A Survey of Free Software for the Design, Analysis, Modelling, and Simulation of an Unmanned Aerial Vehicle

Tomáš Vogeltanz
Department of Informatics and Artificial Intelligence
Tomas Bata University in Zlín, Faculty of Applied Informatics
nám. T.G. Masaryka 5555, 760 01 Zlín, CZECH REPUBLIC
vogeltanz@fai.utb.cz

***Abstract*—The objective of this paper is to analyze free software for the design, analysis, modelling, and simulation of an unmanned aerial vehicle (UAV). Free software is the best choice when the reduction of production costs is necessary; nevertheless, the quality of free software may vary. This paper probably does not include all of the free software, but tries to describe or mention at least the most interesting programs. The first part of this paper summarizes the essential knowledge about UAVs, including the fundamentals of flight mechanics and aerodynamics, and the structure of a UAV system. The second section generally explains the modelling and simulation of a UAV. In the main section, more than 50 free programs for the design, analysis, modelling, and simulation of a UAV are described. Although the selection of the free software has been focused on small subsonic UAVs, the software can also be used for other categories of aircraft in some cases; e.g. for MAVs and large gliders. The applications with an historical importance are also included. Finally, the results of the analysis are evaluated and discussed - a block diagram of the free software is presented, possible connections between the programs are outlined, and future improvements of the free software are suggested.**

Keywords—aerodynamics; aircraft; analysis; CAD; calculator; CFD; design; development; flight dynamics model; ESOTEC software; FlightGear; free software; JavaFoil; JavaProp; JSBSim; MIT software; mechanics; modelling; OpenEaagles; OpenVSP; Public Domain Aeronautical Software; simulation; UAV; Unmanned Aerial Vehicle

I. INTRODUCTION

UAVs are a relatively inexpensive alternative to manned aircraft for a variety of applications, including aerial reconnaissance, environmental monitoring, agriculture, surveying, defense, search and rescue, and detection of biological, chemical, or nuclear materials. [1] [2] [3] [33] [34] [36] [28] [24] [45] [46] [49]

UAVs have a large number of important advantages. First of all, errors arising from the human element are minimized. This is of great significance in terms of reducing crashes; moreover, UAVs can be produced in smaller sizes which contribute to their high performance maneuverability, wide range of use, ease of control and command. [35] [50] [52]

The majority of missions are ideally suited to small UAVs which are either remotely piloted or autonomous. Requirements for a typical low-altitude small UAV include long flight duration at speeds between 20 and 100 km/h, cruise altitudes from 3 to 300 m, light weight, and all-weather capabilities. Although the definition of small UAVs is somewhat arbitrary, vehicles with wingspans less than approximately 6 m and weight less than 25 kg are usually considered in this category. [28]

Because of the availability of very small sensors, video cameras, and control hardware, aerial systems as small as 15 cm with a mass of 80 g are possible to use for some limited missions. These systems are referred to as micro aerial vehicles (MAVs). [28] [24]

An extremely small (less than 15 cm), ultra-lightweight (less than 20 g) aerial vehicle systems, with the potential to perform indoor and outdoor missions, are known as nano aerial vehicles (NAVs). [24]

Although the development and flight-testing of aircraft are well-documented engineering procedures, every UAV design, construction, implementation and test are unique and present different challenges to engineers, operators, and test teams. Because the performance of a UAV is dependent on both effective and highly responsive motor control as well as on aerodynamic efficiency, the high quality of the design and control of a UAV is increasingly required nowadays. Moreover, the criteria for UAVs may differ from those for manned aircraft; for example, the operation times of a UAV can be up to 10 times higher than a manned air vehicle, hence operation times of UAVs should be well considered. In addition, two important design parameters determine the power requirement of a UAV; range (based on the lift to drag ratio with fuel efficiency coefficient) and weight (based on total mass). [2] [4] [5] [36] [35] [30] [38] [46]

There are three basic phases in aircraft design: conceptual, preliminary, and detailed. Each design phase has characteristics which drive the tools and methods used as the design

