} = -\frac{\partial p}{\partial x} \text{ (X-momentum Eq.)} \quad (12)$$

$$\frac{\partial(\rho v)}{\partial t} + \frac{\partial(\rho uv)}{\partial x} + \frac{\partial(\rho v^2)}{\partial y} + \frac{\partial(\rho vw)}{\partial z} = -\frac{\partial p}{\partial y} \text{ (Y-momentum Eq.)} \quad (13)$$

$$\frac{\partial(\rho w)}{\partial t} + \frac{\partial(\rho uw)}{\partial x} + \frac{\partial(\rho vw)}{\partial y} + \frac{\partial(\rho w^2)}{\partial z} = -\frac{\partial p}{\partial z} \text{ (Z-momentum Eq.)}, \quad (14)$$

where Eq. (11) is the same as Eq. (5) but not in vector form, and Eqs. (12) – (14) are the three-dimensional partial forms of Eq. (10) that solve for unsteady and compressible flows. Specifically, Eqs. (11) – (14) all have the time-dependent $\frac{\partial}{\partial t}$ term to account for unsteady situations, and the density ρ is no longer separable from the spatial partial-derivative operator as it varies in space in a compressible flow. To solve for the viscous flow as well, the Euler momentum equations need to be modified to include viscous forces. The set of PDEs that govern the motion of a viscous, compressible, unsteady flow is called the **Navier-Stokes equations**. The Navier-Stokes momentum equations, which are equivalent to Eqs. (12) – (14) but now including the viscous term and other external forces, in the vector form are

$$\underbrace{\rho \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right)}_1 = \underbrace{-\nabla p}_2 + \underbrace{\nabla \cdot (\mu(\nabla \mathbf{u} + (\nabla \mathbf{u})^T) - \frac{2}{3}\mu(\nabla \cdot \mathbf{u})\mathbf{I})}_3 + \underbrace{\mathbf{F}}_4, \quad (15)$$

where ρ is density, p is pressure, μ is dynamic viscosity, \mathbf{F} is the external body force, \mathbf{I} is the identity matrix, \mathbf{u} is the vector field of flow velocity, and T signifies the transpose [22]. In terms of force balance, term 1 describes the initial forces, term 2 describes the pressure forces, term 3 describes the viscous forces, and term 4 describes the external forces applied to the entire fluid body at the boundaries [22]. Complex as Eq. (15) seems, it is essentially $F = ma$ for fluids. On the one hand, term 1 corresponds to ma , where $\frac{\partial \mathbf{u}}{\partial t}$ is the **local acceleration** of the fluid parcel at a fixed point in space while $\mathbf{u} \cdot \nabla \mathbf{u}$ is the **convective acceleration** of the fluid parcel, which predicts how the flow differs in space at the same instant of time [23]. On the other hand, terms 2, 3 and 4 collectively are the forces that cause the acceleration. Notice in our previous incompressible equations, we do

not have term 3, the viscous force, and we have been ignoring gravity and other external forces, which are represented by term 4.

In addition to the three momentum equations, Navier-Stokes equations also include the continuity equation Eq. (11), an energy equation, which is related to the temperature of the fluid, and a fluid equation of state, which links temperature, pressure and density together. We have six independent equations in total as we solve for six unknowns of the fluid: pressure, temperature, density, and velocity field, which has three components. It is interesting that in section 2.5 the integration of Eq. (7), which is equivalent to the momentum Eq. (8), leads to the energy equation. The fact that both momentum conservation and energy conservation lead to the same Bernoulli's equation shows that the energy conservation equation is redundant in the analysis for a steady incompressible inviscid flow [24]. However, the energy equation in the Navier-Stokes equations, although omitted here, is not redundant as we will not be able to solve for all six unknowns without it.

That most of the six unknowns appear in all six Navier-Stokes equations means these equations are coupled and must be solved simultaneously [21]. Solving a set of partial differential equations usually relies on computer programs to solve numerically with some additional boundary conditions. Although Navier-Stokes equations can in theory solve for any flows, in practice they can only give exact solutions to laminar flows due to the immense computational complexity for turbulent flows. Because the turbulent patterns are usually transient in nature, to model them and find the exact solutions are beyond the computational power of most current computers. Therefore, we can use a *Reynolds-Averaged Navier-Stokes (RANS)* formulation of the Navier-Stokes equations, which uses a time-averaged velocity and pressure field [22]. These time-averaged equations can be then solved relatively quickly.

To further reduce the time complexity in my study, I initially adopt both an incompressible and a steady assumption, so that Eq. (15) and Eq. (5) can be simplified. The incompressible assumption gives $\nabla \cdot \mathbf{u} = 0$, and drops $-\frac{2}{3}\mu(\nabla \cdot \mathbf{u})\mathbf{I}$ from the term 3 in Eq. (15). The time-independent assumption eliminates the $\frac{\partial \mathbf{u}}{\partial t}$ term in Eq. (15) and the

$\frac{\partial \rho}{\partial t}$ term in Eq. (5). I also assume there is no external force \mathbf{F} , so Eqs. (15) and (5) reduce to

$$\rho \mathbf{u} \cdot \nabla \mathbf{u} = \nabla \cdot (-p \mathbf{I} + \mu (\nabla \mathbf{u} + (\nabla \mathbf{u})^T)), \quad (16)$$

and

$$\rho (\nabla \cdot \mathbf{u}) = 0. \quad (17)$$

Equations (16) and (17) are used in my original laminar model to solve for pressure and velocity of the flows. The reduced forms prove to be easier to solve because a laminar model takes less time to converge than a turbulence model, which instead uses the compressible and unsteady form of the Navier-Stokes equations.

2.7 The calculation of lift

How much lift an airfoil generates is dependent on the shape of the airfoil, the angle of attack, the velocity, density and viscosity of the fluid, and other conditions. An approach to quantify how these variables affect lift is to relate lift to the **lift coefficient**:

$$C_L = \frac{2F_L}{\rho v^2 A}, \quad (18)$$

where F_L is the lift which is usually measured experimentally in wind tunnels, ρ and v are density and velocity of the fluid respectively [25]. Area A is the **wing area**, defined as the projected area bounded by the leading and trailing edges and the wing tips. Through Eq. (18) we can then compute the lift coefficient of an object immersed in a flow, such as an airfoil or wind turbine blade. A higher lift coefficient means the object is more efficient at generating lift, as per density per velocity square and per unit area, the lift measured will be higher according the Eq. (18).

Since I do not have a wind tunnel, I rely on COMSOL to solve the Navier-Stokes equations, which is discussed in section 2.8. Once the computer finds solutions to the Navier-Stokes equations, we know the pressure at any point in the fluid. As we discussed previously, that lift derives from a pressure difference between the upper and lower surface, we integrate pressure values over the surface of the airfoil to get lift:

$$\mathbf{F}_L = \oint p \hat{\mathbf{n}} dA, \quad (19)$$

where $\hat{\mathbf{n}}$ is the normal vector perpendicular to the surface [26]. Notice Eq. (19) does not guarantee the lift will point upwards. In my project, however, I am only concerned about the y -component of the lift that is either driving or opposing the linear vertical motion of the airfoils. I will give this more discussion in my energy calculation in section 4.1.

2.8 COMSOL and the finite-element method

In section 2.7, I raise the point of solving Navier-Stokes equations using computer programs. Clearly, Eq. (16) and Eq. (17) are still impossible to solve analytically, despite that they are the simpler forms of Navier-Stokes equations and the continuity equation respectively. Therefore, I resort to a piece of software called COMSOL Multiphysics to solve the Navier-Stokes equations numerically. COMSOL stands for “computer solutions” while Multiphysics means that “multiple fields of physics can be solved together simultaneous in one model”, providing valuable integration of related physical phenomenon. COMSOL is a finite-element solver that uses finite-element method to solve partial differential equations (PDEs) [27]. Since my study deals with fluid flows, I use the computational fluid dynamics (CFD) module of the Multiphysics package to solve for pressure and velocity of fluid flows. The software allows me to simulate the energy generation process of the DWG through specifying the geometry, the flow physics, the initial conditions, and boundary conditions, etc. Simple as it sounds, learning how to use the COMSOL Multiphysics CFD module to produce simulation results that are consistent with theoretical predictions is quite challenging, which I will discuss in Chapter III.

At the core of COMSOL’s capacity to solve complex PDEs is a numerical technique called the **finite-element method (FEM)**. FEM divides a complex PDE problem into finite-elements, which are small and discrete regions where the solutions are easier to find. Once the solutions for these finite-elements are found, they are recombined at the element boundaries to form a continuous approximate solution to the original PDEs [28]. Figure 12 illustrates this process by trying to find a piecewise function that approximates a curve. The curve is first divided into four pieces by equally spaced parallel black dotted lines. For each part of the curve, FEM tries to find as the solution a

line or curve segment that minimizes the difference between itself and the original curve, which in Fig. 12 is the area between the curve and the line segments. Clearly, line segments 2, 3, 4 are good solutions as they fit their respective part of the curve well. So when we combine line segments 2, 3, 4, we get a good approximation of that part of the curve. In contrast, line segment 1 is a bad fitting to its corresponding part of the curve as it leaves a considerable gap between itself and the curve. We will encounter a non-convergence problem if the area of the gap exceeds the tolerance of the FEM [31], which is a predefined maximum relative difference allowed between the true solution and the computer's guessed solution. When solving PDEs, COMSOL will try until a certain number of iterations to update the solution to reduce the difference between its guessed solution and the true solution before declaring non-convergence. So in this case, a computer program may try to update the fitting line 1 to a fitting polynomial, which may potentially resolve the convergence problem.

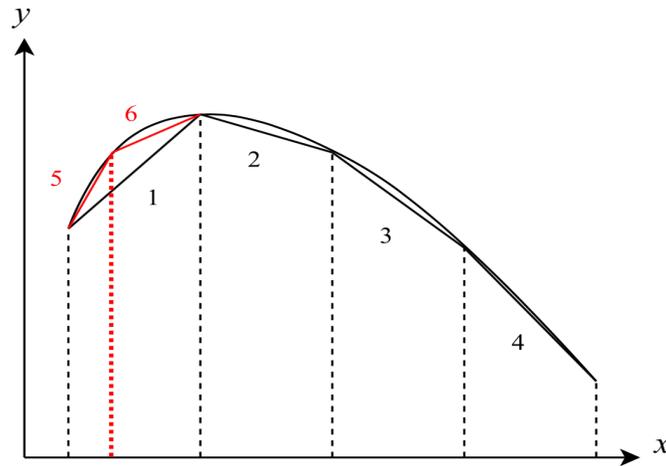


Fig. 12. Illustration of the finite-element method to find a piecewise function to approximate the curve.