This paper is loosely based on [44].

progresses. While it may be desirable to have the same suite of tools and methods spanning the design process, the widely varying characteristics of each of the phases makes this extremely difficult. For those organizations whose activities span all of these phases, there is a strong desire to have tools and methods which also span all of the phases, and this is most evident in the area of geometry definition. [177] [180]

II. Fundamentals of Flight Mechanics and Aerodynamics

Flight mechanics is the application of Newton's laws (1) and (2) to the study of vehicle trajectories (performance), stability, and aerodynamic control. [6] [3]

$$F = m \cdot a \tag{1}$$

$$M = I \cdot \alpha \tag{2}$$

The equations of motion are composed of translational (force) equations (1) and rotational (moment) equations (2) and are called the six degree of freedom (6DOF) equations of motion. The 6DOF means that aircraft can move in three dimensions in space and can rotate around three axes. Motion caused by gravity, propulsion, and aerodynamic forces contribute to the forces and moments which act upon the body of the airplane. Fig. 1 shows the three axes and the forces and moments acting on an aircraft. The center of gravity of the aircraft is at the intersection of the axes. [1] [2] [3] [6] [7]

For trajectory analysis (performance), the translational equations are uncoupled from the rotational equations by assuming that the airplane rotational rates are small and that control surface deflections do not affect forces. The translational equations are referred to as the three degree of freedom (3DOF) equations of motion. [6] [7]

A. Forces and Moments

The forces of lift, weight, drag, thrust, and side act along the axes and they force the aircraft to move in the axis direction. On the other hand, the three moments, yaw, roll, and pitch force the aircraft to turn around the axes. [1] [2] [3] [51]

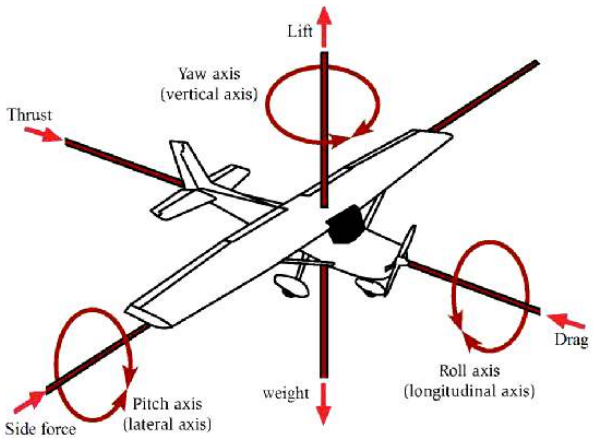

Fig. 1. The forces and moments acting on an aircraft [3]

Table I. defines each of the state variables. Although the forces and moments are relative to the atmosphere, the state variables are defined relative to the earth. [2]

TABLE I. State variable definition [2]

Variable	*Symbol*
Roll Rate (rad / sec)	P
Pitch Rate (rad / sec)	Q
Yaw Rate (rad / sec)	R
Velocity (m / sec)	V
Sideslip Angle (rad)	β
Angle of Attack (rad)	α
Bank Angle (rad)	μ
Flight-Path Angle (rad)	γ
Heading Angle (rad)	χ
North Position (m)	ξ
East Position (m)	η
Altitude (m)	h

V, χ, and γ represent the magnitude of the velocity vector, heading angle, and flight path angle. P, Q, and R represent the components of angular velocity; roll, pitch, and yaw. The position of the aircraft relative to the earth in Cartesian coordinates is expressed by ξ, η, and h. Body attitude relative to the velocity vector are μ, β, and α. [2]

All longitudinal motion occurs in the xz-plane of the aircraft. Stability along the longitudinal axis is both static and dynamic. Longitudinal static stability is the tendency of the airplane to return to pitch equilibrium following an angle of attack disturbance. Static stability is the aircraft's initial response to an input command. Aircraft is considered as statically stable when it immediately tends to return to its steady level flight condition. The aircraft is statically stable if the center of gravity is located at the wing aerodynamic center. When viewed over time, the aircraft is dynamically stable if it tends to return to steady level flight condition. [2] [6] [7]

B. Airfoils and Reynolds Number

The design of efficient airfoils and wings is critical. The proper functioning of the airfoil is the prerequisite to the satisfactory performance of the lifting surface itself. [24] [29] [28]

Accurate prediction of airfoil performance is especially necessary in the design of efficient low-speed airfoils. Parameters such as a wingspan and a chord length of the aircraft should be sized according to the shape and density of the structural model. The ideal shape of an airfoil depends profoundly upon the size and speed of the wing of which it is the core; in other words, different sizes of airfoils require different shapes. This dependence is called scale effect. [28] [31] [29] [38] [5]

The scale effect relates to the phenomenon that an airfoil which has most excellent qualities on an insect or bird may not exhibit these advantages when scaled up for an airplane wing, and vice versa. The scale effect is characterized by the chord Reynolds number (*Re*) defined by equation (3), where V is the flight speed, c is the mean wing chord, and v is the kinematic viscosity of the fluid in which the airfoil is operating. A Reynolds number calculator can be found in [98]. [29] [28] [6] [38] [5]

$$\mathrm{Re} = \frac{V \cdot c}{v} \tag{3}$$

The Reynolds number quantifies the relative importance of the inertial (fluid momentum) effects on the airfoil behavior, compared with the viscous (fluid stickiness) effects. It is the latter effects that essentially control the airfoil performance since they dictate the drag or stream-wise resistance as well as limiting and controlling the maximum lift of the airfoil. Normally, these qualities are described by the lift and drag coefficients, C_L and C_D defined as (4) and (5), where L and D are the lift and drag per unit span, q is the flow dynamic pressure, and c is the wing chord. The lift and drag coefficients depend on the Reynolds number as well as on the angle of attack of the airfoil which represents its geometric inclination to the incoming flow. [29] [6]

$$C_L = \frac{L}{q \cdot c} \tag{4}$$

$$C_D = \frac{D}{q \cdot c} \tag{5}$$

The chord length and the wingspan are based on the aircraft's mass and payloads weight; for instance, increasing payloads also increases the mass of the aircraft where Reynolds number rises as well as the wing-span. [38]

The combination of small length scale and low velocities results in a flight regime with low wing chord Reynolds numbers (i.e., chord Reynolds numbers ranging from 10,000 to 500,000). [28] [29]

The aerodynamics of fixed-wing UAVs is critically dependent on the Reynolds number and aspect ratio of the wing. Existing airfoil design methods produce good results down to Reynolds numbers of 200,000. When the aspect ratio decreases below 1.5, the nonlinear lift from the tip vortices dominates, especially at high angles of attack. For this reason, MAVs tend to cruise at higher angles of attack than higher aspect ratio vehicles. [28]

Fig. 2 shows this huge scale range, which spans the Reynolds numbers from 10^2 to 10^9. Below the lower limit, viscous effects are dominant and it is unlikely that any airfoil-like performance can occur. [29]

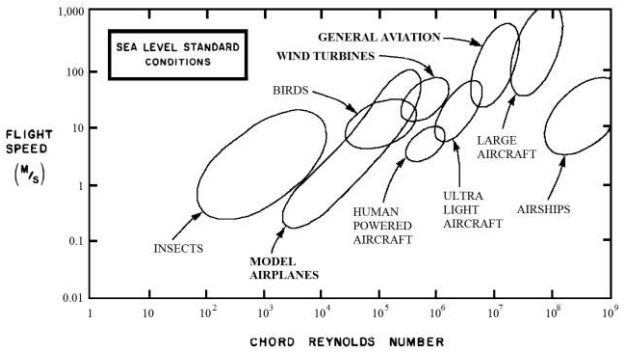

Fig. 2. Flight Reynolds number spectrum [29]