However, if we are constrained to using a straight-line segment, it is almost certain that the computer program will reach the maximum number of iterations before successfully bringing the difference down to within the tolerance. This non-convergence issue is what happened to my modeling frequently in COMSOL, and to avoid it is my major motivation to start with a model as simple as possible, so that it converges, before upgrading it gradually to the level of complexity that matches my DWG model. Apart

from simplifying the model as a way to increase the chance of convergence, one can also refine the mesh. **Meshing** is the process of discretizing the problem domain into fine elements. Dividing the curve by four equally spaced dotted lines resembles an automatic meshing process by COMSOL, which has no information about the geometry. For a user who knows the geometry, he can strategically refine the mesh at places of large curvatures, such as corners, where the solution tends to be more complex compared to the solutions along straight lines. For example, introducing a red dotted line to split the part of the curve in region 1 into two “finer” elements reduces significantly the gap between the curve and the fitting straight-line segments. Refining the mesh at places with large curvature not only increases the chance of convergence for models that do not originally converge, but also reduces the number of iterations tried to find the solution for models that do converge originally.

III. Modeling: How to simulate fluid flows in COMSOL CFD?

3.1 Model and parameter specification for dual-wing generator

I base the geometry of DWG on the NACA 0012 airfoil. Airfoil shapes are commonly characterized with a numbering system originally defined by the National Advisory Committee for Aeronautics (NACA). This characterizing system defines airfoil shapes with a series of digits corresponding to non-dimensionalized airfoil properties. The number of digits used to describe an airfoil corresponds to the complexity of the airfoil [8]. A detailed equation that describes the shape of the 0012 airfoil will be given later in this document.

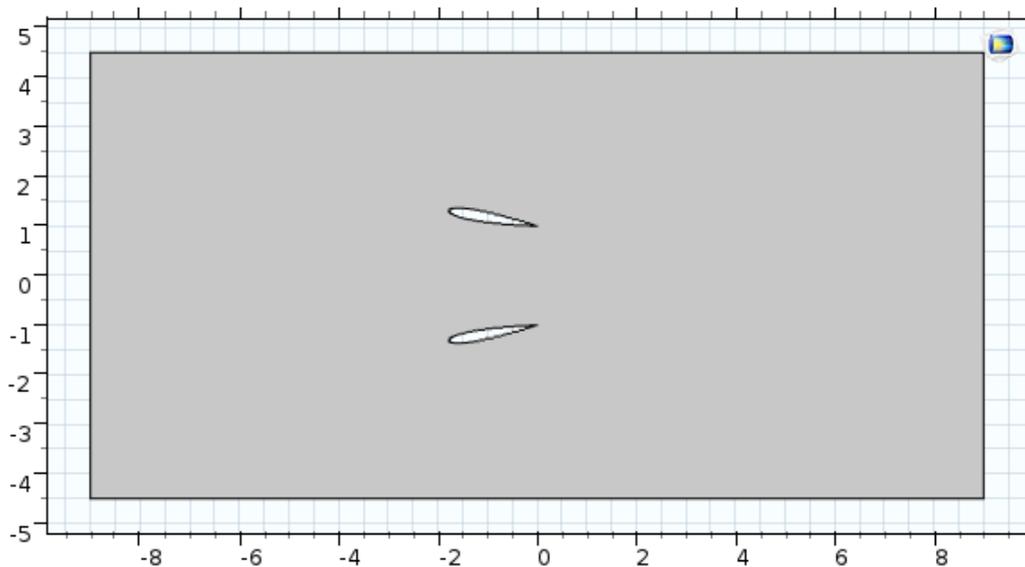


Fig. 13. Geometry of the DWG in a domain of airflow (grey). Inlet is the left boundary. The two airfoils are 2 m apart and tilted 10° from horizontal position.

Now I proceed to build my dual-wing system with the COMSOL Multiphysics software 5.2 with its Computational Fluid Dynamics (CFD) module. In the starting interface, I choose the “Model Wizard” initialization that leads to predefined models, whose specifications are asked for in several follow-up prompts. For “space dimension”, I chose the “2D dimension” model; for “physics interface” I originally chose “laminar flow (spf)”, where “spf” stands for single-phase flow; and for “study” I chose “stationary

study”, which means the model is time-independent. I use the 2D model instead of the 3D model because it reduces significantly the computational complexity without sacrificing the validity of my simulation. By using the 2D cross section of the airfoils, I am assuming the airfoils are much longer than they are wide, and that the major energy efficiency improvements are achieved through the configuration in the two dimensions presented in the model. I also chose laminar flow instead of turbulent flow originally, because turbulent flow models seldom result in a converging result. Therefore, for the same benefit of having affordable computational complexity given the time constraint on this project, I opted for the laminar model as my original choice. The assumption is that the geometric benefit of this unique dual-wing configuration should carry over from one model to the other, be it turbulent or laminar. This choice, however, later proves to be problematic as I discuss in section 3.5, and I eventually switch to a turbulence model for my analysis. Despite this final change, I stick to using laminar model here as an example for this section. The single-phase flow means the dual-wing system is entirely immersed in a single phase, in this case air, where the simulation does not involve boundary crossing between two different media. Lastly, the choice of the stationary model is aimed at reducing simulation complexity. The time-dependent study requires a different setup, and to which the transfer of our stationary simulation result needs additional time and work beyond the scope of this project. But my stationary simulation at different stages of the dual-wing movement cycle still gives us insights into and approximation about what a continuous simulation looks like.

Before building the geometry in Fig. 13, I set some parameters for the model. I create a “Parameters” tab under “Global Definitions” to specify global parameters. Global parameters are variables used throughout the model, but can be reset by users for different values. There are three variables I hold constant in my simulation, although I occasionally change them for the purpose of testing prior to the actual simulation.

First, I set “ U_{inf} ”, the free-stream velocity of the medium to be 2 m/s. This is the velocity I assign to the inlet boundary condition. It is the velocity of air far from the dual-wing configuration, and “inf” stands for “infinity” although the air is not in reality coming from infinitely far away. As the air approaches the DWG, its velocity changes. I also start with a relatively conservative inlet velocity of only 2 m/s to ensure the

Reynolds number is within the laminar flow region. If I choose too large a velocity, such that the Reynolds number exceeds the laminar flow limit, the simulation will not converge. Readers should notice that, however, this is my original assumption. I later realized the problem with a laminar model is largely due to the extremely small value of air viscosity rather than using a large velocity.

Second, I set “ L ”, the horizontal dimension of the domain, to be 18 m, mainly because it is large enough compared to the chord length “ c ” which I set to be 1.8 m. It is crucial to choose a small enough chord size so that the dimension of the domain is large enough relative to the dimension of the airfoil and thus that the boundaries of the domain appear far away, even though not at infinity, from the dual-wing system. In this case, the ratio of domain dimension to the airfoil dimension is 10 : 1. Readers should again know that these values are temporary and are revised later. I stick to these assigned values here because the first half of my project uses these values consistently. Newly assigned values are reported in section 4.1.

There are two other parameters that I change in my simulation for my quasi-static simulation. One is “ r ”, the rotation of the airfoil from its horizontal position, which represents the angle of attack, and the other one is “ d ”, the displacement of the trailing edge of the airfoil from the origin of the domain. Notice the displacement is half the separation between two airfoils. In the model shown in Fig. 13, r is 10° while d is 1m.

3.2 Geometry of DWG

After parameter specification, the next step in the model is to build the geometry of the configuration in Fig. 13. I start with building a single airfoil embedded in the domain, which is symmetric with respect to the chord. First, under the “Geometry”, I draw a parametric curve that represents the shape of the upper surface above the chord of the NACA 0012 foil. The expression for y is $c \times 0.594689181 \times (0.298222773 \times \sqrt{s} - 0.127125232 \times s - 0.357907906 \times s^2 + 0.291984971 \times s^3 - 0.105174696 \times s^4)$ [29], while the expression for x is $c \times s$, where c is the cord length, set to be 1.8 m initially, and s is the horizontal proportional displacement from the origin and has a value from 0 to 1. With these two expressions, COMSOL draws a curve highlighted in blue in Fig. 14(a).

Secondly, I draw the second half of the surface using the same expressions, and take the “union” operation to consolidate the two symmetric curves into one curve, called “uni 1”. This “uni 1”, however, is still just a curve object. To turn the curve into an airfoil, I perform the “convert to solid” operation that solidifies the area bounded by the curve into one two-dimensional object. Notice that the airfoil is grey, suggesting the airfoil body is currently the domain bounded by hollowness. But we do not want to perform the finite-element analysis within the airfoil body, but rather we want to perform the finite-element analysis in the air around the airfoil. Thus we need to invert the domain area so that the airfoil is a hollow shape embedded in a solid domain.

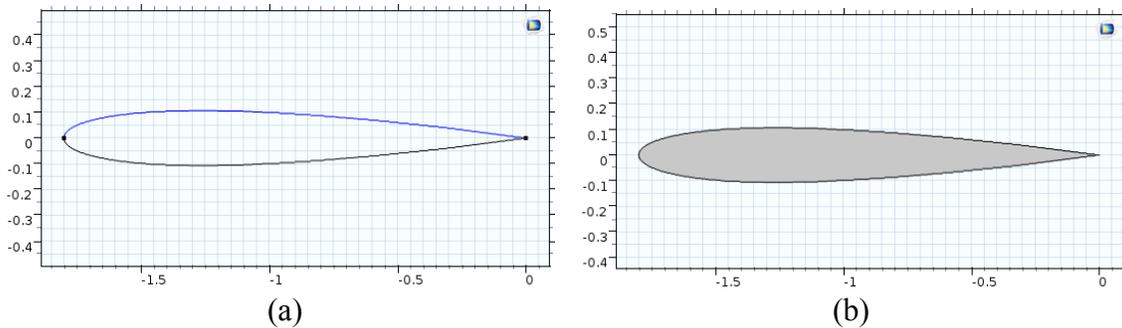


Fig. 14. (a) Curves denote the surface of the airfoil. (b) A solid airfoil.