The insects and NAVs are in the range up to 10^4, where the flow is characteristically strongly and persistently laminar. The range 30,000 ≤ *Re* ≤ 70,000 is of great interest to MAV designers. The range of flying animals, MAVs, and small UAVs is at somewhat higher Reynolds numbers up to 10^5. [29] [28] [24] [53]

In regime up to about 10^6, the airfoil performance improves significantly because the parasite drag decreases. There is also the coexistence of a number of fascinating flight systems to support this claim; for instance, large soaring birds of quite remarkable performance, some small UAVs, foot-launched ultralight, man-carrying hang-gliders, the human-powered aircraft, and also the airfoils for small modern wind turbines. Sailplanes, light aircraft, and jet transports operate at Reynolds numbers up to and beyond 10^7. [29] [28] [53]

A convenient parameter to measure the effectiveness of an airfoil (also known as aerodynamics efficiency) is its lift-to-drag ratio (C_L/C_D); the maximum value of this quantity gives a good indication of the airfoil effectiveness. For design purposes, it is desirable that this maximum occur at a high lift coefficient so that the physical size of the lifting surface is minimized. [29] [24] [53]

At the lower Reynolds number values the viscous effects are relatively large, causing high drags and limiting the maximum lift, while at the higher values the lift-to-drag ratio improves. There is a critical Reynolds number of ca. 70,000 at which this performance transition takes place. This dramatic increase can be seen the most intensively in Fig. 3. [29] [24]

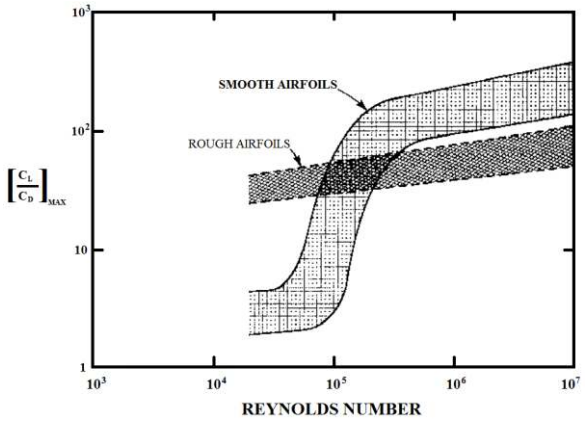

Fig. 3. Low Reynolds number airfoil performance [29]

The striking change in performance for smooth airfoils is near the critical Reynolds number where the lift-to-drag ratio increases more than an order of magnitude. It is of great interest that a rough or turbulent airfoil does not exhibit this abrupt performance change with Reynolds number. It is important to know that this critical Reynolds number divides the airfoils of the insect class (less than 10^4) from those of the large airplane class (above 10^6) (as illustrated in Fig. 2). [29] [53]

Some representative airfoil sections of this transitional range are shown in Fig. 4. At the low end, there are the insects with the interesting feature that it is not necessary to have a smooth surface; in fact, it is likely that the discontinuities are desirable to delay flow separation. For birds, however, smoothness begins to be important, as shown by the pigeon section. In the middle range is the Eppler 193, an airfoil with excellent performance at a Reynolds number of about 10^5, and at the high end, the Lissaman 7769, the airfoil used on the Gossamer Condor and Albatross, and the Liebeck L 1003, an airfoil of striking performance which provided clues on which the design of the Lissaman 7769 was based. [29]

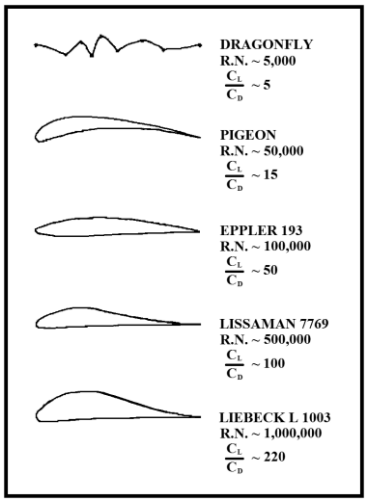

Fig. 4. Representative low-Reynolds-number airfoils [29]