Thirdly, I create a domain object that is large enough to contain the airfoil object to embed the airfoil in the domain. Specifically, I draw a circle of radius L centered at the origin and take the difference between the circle object and the airfoil object, resulting in an object called “dif 1” where a hollow airfoil is embedded in the circle domain. Notice the trailing edge of the airfoil is positioned at the origin, so that when I rotate the “dif 1” object, the position of the trailing edge remains unchanged. I choose a circle as the shape of this tentative domain because the outline of a circle domain will not change when it is rotated with respect to the center of the circle.

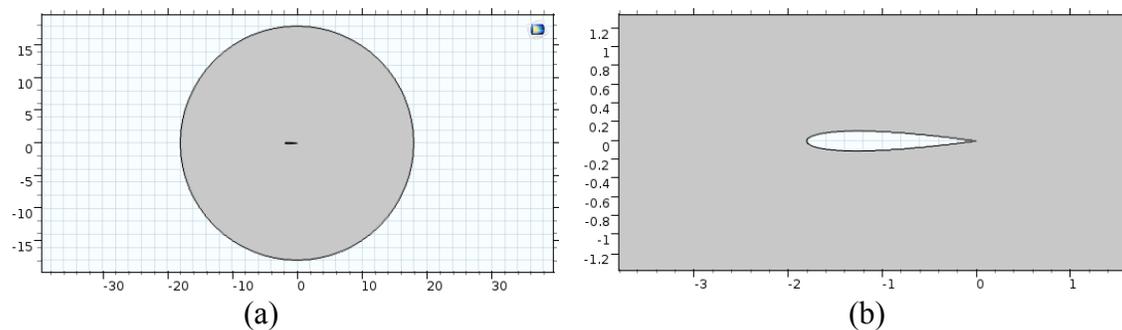


Fig. 15. (a) A temporary circle domain of air. (b) The embedded airfoil shape.

Fourth, I rotate the “dif 1” object by “ r ”, the parameter for angle of attack, and the new object created is labeled “rot 1”. Having the rotation angle as a parameter gives me the option to build airfoils tilted at other angles for the quasi-static analysis later on. In this case the rotation will be 10° for this generic model. Fifth, I move the “rot 1” object in the y direction by “ d ”, the displacement of the airfoil from x -axis, which can be both a negative or a positive value. Again, having the displacement as a parameter allows future adjustments to our geometric configuration. With my d value currently set to 1 m, this operation moves the “rot 1” object upwards by 1 m and the new object is called “mov 1”.

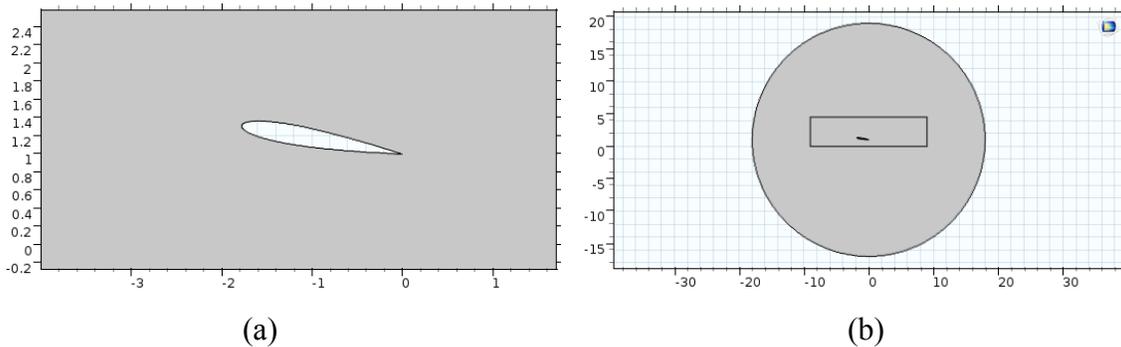


Fig. 16. (a) One airfoil tilted. (b) The rectangle domain on top of the tilted airfoil.

Sixth, I create the final domain and perform a mirror operation to complete the dual-wing configuration. I draw the upper half of the rectangle domain in Fig 16(b) on top of our current “mov 1” object. I then perform an intersection operation that leaves only the overlap of the “mov 1” object and my newly drawn rectangle object, labeled as “int 1”. Notice I could have initially used a rectangle rather than a circle for my tentative domain. But my rotation operation will rotate the rectangle as well, after which the resulting object may not be able to contain the final rectangle of length L for the intersection operation, unless a rather large rectangle domain is tentatively deployed. Next, with the “mirror” operation with respect to the x -axis, I create a symmetric domain and “union” the two domains, giving me the final domain with embedded airfoils as shown in Fig. 17(b). Notice the rectangle domain is centered at the origin and has width $L/2$ and length L , the dimension I specify in the global parameter section. The two airfoils are both rotated by r away from the x -axis. Their trailing edges are on the $x = 0$ line and are $2d$ apart. In my quasi-static analysis, I leave the domain dimensions unchanged but I

actively change r and d to capture the configuration of the dual-wing system at different stages in the cycle.

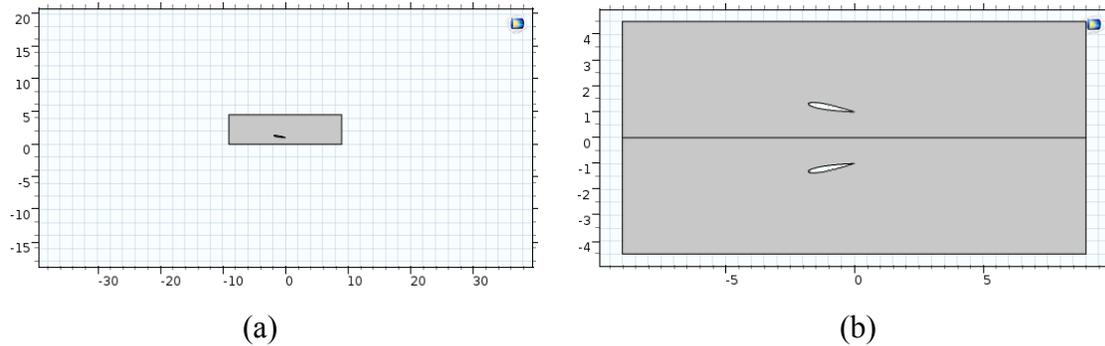


Fig. 17. (a) One half of the final domain. (b) Overview of the final domain.

3.3 Boundary conditions

After building the geometry, essential to the finite-element analysis is the specification of boundary conditions. In Fig. 18 I highlight all the boundaries of the domain. The red boundary designates the inlet, while the yellow boundaries designate outlets. Inlet is where the air flows in, and specifically at which the velocity is U_{inf} in the x direction. There is no y component of the velocity of the incoming air. Outlet is where the air flows out and can be regarded as a sink of the fluid. Blue boundaries are walls which air cannot flow across. At walls, in this case the surface of the airfoils, the “no slip condition” is applied. “No slip condition” means that friction prevents the fluid from “slipping”; in other words, the velocity of the fluid must be 0 at the walls. So there should be a velocity gradient in the direction normal to the walls, so that the speed drops to 0 at the wall. Because of this velocity drop, I expect to see pressure build up in some regions close to the airfoils.

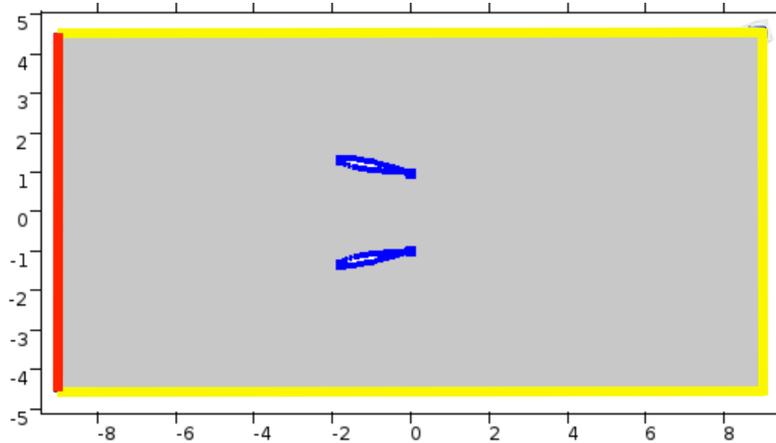


Fig. 18. Boundaries of the domain. The blue boundaries are walls, the red boundary is the inlet, and the yellow boundaries are the outlets.

The decision of which boundary to assign the inlet is straightforward because in my analysis the air blows from left to right, initially in the positive x direction. Also, the decision regarding the walls is straightforward, because the only non-penetrable boundaries in this domain are the surfaces of the airfoils. However, it might seem a little tricky for the assignment of outlet boundaries. Clearly, the right side of the rectangle must be an outlet where most of the airflow exits. As for the upper and lower boundaries, it should not matter whether they are walls or outlets if they are far away from the airfoils. But because I am using a domain that's relatively small compared to the size of the airfoils, I often run into non-convergence issues with the finite-element computation when the upper and lower boundaries are set to be walls. If I assign these two boundaries to be outlets, then the non-convergence issue is usually resolved without additionally mesh refinement. This setup also corresponds better to the actual situation of wind blowing to the dual-wing system, where it is possible for air molecules to escape from the imagined boundaries of the airflow into its ambient environment.

3.4 Meshing of DWG

Having set the boundary condition, the next step is meshing, the process of dividing the domain into fine elements over which the FEM is performed. A generic triangular element is by default the shape of a 2D fine element. Each triangle is governed by an independent set of Navier-Stokes equations, which specify the velocity and pressure of any point within the fine element. Therefore, finer the mesh is, in other words, smaller and more concentrated the triangles are, better the approximations are. Yet an equally fine mesh over the entire domain is not necessary. First, the airflow far from the airfoils is less variable in pressure and velocity, so the same equations can accurately describe a larger element. A coarser mesh can therefore be used in regions far from the surface of the airfoils as illustrated in Fig. 19(a) to reduce the computation complexity. Second, pressure variations tend to occur at surfaces where normal vectors change rapidly, such as at sharp corners. An automatic meshing sequence of COMSOL assigns more triangle elements to the leading edge of the airfoil to accommodate the pressure variation as shown in Fig. 19(a). Along the upper or lower surface of the airfoil, where normal vectors do not vary significantly, rectangular mesh elements are assigned by default. Blue dots in Fig. 19(a) mark the distribution of the rectangle mesh elements along the surface. Each two neighboring dots form the longer side of a rectangle, and more rectangles stack on top in the normal direction to the surface. This mesh generated automatically by the program does not lead to a converging finite-element analysis for it leaves the trailing edge of the airfoil with a coarse mesh, at a corner that is even sharper than the leading edge of the airfoil. In Fig. 11(b), I replace the rectangular elements along the surface with triangular elements and refine the mesh at the trailing edge significantly. As a result, the FEM converges, and this is the mesh pattern I use throughout my quasi-static analysis.

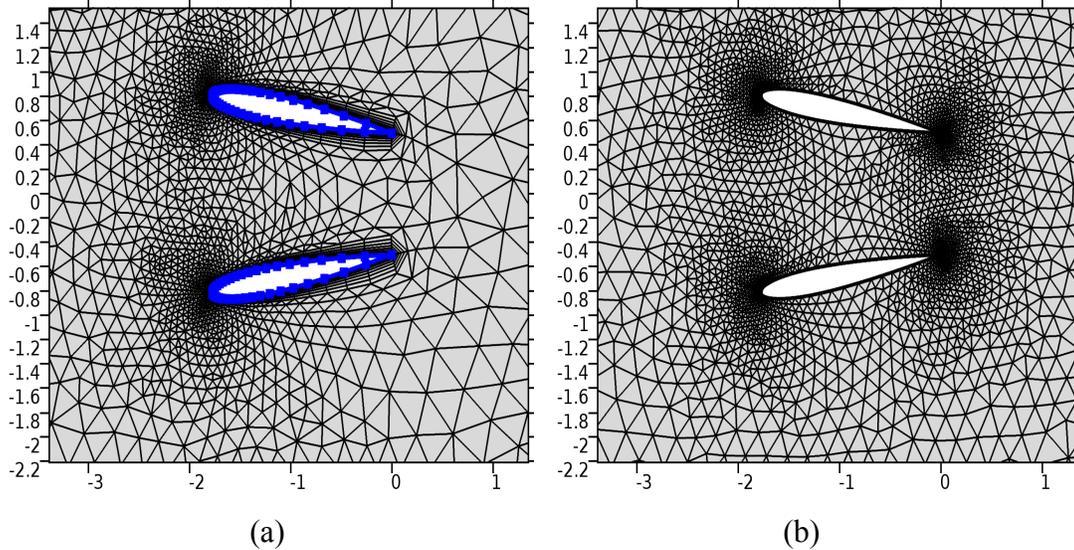
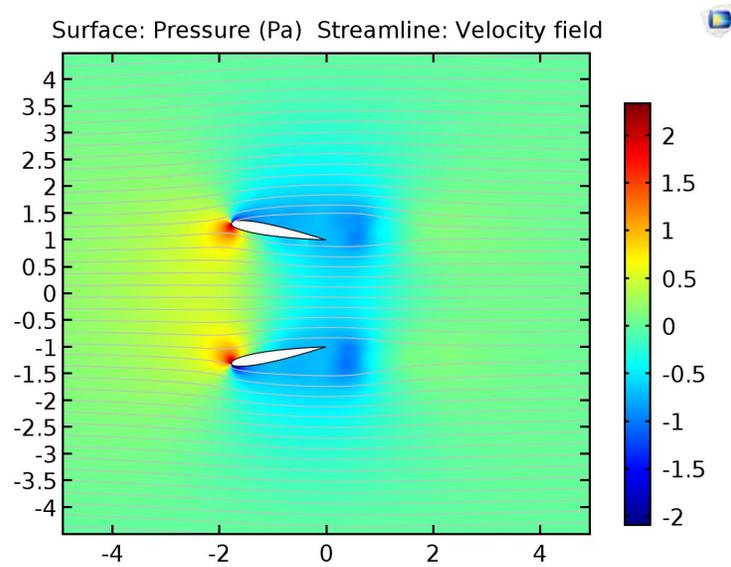


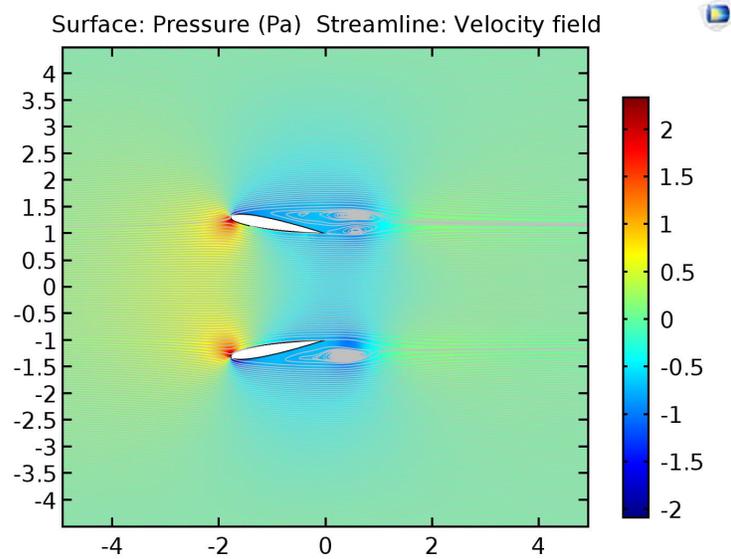
Fig. 19. (a) Automatic mesh sequence generated by COMSOL. (b) My mesh sequence after refining the mesh at the trailing edges.

3.5 Validity of the model

Having built the DWG model based on a laminar flow assumption, it is important to check its validity before we proceed to any analysis. Running the simulation with the current settings gives us a flow plot in Fig. 20 (a), which looks like a valid result at first glance. First, the model converges, confirming that we have solutions to the PDEs. Second, we do see a pressure build-up at the stagnation point and a pressure drop at the outer surface of both airfoils, producing lift at both surfaces of both airfoils. All seem to encourage us to move forward to analysis, except for an unusual observation of a flow separation starting almost at the leading edge of either airfoil. The separation of flow indicates the viscous force is not large enough compared to the inertial force in order to “pull” the airflow close to the contour of the airfoil. It suggests a high Reynolds number and a turbulent flow, and contradicts our laminar assumption. In fact, if we give the streamline plot a higher density of streamlines, we will see a large region of turbulence as shown in Fig. 20 (b).



(a)



(b)

Fig. 20. Surface plot of pressure and streamline plot of velocity for DWG in a laminar model. $L = 18$ m, $c = 1.8$ m, $U_{inf} = 2$ m/s, $d = 1$ m, $r = 10^\circ$. The surface plot in color shows the pressure above 1 atm and the streamline plot shows the velocity field. The difference between part (a) and (b) is the number of streamlines per unit area plotted.

Although our theory [32] allows a transition from a laminar zone into a turbulent zone along a surface, having it occur right after the leading edge is problematic. The

airfoil misses a large mass of airflow to deflect downwards and thus cannot produce meaningful lift. That streamlines closest to the airfoil in Fig. 20 (a) pass through the airfoil almost horizontally without turning downwards is a confirmation of lift generation being missed. This conclusion follows fundamentally from Newton's third law, as discussed in section 2.3. A further proof of the invalidity of this model is that when I calculate the lift produced by this model with different r , which corresponds to angle of attack, I do not see the stall effect on lift generation. A **wing stall** describes the sudden loss of lift when the angle of attack of a wing increases past a point when major flow separation occurs. Before the stall point, lift increases linearly with angle of attack until the stall point, where it takes a downturn and begins to decrease. This laminar model fails to predict a lift drop when I increase r past 14° , an angle where a wing stall usually occurs [29]. When I decrease r to close to 0° , I also fail to resolve flow separation, contrary to that fact that low angle of attack should not produce much turbulence.

To resolve the flow separation, I investigate the same model but with only one airfoil at the center of the domain in order to reduce complexity. All the parameter settings remain the same, including r , d , and U_{inf} . I start with the equation of Reynolds number, Eq. (2), $Re = \frac{\rho V c}{\mu}$, to find what parameter contributes to an inertial force and viscous force imbalance and thus a large Re value. Notice I substitute chord length c for the characteristic length L in Eq. (2), because it is over the surface of the airfoil of chord length c that a boundary layer develops. Among all the values I assign to the model, the air viscosity μ is the most extreme. Compared to water, air has low level of "stickiness" and thus little viscosity, specifically, 1.81×10^{-5} kg/m·s [29]. This extremely small value gives rise to a large Reynolds number that far exceeds the range for laminar flow. As I artificially increase the viscosity of air, the flow separation starts to disappear when μ increases to a level of 10^{-2} kg/m·s. So the model would work if air has 1000 times higher viscosity. However, air is air. Therefore to base the model on laminar flow is an invalid assumption to start with, despite its convenience of low complexity.

To switch from a laminar model to a turbulence model is my next step. This switch introduces some more nonlinear PDEs that increase the challenge and time complexity for COMSOL to find a numerical solution [33]. Therefore, it is crucial to find a simple yet accurate enough turbulence model so that the simulation converges in a short

period of time. After repeated trial and error, I find the $k-\varepsilon$ turbulence model, which calculates a time-average of turbulence, converges relatively quickly and produces a beautiful result, shown in Fig. 21, which is compatible with the theory of lift generation. Specifically, we see streamlines separate at the stagnation point and meet again at the trailing edge of the airfoil, wrapping around the airfoil in a tight fashion. This suggests the characteristic length, in this case the chord length c , is within the laminar range calculated by this turbulence model, and thus we do not see a transition into the turbulent zone along the airfoil surface in this plot. As a result, a clear pattern of streamlines being deflected downwards by the shape of the airfoil emerges, indicating a good amount of lift generated in the process according to Newton's third law.