1) Rotary Wing Airfoil

There are two main types of the rotor blade: symmetrical and asymmetrical, as illustrated in Fig. 5. [41]

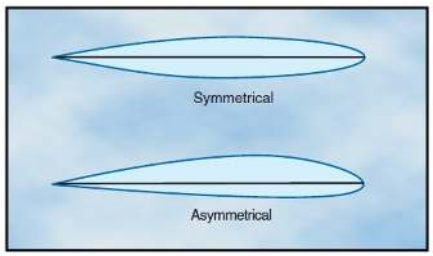

Fig. 5. Symmetrical and asymmetrical airfoils [41]

Symmetrical blades are very stable, which helps keep blade twisting and flight control loads to a minimum. This stability is achieved by keeping the center of pressure virtually unchanged as the angle of attack changes. [41]

Asymmetrical airfoils normally would not be as stable, but this can be corrected by bending the trailing edge to produce the same characteristics as symmetrical airfoils; this correction is called "reflexing". Using asymmetrical type of rotor blade allows the rotor system to operate at higher forward speeds. The advantages are more lift production at a given angle of attack than a symmetrical design, an improved lift-to-drag ratio, and better stall characteristics. The disadvantages are greater production costs and center of pressure travel of up to 20% of the chord line (creating undesirable torque on the airfoil structure). [42]

C. Weight and Power

During the design process of a UAV, both the weight budget and the power budget should be carefully monitored. In particular, the total mass of the vehicle should be kept as low as possible, since added weight will increase power consumption. The minimum power required to keep a fixed-wing aircraft in level flight can be expressed as (6), where T is the thrust, V is the velocity, η_p is the propeller efficiency, m is the aircraft mass, g is the gravitational variable, S is the wing area, ρ is the density of fluid, L/D is lift-to-drag ratio, and C_L is the lift coefficient. This means that doubling the weight nearly triples the power consumption. [24] [244]

Similarly, for hovering flight, the power requirement is expressed as (7), where M is the figure of merit of the rotor and V_h is the induced velocity in hover. Similar to that described above, a doubling of the weight increases the required power by a factor of nearly 3. [24] [244]

$$P = \frac{T \cdot V}{\eta_{\mathrm{p}}} = \frac{m \cdot g}{L/D} \frac{\sqrt{\frac{2 \cdot m \cdot g}{S \cdot \rho \cdot C_L}}}{\eta_{\mathrm{p}}} \tag{6}$$

$$P = \frac{T \cdot V_h}{M} = \frac{m \cdot g}{M} \sqrt{\frac{m \cdot g}{2 \cdot S \cdot \rho}} \tag{7}$$

III. The Structure of a UAV System

The study of structure, composition and function of the UAV system (UAS) is the premise. Any UAV system depends on its mission and range; however, most UAV systems include: airframe (physical and material structures) and propulsion systems, control systems, sensors for information collection, launch and recovery systems, communication links to get collected information from the UAV and send commands to it, and a ground control station. The typical UAV system is shown as Fig. 6. It is obvious that the health state of the whole system is dependent on the composed sub-systems. [3] [5] [34] [43] [38] [45] [48] [49]

UAVs can also require additional sensors to avoid obstacles, i.e. power lines, birds, trees, buildings and other barriers. These types of avoidance sensors are called see-and-avoid or sense-and-avoid. [38] [49]

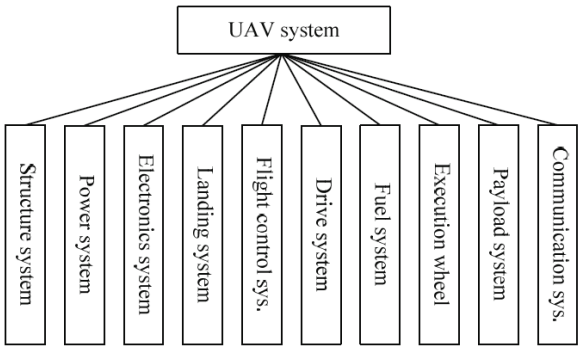

Fig. 6. The composition of a UAV system [43]