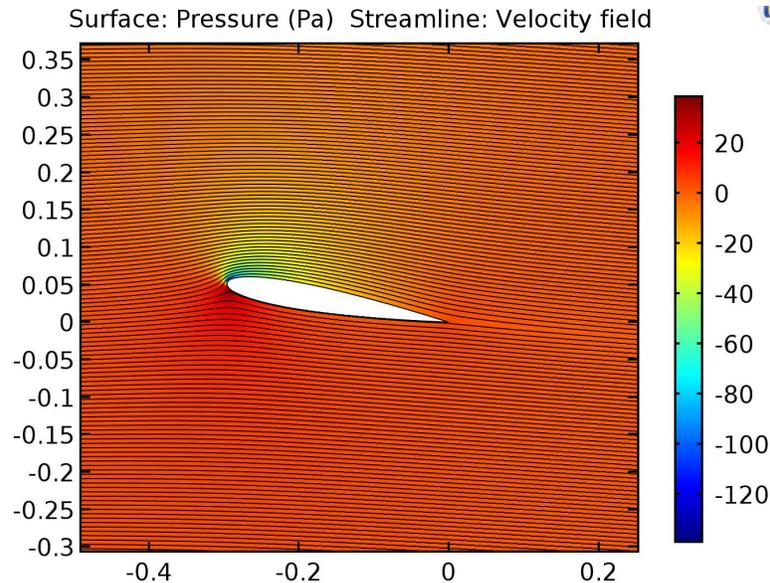


Fig. 21. Surface plot of pressure and streamline plot of velocity for DWG in $k-\varepsilon$ turbulence model. $L = 6$ m, $c = 0.3$ m, $U_{inf} = 8$ m/s, $r = 10^\circ$. There are two clear stagnation points, where the streamlines meet the airfoil right below the leading edge and then separate into two groups to wrap around the airfoil, and where the two groups of streamlines meet again to form one at the trailing edge. Notice I changed the parameters for this simulation, but these changes do not lead to the improved streamline pattern if I do not switch to the turbulence model. In the plot, readers may see alternating dark and bright fringes formed by streamlines. These are moiré patterns, which originate from the interference between pixilation and markings in my figure and have nothing to do with the flow. They emerge in this plot in particular due to the high density of streamlines.

In addition, I also test the $k-\varepsilon$ turbulence model with different angles of attack. The lift for different r is shown in the scatter plot in Fig. 22. We see clearly lift first increases linearly with the angle of attack before it reaches stall at 20° and begins to turn downwards. The close correspondence between this turbulence model and the theory gives me the green light to use it as my DWG model for analysis. When I extend the model to the two-airfoil configuration I build in the previous laminar model, the model continues to converge and the stall phenomenon continues to occur, but this time at 14° , where the airfoils produce highest combined lift. This angle, $r = 14^\circ$, is what I use for the two airfoils when they are moving toward and away from each other. Using the turbulence model also relieves me from sticking to a small U_{inf} , and now I have the freedom to test the model with different wind speeds in my analysis in Chapter IV. The specifics of $k-\varepsilon$ turbulent, where k stands for the turbulent kinetic energy, and ε the rate of dissipation of kinetic energy, and why this model works is beyond the scope of this document and is thus omitted here [34].

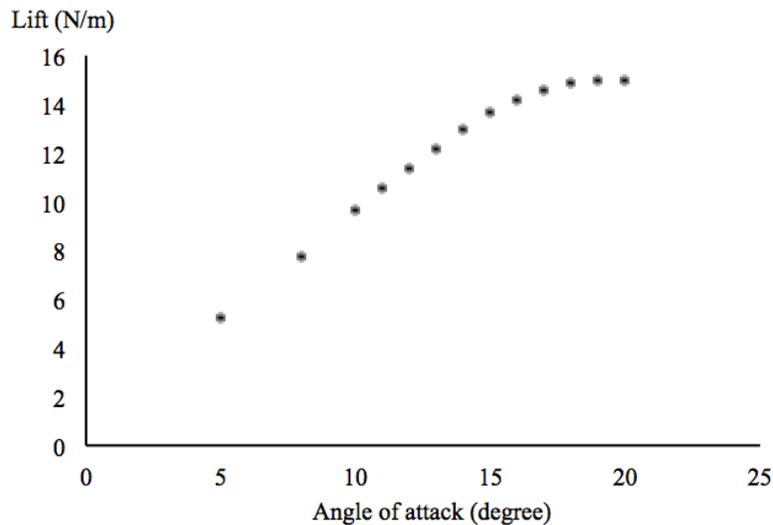


Fig. 22. Lift vs. angle of attack for a single airfoil immersed in an airflow with $U_{inf} = 8$ m/s. $L = 6$ m, $c = 0.3$ m. Selected angles of attack are 5° , 7° , 10° , 11° , 12° , 13° , 14° , 15° , 16° , 17° , 18° , 19° , 20° . The finite-element method ceases to converge above 20° , indicating stall. The lift has unit N/m—that is, lift force per meter of airfoil length—because this is a 2D model rather than a 3D model.

IV. Results: Is DWG more efficient than wind turbines?

4.1 DWG lift calculation in a static simulation

Finally, with the geometry drawn, boundary conditions set, mesh complete, I conduct a parametric study where I modify the geometry for each round of finite-element analysis by assigning different values to the parameters used to draw the geometry. This allows me to simulate the different separation and angle of attack associated with the different stages in a cycle listed in Fig. 2 (a), so that I can conduct a quasi-static analysis. A quasi-static analysis is based on the assumption that the airfoils move slowly enough that each stage in its dynamic process can be considered static. Therefore I can use the time-independent stationary model in COMSOL CFD to simulate the multiple static stages in a DWG cycle as a set of continuous snapshots of the dynamic process.

To simplify the model, I focus on the energy generation from stage 2 – 3 where the two airfoils move apart at a positive angle of attack, and from stage 3 – 4 where the two airfoils move towards each other at a negative angle of attack. I ignore the turning of the airfoils from one angle of attack to the other, both at the upper and lower ends of the DWG; as I have explained in section 1.1, the turning process adds approximately zero net energy.

After many trials and tests, I reset the parameters for the DWG simulation, and the final settings for my analysis are: $c = 0.3$ m, $L = 6$ m, $U_{inf} = 8$ m/s, and the angle of attack = 14° . I change the original chord length of 1.8 m as I want to match my simulation to the actual DWG. The new choice of $c = 0.3$ is from my eye estimate of the actual size of the DWG airfoil presented in Festo's DWG brochure [1]. Accordingly, I reduce the horizontal dimension of the domain L from 18 m to 6 m. In the process, I nevertheless increase the ratio of domain dimension to the airfoil dimension from the original 10:1 to 20:1, as it helps convergence at larger U_{inf} values. I also increase the free-stream velocity U_{inf} from the previous 2 m/s to 8 m/s. On the one hand, I can afford the increase of velocity now without any convergence issues as I have switched to a more accurate turbulence model than the previous laminar model. On the other hand, the wind speed of 8 m/s is the maximum threshold at which the DWG still outperforms

the traditional wind turbine in Fig. (3), and I want to test this result. I chose the angle of attack to be 14° , as stall occurs beyond this angle and lift drops for the DWG.

Once COMSOL finds a converging solution for the DWG model with the above settings, I have access to the pressure values along the airfoil surface, which are stored in the expression “ p ” in COMSOL. Equation (19) suggests that with pressure information, I can calculate lift through a surface integral. Because I use a 2D model, instead of a surface integral of pressure over the airfoil’s surface to get force, I perform a line integral of the pressure along the airfoil surface, which is the boundary line in this 2D model, and thus I get force per unit length (N/m). The integrand of my line integration is “ $\frac{dy}{y} \times n_y \times p$ ”, where n_y is the y -component of the normal vector of the surface, which means the integral is not lift but the y -component of the normal vector of the surface. Because the airfoils only move in the y -direction, only the force pointing in the y -direction contributes to energy generation. As a disclaimer, in the following passage when I say lift I mean the y -component of the lift, unless otherwise specified. Notice the airfoil is tilted, and thereby the drag also has a y -component that contributes to the energy output. However, because the airfoil is only tilted at 14° , giving $\sin(14^\circ)$, a small value. Also, because air has extremely low viscosity, I leave out the consideration of drag in my energy calculation.

When I perform the line integration, I need to select all the boundaries, including the upper and lower surface of both the upper and lower airfoil, over which the integration occurs. But the lift generated by the two airfoils has opposite directions as the airfoils have opposite angle of attack. Consequently, COMSOL’s output of the net lift is 0 as the symmetry with respect to the x -axis gives the two lift values same magnitude but different signs. Therefore, we want to instead sum up the absolute value of the lift generated by the two airfoils. As the two airfoils differ in the positivity of their y -coordinates, and that COMSOL happens to have an absolute function, I create a sign function $\frac{|y|}{y}$ that can make the negative lift value from the lower airfoil, which has negative y -coordinate, positive. Then I can integrate over the surface of both airfoils at once in a single line integration to get the total y -component of lift.

4.2 DWG lift results and energy calculation

In the $U_{inf} = 8$ m/s DWG cycle, the wind exerts a force on the airfoil in the direction of the airfoil's motion. Therefore the wind does work on the airfoil and transfers its wind energy to the kinetic energy of the airfoils. Energy conservation has [14]

$$\Delta E = \int F_y dy \quad (20)$$

where F_y is the y -component of lift in the y -direction. If I take a small enough dy , the F_y can be regarded as the same, and I can then approximate the integral by $F_y \times \Delta y$. In my quasi-static analysis, the consecutive simulations differ only in the separation between the two airfoils by a small amount of 0.02 m. In other words, I vary d to simulate the moving of two airfoils first apart and then together, specifically from $d = 0.1$ m to $d = 0.3$ m when $r = -14^\circ$, and then $d = 0.3$ m to $d = 0.1$ m when $r = 14^\circ$. The choice of $d = 0.1$ m and 0.3 m is again based on the eye estimate of the minimum and maximum separation between the two airfoils in a DWG cycle [1]. The $r = -14^\circ$ corresponds to a positive angle of attack, and vice versa, as I rotate the chord line against the origin clockwise by 14° so that the extension of the chord forms a -14° angle with the positive side of the x -axis.

For each complete cycle from $d = 0.1$ m to $d = 0.3$ m, consisting of the airfoils moving apart when $r = -14^\circ$ and moving towards each other when $r = 14^\circ$, I also vary the incoming wind speed from $U_{inf} = 8$ m/s to 4 m/s, 6 m/s, 10 m/s and 12 m/s. The lift results are shown in Table I, and there are several interesting observations. First, for the same U_{inf} , the increase in d does not affect the lift much when the two airfoils are tilted apart from each other at $r = -14^\circ$. But it does decrease the lift when the two airfoils are tilted towards each other at $r = 14^\circ$. Second, when the two airfoils are tilted towards each other, the lift is consistently and at times considerable larger than when the airfoils are tilted apart from each other. Third, an increase in wind speed, does boost the lift for both angles of attack, but unequally, as the lift difference between the two angles of attack increases across the table, especially in the first few rows in Table I. Lastly, doubling the lift of a single airfoil with the same parameter settings gives a result similar to the DWG's lift when the two airfoils are tilted away but less than the DWG's lift when the two airfoils are tilted towards each other. The single airfoil's lift is doubled for comparison, as all the other entries in Table I are the sum of lift on two airfoils.

Table I. Lift results of 11 configurations of the DWG in my quasi-static analysis. Angle of attack is controlled by the parameter “rotation”, r , and the separation between the trailing edges of the two airfoils is controlled by the parameter “displacement”, d . Notice the r values are negative, but they correspond to positive angles of attack, while positive r values correspond to negative angles of attack. The values overlaid with grey bars are the lift results (N/m) from simulation. The longer the grey bar is, the larger the lift is comparatively. In the last row, “single x 2” shows the lift, doubled, of a single airfoil tilted at the same angle immersed in an airflow of the same velocity.

| d (m) | U_{inf} (m/s) | | | | | | | | | |
|------------|--------------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|
| | 4 | | 6 | | 8 | | 10 | | 12 | |
| | r (°) | r (°) | r (°) | r (°) | r (°) | r (°) | r (°) | r (°) | r (°) | r (°) |
| | -14 | 14 | -14 | 14 | -14 | 14 | -14 | 14 | -14 | 14 |
| 0.10 | 6.56 | 8.60 | 14.99 | 20.20 | 26.92 | 37.06 | 42.33 | 59.28 | 61.24 | 86.94 |
| 0.12 | 6.40 | 9.23 | 14.65 | 21.51 | 26.30 | 39.07 | 41.36 | 61.92 | 59.85 | 90.12 |
| 0.14 | 6.36 | 9.19 | 14.56 | 21.34 | 26.13 | 38.64 | 41.10 | 61.10 | 59.48 | 88.73 |
| 0.16 | 6.35 | 8.91 | 14.54 | 20.64 | 26.11 | 37.32 | 41.08 | 58.95 | 59.46 | 85.55 |
| 0.18 | 6.34 | 8.51 | 14.52 | 19.66 | 26.08 | 35.48 | 41.02 | 55.95 | 59.36 | 81.12 |
| 0.20 | 6.26 | 8.45 | 14.35 | 19.51 | 25.78 | 35.20 | 40.55 | 55.50 | 58.71 | 80.50 |
| 0.22 | 6.33 | 8.16 | 14.51 | 18.82 | 26.92 | 33.90 | 40.97 | 53.42 | 59.30 | 77.43 |
| 0.24 | 6.32 | 7.95 | 14.48 | 18.32 | 26.30 | 33.01 | 40.85 | 52.04 | 59.12 | 75.44 |
| 0.26 | 6.39 | 7.85 | 14.65 | 18.06 | 26.31 | 32.53 | 41.38 | 51.26 | 59.90 | 74.29 |
| 0.28 | 6.33 | 7.66 | 14.50 | 17.60 | 26.13 | 31.68 | 40.98 | 49.90 | 59.33 | 72.31 |
| 0.30 | 6.39 | 7.57 | 14.65 | 17.41 | 26.30 | 31.34 | 41.36 | 49.37 | 59.86 | 71.52 |
| single x 2 | 6.33 | | 14.43 | | 25.87 | | 40.66 | | 58.82 | |

To explain these curious results, I make the pressure surface plot and velocity streamline plot for these different configurations. In Fig. 24, I present the first four entries in both sub-columns within the $U_{inf} = 8$ m/s column, in total eight configurations. I choose the four $d = 0.10$ m, 0.12 m, 0.14 m, and 0.16 m pairs of configuration because they present the largest difference in lift and show a clear pattern of the suction effect which Festo attributes to the cause of the additional lift [1]. Because the lift is the same for the same angle of attack regardless of whether the two airfoils are moving apart or together, I reverse the order of the eight configurations in Fig. 24 to better illustrate the emergence of the suction effect. Specifically, as the two airfoils gradually move together from Fig. 24 (a) \rightarrow (c) \rightarrow (e) \rightarrow (g), we see the two separate green-yellow low-pressure regions between the airfoils move closer and gradually combine into one region where

the pressure drop is even larger. Such a combined effect is absent in the process (h) \rightarrow (f) \rightarrow (d) \rightarrow (b) when the two airfoils are moving apart. It is not even seen in (h) when the two airfoils are the closest. That the two airfoils are tilted away from each other gives additional separation to the leading edges, at which the pressure gradient is the largest. Therefore, we do not see pressure build up and this process is less lift productive than the previous process. As one process enjoys additional pressure drop and the other misses potential pressure build up, we see the increase of wind speed having asymmetrical impact on the lift generation of the two processes in Table I. Furthermore, neither the pressure pattern nor the velocity streamlines between the two airfoils change much as the two airfoils move apart from (h) \rightarrow (f) \rightarrow (d) \rightarrow (b), suggesting that the two airfoils can be regarded far away from each other even from the beginning in configuration (h). This explains why the lift does not change much for $r = -14^\circ$ as I vary the separation d , and confirms that the result that the doubled lift from a single airfoil matches the total lift produced by the DWG in the process when the two airfoils move apart. Readers may notice a small glitch in (g) as the flow is not symmetric and there is an unexpected flow separation. This is likely due to the fact that the mesh I use is not fine enough, and COMSOL has guessed an inaccurate solution to the Navier-Stokes equations that nevertheless is within the tolerance of the turbulence model.

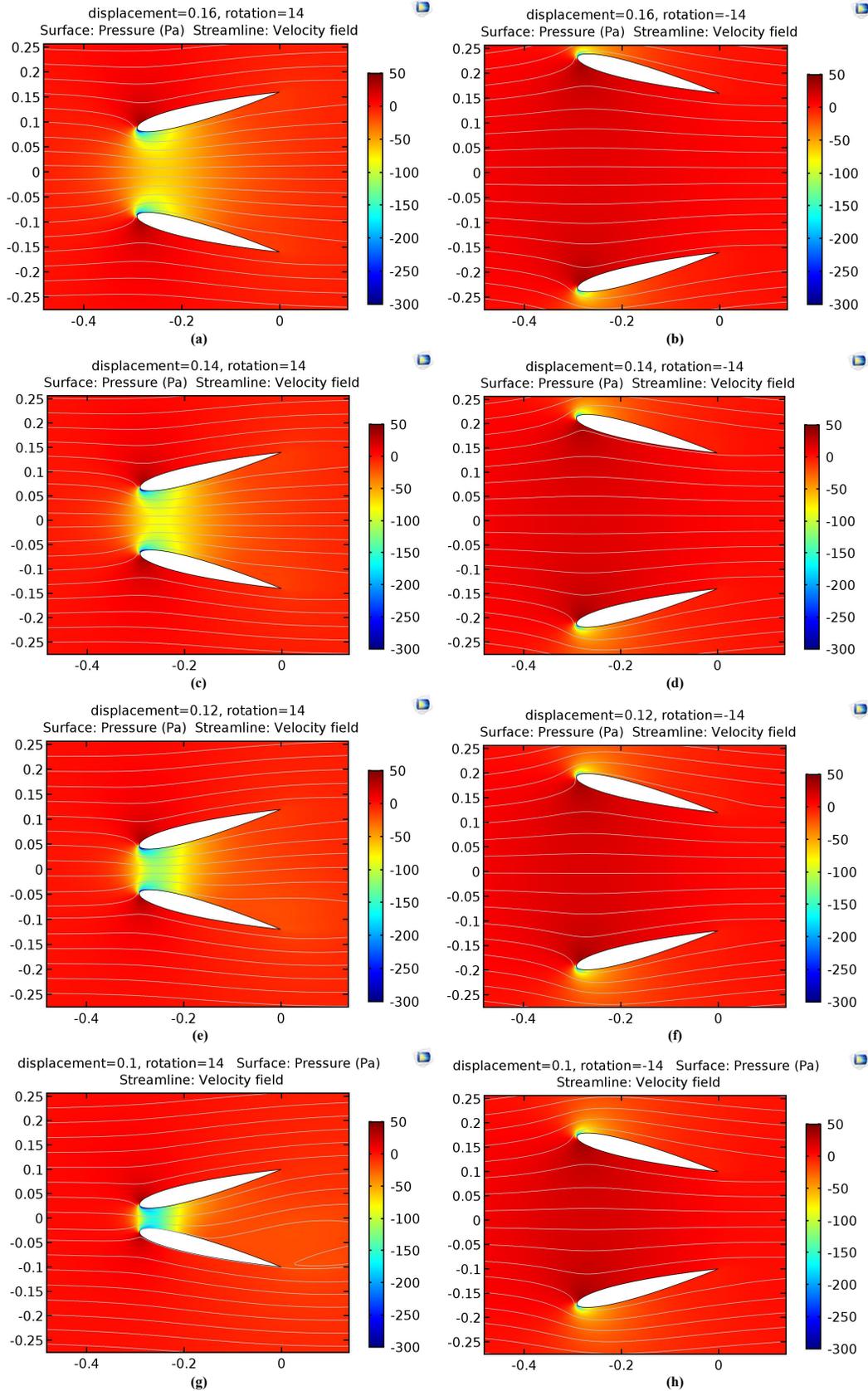


Fig. 24. Pressure and velocity field of eight configurations at $U_{inf} = 8$ m/s.

Because I use the single airfoil model to approximate wind turbine blades as a comparison to my DWG, it is worth the time and effort to take a closer look at the two configurations side by side. This time I look at the velocity surface plot overlaid with velocity streamlines and a pressure contour plot for the $d = 0.14$ configuration when the two low-pressure zones begin to merge in Fig. 24(c). In Fig. 25(a), the flow is much faster, marked as orange and yellow, between the two airfoils in the dual-wing configuration than at the lower surface of the single airfoil. The flow below the single airfoil is an unbounded flow, while the flow between the two airfoils in the dual-wing configuration is constricted by two surfaces and thus moves faster. From another perspective, compared to a fluid parcel at the lower surface of the upper airfoil in the dual-wing configuration, whose higher-than- U_{inf} velocity is sustained by the fast moving layer below it, a fluid parcel at the same position at the lower surface of the single airfoil experiences in contrary a shear stress in the direction opposite to the flow due to the slower moving layer below it, which slows down the fluid parcel. As a result we see a more extended region of fast flowing flow and the formation of a suction effect in the dual-wing configuration. A confirmation from the pressure contour plot in Fig. 25(b) is that the pressure contour lines are more sparsely spaced along the airfoil surface in the dual-wing configuration, indicating a smaller pressure gradient, and thus a greater portion of the airfoil surface is exposed to low pressure.