Although human personnel are part of the overall system, UAV systems include different levels of autonomy, ranging from remote control to fully automated mission completion including adaptation and decision making in response to changing operational conditions. [25] [46]

A. Airframe

Fig. 7 shows the main components of fixed-wing aircraft. The airframe consists of fuselage, wings, horizontal stabilizer, elevator, vertical stabilizer, and rudder. The weight of the airframe does not only come from the wings (pylon, flaps, and ailerons), fuselage, empennage, and nacelle but also from propulsion, avionics, sensors, and other payloads. The payload of medium range light aircraft is usually about 40–50%. [3] [38] [5]

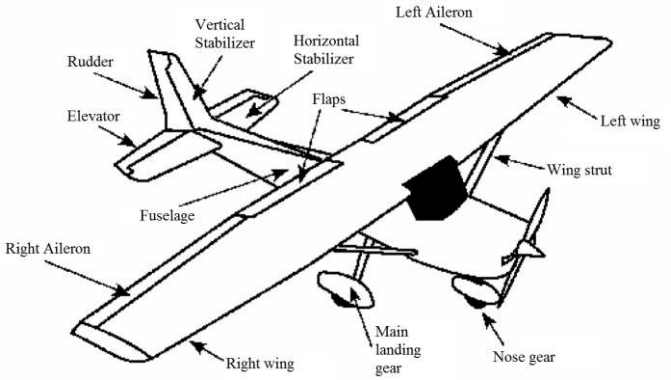

Fig. 7. Fixed-wing aircraft components [3]

The elevator is used to control the pitch of the aircraft. The rudder is used to control the yaw of the aircraft. The ailerons, which are movable surfaces on the outer trailing edge of the wings, are used to control the roll of the aircraft. The flaps, which are hinged parts on the inner trailing edge of the wings, are used to produce higher lift at low speed, and to increase drag on landing to get the required landing speed and approach angle. [3]

The landing gears configurations are either tricycle landing gear, which has the main landing gears just behind the aircraft center of gravity and a steerable nose gear, or tail dragger, which has the main landing gear forward of the aircraft center of gravity and a small steerable gear at the tail. [3]

Carbon fiber composites are the main materials used for UAV airframes, because they have high strength/weight ratio and are easily accessible. [24]

B. Propulsion

The propulsion system of a UAV consists of the following elements: [37] [38]

- energy source; e.g. chemical fuels (fossil fuels, biofuels, and chemicals), electricity, solar energy (in conjunction with photovoltaic cells), hydrogen, methanol, and energy mechanics
- storage media; e.g. tanks, batteries, capacitors, and metal hydrides
- mechanical energy converter; e.g. internal combustion engine, and fuel cell + electric motor
- lift/thrust converter; e.g. rotor, fan, propeller, and jet engine

Lift/thrust conversion systems are closely linked to the type of aircraft (fixed wing, rotary, lighter than air, etc.). In addition, propulsion systems usually include power control, RPM control, heat management system, and an auxiliary electrical power generator. [37]

Three main types of propulsion systems exist for UAVs: [37]

- alternative thermal systems; where different thermodynamic cycles, fuel, or engine types can be used (e.g., spark-ignition reciprocating engines fuelled by gasoline)
- electrical systems; where the required power is obtained through an electric motor and the power is generated or stored in different ways
- hybrid systems; combining any of the systems listed above, even the same type (e.g. a combination of fuel cell and battery or Regenerative Fuel Cell Systems, RFC, which combine fuel cell, battery, and photovoltaic cells)

Currently, mainly MAVs and small UAVs are powered by batteries and electric motors due to their high efficiencies, reliability, and ease of control. Electric motors convert electricity into mechanical energy by moving a propeller, fan, or rotor. They have the advantage of being the quietest and having one of the lowest thermal signatures. [37] [24]