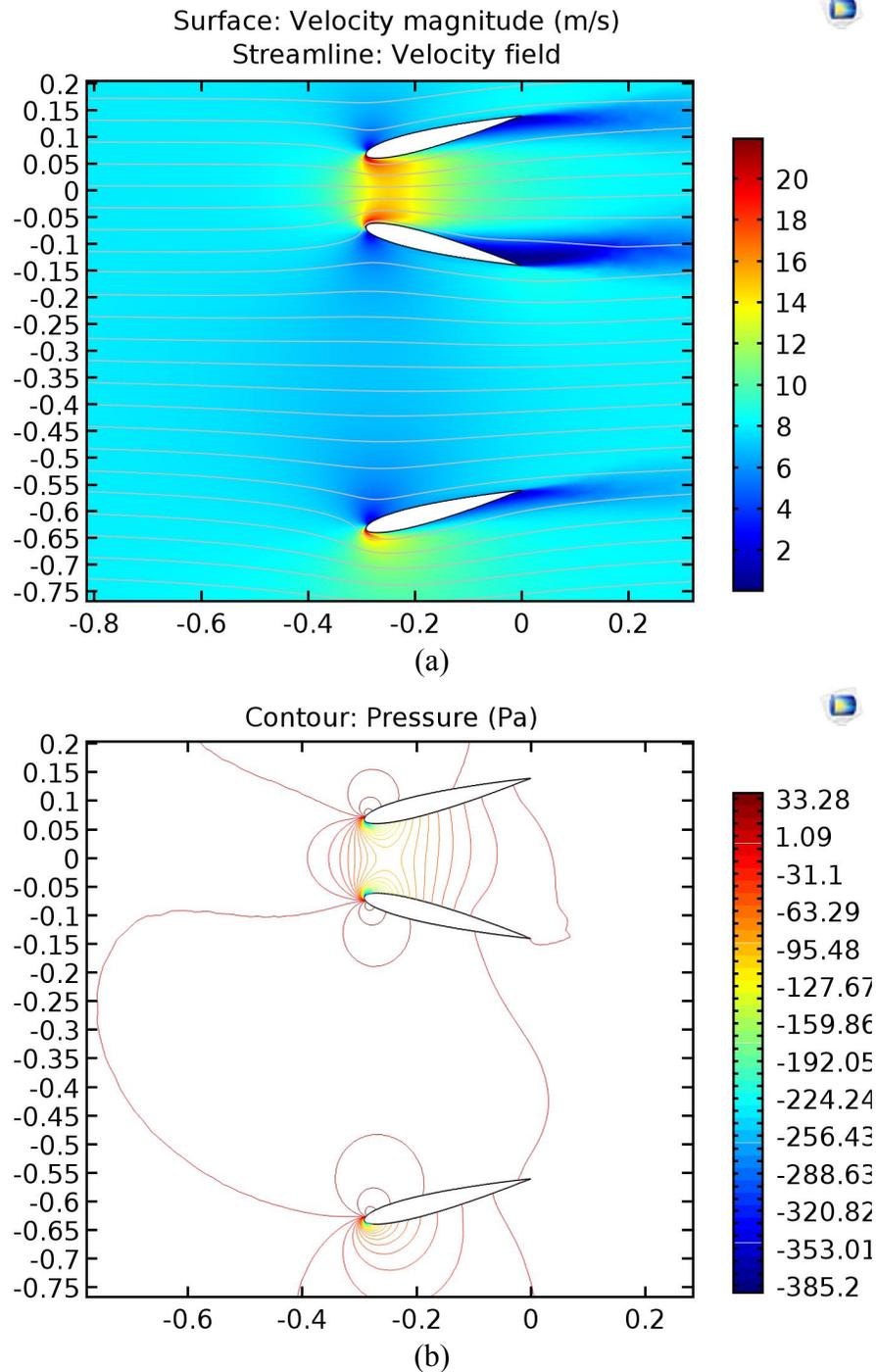


Fig. 25. (a) Velocity surface and streamline plot and (b) pressure contour plot for DWG at $d = 0.14$ m. Wind velocity $U_{inf} = 8$ m/s.

With lift values, the last step in this project is to calculate the energy output via my quasi-static approach based on Eq. (20). The product of the displacement and the y-component of the lift for each stage gives the work done by the wind per unit length. The

work in this case has unit J/m rather than J because my model is based on the cross-section of the airfoils. As an illustration, I plot the $U_{inf} = 8$ m/s column in Table I in Fig 26, where the red dots correspond to results when $r = -14^\circ$ and blue dots when $r = 14^\circ$. The single straight-line shows lift on a single airfoil, which is intended to simulate the lift on a single wind turbine blade in configuration 2(b). In Fig. 26, the blue dots and the red dots represent the lift generated and the area under the dots shows the energy extracted. It is clear that the blue dots, which correspond to the half cycle where the two airfoils are tilted towards each other, enclose more area under them than the red dots, which correspond to the other half cycle. If the single airfoil line represents the wind turbine output correctly, then the DWG is more efficient than the turbine as the area under the red dots and the blue dots together surpass twice of the area under the single airfoil line. The area between the blue dots and red dots is where the additional energy comes from. As the blue dots begin to converge to the red dots as d increases, although not fully shown in Fig. 26, one can speculate that at a large d , the half cycle corresponding to the blue dots is as energy productive as the other cycle. It is true that when the two airfoils are far from each other, they each function as an individual airfoil, regardless of their angle of attack. So it makes sense for Festo to design a DWG where the two airfoils each only traverse a displacement at a scale close to 0.2 m and turn back before the blue dots fully converge to the red dots.

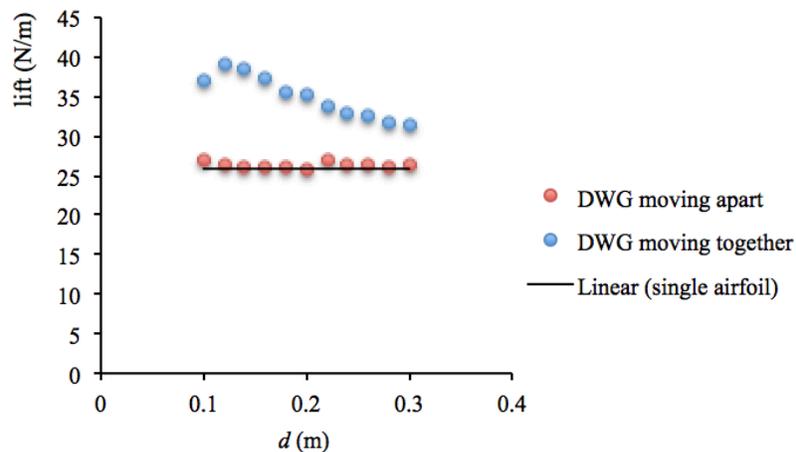


Fig. 26. Difference in lift values generated by the first half and the second half of the cycle at $U_{inf} = 8$ m/s. The first blue dot is an outlier and the velocity field is flawed due to a big separation as shown in Fig. 23(a). The single airfoil line represents the lift produced by a single airfoil.

To calculate the quasi-static energy output for $U_{inf} = 8$ m/s, I compute $0.02 \text{ m} \times \sum_{0.12 \text{ m}}^{0.3 \text{ m}} F_y$, where F_y is the y -component of lift we have in Table I. I leave out the first lift value at $d = 0.01$ m, which corresponds to the first blue and red dots, because they are outliers as shown by the flow separation in Fig. 24 (g). Thus I am left with 10 different lift and 10 displacements of 0.02 for each half cycle, and I assume the airfoils traverse 0.02 m under each lift, during which the force remains constant. Once I get the energy output for 8 m/s, I extend the calculation to the other wind velocities listed in Table I, and the result is shown in Fig. 27.

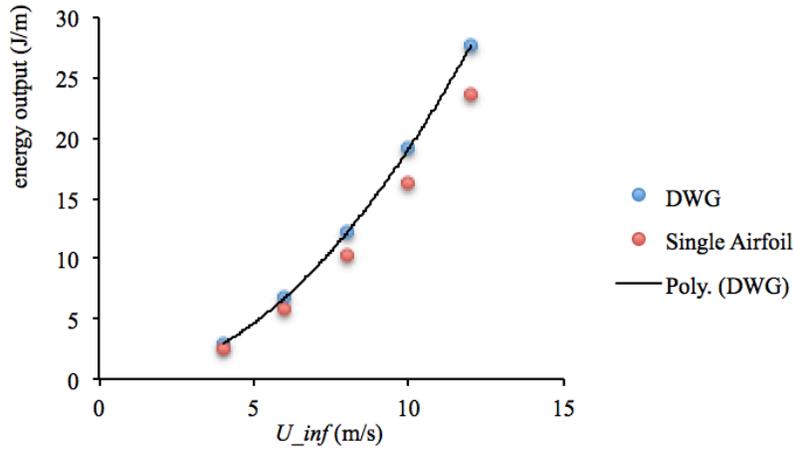


Fig. 27. Scatter plot of energy output for a DWG cycle against U_{inf} . The red dots show the energy produced by one single airfoil doubled.

In Fig. 27, the blue points correspond to the DWG, while the red points correspond to a single airfoil, both for $U_{inf} = 4$ m/s, 6 m/s, 8 m/s, 10 m/s, and 12 m/s. It is noticeable that the superior energy output DWG has over the single airfoil widens as the U_{inf} increases. Although a positive confirmation of the energy efficiency of DWG, it does not match the experimental result presented in Fig. 3, where the wind turbine energy output overtakes the DWG energy output past $U_{inf} = 8$ m/s. To find a potential answer to this curious observation, I first investigate the dependence of DWG energy output on the wind speed by fitting the five blue points with a polynomial of order three. The choice of a cubic polynomial comes from Betz's law. Betz's law states that the maximum power that can be extracted from the wind by a wind turbine is related to the cube of the wind speed [30]. This is intuitively applicable to both the DWG and the

single airfoil: On the one hand the air kinetic energy is dependent upon the mass flow rate and wind speed square; on the other hand the mass of air that passes through the airfoils per unit of time is related to the first order of wind speed. Therefore the transfer of kinetic energy per unit time to airfoils from wind is cubic of the wind speed.

However, the best fitting polynomial to the blue dots I find using Excel is $y = 0.0002x^3 + 0.1941x^2 - 0.0540x$, which is not dominated by the cubic term. The coefficient before the x^3 is almost 1000 times smaller than that of x^2 , indicating the blue points constitute a quadratic curve rather than a cubic curve. This inconsistency is due to my oversimplification of a dynamic process into a static one. By assuming the displacement of the airfoils between two consecutive stages to be a constant 0.02 m rather than a time-dependent velocity of the airfoils in the y -direction, I leave out the third order of speed and essentially calculate the sum of lift rather than energy. As the air density, wing area, shape of airfoil, and angle of attack remain constant for different wind speeds and separations of the airfoils, the lift-coefficient equation, Eq. (18), predicts a quadratic relationship between the wind speed and lift, which is consistent with our curve in Fig. 27. Thus it is more reasonable to conclude that, instead of proving the DWG has superior energy output than a wind turbine, I prove that DWG has higher lift coefficient than a wind turbine. Consequently, I exempt myself from explaining the previous inconsistency that my model does not show a wind turbine will outperform the DWG at high wind speeds, as I never compute the energy output in a real-world dynamic process.

4.3 Concluding remarks

So far I have finally completed my quasi-static energy analysis of the dual-wing generator (DWG), if I ignore the impact of reducing a dynamic process into a static process. To estimate the quasi-static energy output of my DWG, I first find the pressure on the two airfoils pushed apart and together by wind in different stages of the DWG cycle based on a 2D turbulent simulation in COMSOL. With the pressure values, I conduct a surface integral to compute the lift for each stage. Assuming the airfoils travel a constant separation between each stage, I sum up the product of lift and separation to result in the quasi-static energy output. Meanwhile, I do the same for a single airfoil from

the DWG to compare the results. I use the single airfoil as a proxy for the turbine blades in a traditional wind turbine, which is a considerable simplification admittedly. However, I do find that the dual-wing configuration produces higher lift than a single airfoil during the process where the two airfoils are tilted towards each other. The additional lift comes from the suction effect when the two airfoils are close to each other, and consequently results in a higher lift coefficient for the DWG than a comparable wind turbine in theory. Notice this result applies to all different wind speeds and becomes more striking at large wind speeds. It will nevertheless be wishful thinking to extend the same quasi-static result for lift to energy output, because my model does not account for how the DWG airfoils move with respect to the wind, which can significantly change the results. Therefore, I have to conclude that, despite that I verify the contribution of suction effect to a higher lift and energy output produced by the DWG from a quasi-static process, regrettably, I fail to verify the relationship between the wind speed and the energy output of the DWG in a dynamic process presented by Festo in Fig. 3.

Nevertheless, I want to reassure readers that this project is meaningful. For the Physics Department, Middlebury College, I hope this project sparks interest in the continual study in fluid dynamics and bio-inspired technology, motivates potential projects to upgrade my stationary model to a time-dependent model in order to calculate energy, and maybe even persuades the faculty to teach a course in fluid dynamics for physics majors.

For myself, not only do I get to re-experience the beauty of Newton's second, third law, the conservation of mass, momentum, and energy, but also do I discover the beauty of partial differential equations, especially the beauty of the beastly Navier-Stokes equations; the beauty of fluid dynamics and the lovely physics behind it; the beauty of computer simulation, which is both time-wise and money-wise economic; the beauty of modeling in COMSOL, which allows you to do crazy things like signing up a creative writing class to write a fiction, or like constructing a new world from scratch so that you can destroy it in *Inception*; and the beauty of Festo's biomimetic robotics; the beauty of applying natural principles to an artificial machinery; the beauty of human imagination; the beauty of Sisyphus rolling the rock; the beauty of our life; and last but not least, the beauty of flight—so simple, elegant, and lifting.

At the end of this memorable journey I wish not to end, I want to give my most sincere and wholehearted thanks to:

Professor **Richard Wolfson**, my advisor on this project, whose technical and personal support to this project is like Newton's third law to my airfoils. I thank him for his insights, encouragements, and immense patience and tolerance of that I keep calling this document a thesis, mistakenly of course, while he keeps calling it a project;

Professor **Susan Watson**, whose encouraging smiles and wonderful leadership of both my thesis discussion and my experimental physics lab always make them a fun part of and a significant contribution to my project, which compels me to attend her sessions, despite my human weakness of wanting to skip them;

Professor **Anne Goodsell**, whose detailed-oriented explanation of impossible concepts and motherly consideration for her imperfect students always help me improve—fingers crossed—my scientific writing, which is crucial to the fact that you are reading this document in English;

and my friends: Liushao, who rescued a draft of this document from my trash bin, Jiguang, who cooked instant noodles at night, and Xingzi, who corrected my grammar, for their generous friendship and kind support;

and countless epic music soundtracks which help me stay awake when I should be in bed, especially *Two Steps From Hell - Compass* (feat. Merethe Soltvedt);

and last but not least, my parents, for their unconditional support for my prolonged undergraduate education, without which this project is a mere impossibility.

最后，我最诚挚的敬意致给我的父母。感谢你们对我疯狂人生的无条件支持。因为你们，一切成为可能。

References:

- [1] Festo, “Dual-Wing Generator”,
https://www.festo.com/net/SupportPortal/Files/333988/Festo_DualWingGenerator_en.pdf.
- [2] Streeter, Victor L, E. Benjamin Wylie, and Keith W Bedford. *Fluid Mechanics*. 1st ed. Boston: WCB/McGraw Hill, 1998.
- [3] NASA, “Similarity Parameters”,
<https://www.grc.nasa.gov/www/k12/airplane/airsim.html>.
- [4] White, Frank M. *Viscous Fluid Flow*. 1st ed. New York: McGraw-Hill, 1974.
- [5] Clancy, L. J. (1975). *Aerodynamics*. London: Pitman Publishing.
- [6] COMSOL, “Which Turbulence Model Should I Choose for My CFD Application?”,
<https://www.comsol.com/blogs/which-turbulence-model-should-choose-cfd-application/>.
- [7] NASA, “Boundary Layer”,
<https://www.grc.nasa.gov/www/k12/airplane/boundlay.html>.
- [8] Kapania, Nitin, Katherine Terracciano, and Shannon Taylor. "Modeling The Fluid Flow Around Airfoils Using Conformal Mapping". *SIAM Undergraduate Research Online* 1.2 (2008): 70-99.
- [9] Nave, R. “Bernoulli or Newton's Laws for Lift?”,
<http://hyperphysics.phyastr.gsu.edu/hbase/Fluids/airfoil.html>.
- [10] NASA, “Skipping stone theory”,
<https://www.grc.nasa.gov/www/k12/airplane/wrong2.html>.
- [11] NASA, “Longer path theory”,
<https://www.grc.nasa.gov/www/k12/airplane/wrong1.html>.
- [12] “Flow around an airfoil”, http://www.diam.unige.it/~irro/profilo1a_e.html.
- [13] “Timelines animation”,
[https://en.wikipedia.org/wiki/Lift_\(force\)#/media/File:Karman_trefftz.gif](https://en.wikipedia.org/wiki/Lift_(force)#/media/File:Karman_trefftz.gif).
- [14] Wolfson, Richard. *Essential University Physics*. 3rd ed. Pearson, 2016.

- [15] Babinsky, Holger. "How do Wings Work?" *Physics Education*, vol. 38, no. 6, 2003., pp. 497-503.
- [16] "Streamtube", <http://galileo.phys.virginia.edu/classes/311/notes/fluids1/node7.html>.
- [17] NASA, "Conservation of mass", <https://www.grc.nasa.gov/www/k12/airplane/mass.html>.
- [18] NASA, "Bernoulli's equation", <https://www.grc.nasa.gov/www/k12/airplane/bern.html>.
- [19] Cengel, Yunus A. *Thermodynamics An Engineering Approach*. 8th ed. McGraw-Hill, 2014.
- [20] NASA, "conservation of momentum", <https://www.grc.nasa.gov/www/k12/airplane/conmo.html>.
- [21] NASA, "Euler equations" <https://www.grc.nasa.gov/www/k12/airplane/eulereqs.html>.
- [22] COMSOL, "Navier-Stokes Equations", *Multiphysics Cyclopedia*. <https://www.comsol.com/multiphysics/navier-stokes-equations>.
- [23] Granger, Robert Alan and Robert A Granger. *Fluid Mechanics*. 1st ed. New York: Dover, 1995.
- [24] Ragheb, M. "Fluid Mechanics, Euler and Bernoulli Equations", <http://mragheb.com/NPRE%20475%20Wind%20Power%20Systems/Fluid%20Mechanics%20Euler%20and%20Bernoulli%20Equations.pdf>
- [25] NASA, "Lift coefficient", <https://www.grc.nasa.gov/www/k-12/airplane/liftco.html>.
- [26] NASA, "pressure force", <https://www.grc.nasa.gov/www/k-12/airplane/presar.html>.
- [27] Friedrichs, Drew. "A COMSOL Fluid Dynamics Evaluation of the R/V David Folger". Middlebury College, 2015.
- [28] Drake, "Finite-element method and the solar corona: energy storage in quadrupolar magnetic fields", Middlebury College, 2008.
- [29] COMSOL, "Flow Around an Inclined NACA 0012 Airfoil", *COMSOL Model Application Gallery*. https://www.comsol.com/model/download/318931/models.cfd.naca0012_airfoil.pdf

[30] “Betz’s law”,
<http://mragheb.com/NPRE%20475%20Wind%20Power%20Systems/Wind%20Energy%20Conversion%20Theory%20Betz%20Equation..pdf>.

[31] COMSOL, “COMSOL Multiphysics CFD Module Users Guide”. 2015. p.71.

[32] COMSOL, “Transitional Zone”, <https://www.comsol.com/blogs/which-turbulence-model-should-choose-cfd-application/>.

[33] COMSOL, “Viscosity Ramping”, <https://www.comsol.com/blogs/viscosity-ramping-improves-the-convergence-of-cfd-models/>.

[34] COMSOL, “Which turbulence model”, <https://www.comsol.com/blogs/which-turbulence-model-should-choose-cfd-application/>.