Since coreless motors are lighter and smaller than, for example, direct current (DC) iron-core motors, they are considered more suitable. In addition to the small dimensions and the low weight, another advantage of coreless motors is the lack of iron losses that are reflected in a higher efficiency. Furthermore, since the rotor is very light, it has a small inertia that allows extremely fast accelerations and decelerations. However, the lack of iron in the center reduces the motor heat dissipation. To avoid overheating and thermal problems, they are only used for small and low-power motors. [24]

Innovations exist in small propulsion; for example, there are mini-ducted fan, mini-turboprop, mini-gas turbines, and mini-internal combustion engines. [24] [5]

C. Energy Storage

Electrical energy is supplied by a battery, photovoltaic or fuel cell. Although they are undergoing continuous improvement, electricity demand comes not only from the engine, but also from the electronic circuits, sensors, actuators and communication systems; consequently, the endurance or speed is limited. However, a large part of the electric energy is used for the electric motors. [37] [24]

Since energy stored in batteries does not require any conversion to be useful for both the electronics and propulsion, batteries seem to be the most appropriate for an electric UAV. Furthermore, the energy density of the batteries has steadily increased during the last years as shown in Fig. 8. [24] [5]

The most advanced batteries (intelligent batteries) include circuit that optimizes the cells' discharge curves with respect to the loads. Despite these improvements, the most advanced batteries also provide much lower energy densities than sources, such as gasoline or methanol. [24] [5]

Another alternative is fuel cells. A fuel cell system is conceptually a sort of battery in which the fuel is transformed into electric current trough an electrochemical process. There are several kinds of fuel cells which mainly differ with respect to the principle of energy conversion. Currently, the most promising fuel cells for a UAV are proton exchange membrane (PEM) fuel cell and direct methanol fuel cell (DMFC) which could be considered as a subcategory of the PEM. [24] [37] [5]

Besides fuel cells, ultra-capacitors have become interesting the last few years. The latest improvements have made this power storage principle attractive also for UAVs, and they have already been used in some prototypes. [24]

Since they are an evolution of normal capacitors, their main features are fast charging, high peak current, and virtually unlimited charge-discharge cycles. The main drawback consists of the output voltage that strongly depends on the charge status of the capacitor. Moreover, when compared with other energy sources, they have a relative low energy density as can be seen in Fig. 8. [24]

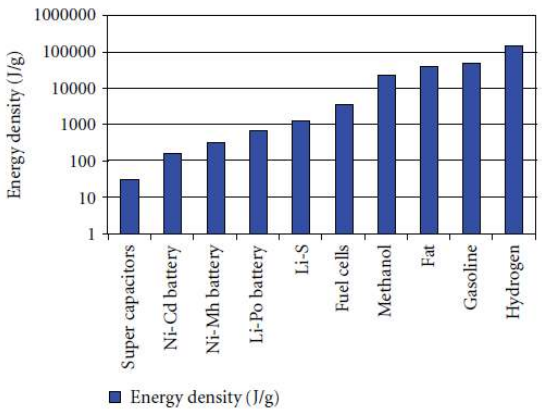

Fig. 8. Energy densities of various energy storage systems [24]

Although solar cells are also a potential useful energy source, photovoltaic systems cannot be used for every type of the UAV; for example, the small dimensions of NAVs, the weight constraints, the indoor application area (low light), and some UAV typologies limit the efficiency and the availability of this energy source. [24]

D. Transmission

For a UAV system, two different kinds of signals have to be transmitted: control signals and data signals. The control signals are needed for take-off, landing and for piloting the vehicle in general while the data signals are the data collected by onboard sensors of a UAV system (such as camera, microphones, gas sensors, and other devices). Control signals are mainly transmitted from the ground station to the vehicle while the data signals are sent from the vehicle to the ground station. [24] [5] [48]

Currently, the only system known to be operative is communication by radio, directly or via satellites or other means of radio relay. [5]

The specifications for communications performance include two fundamental parameters: [5]

- data rate; which is the amount of data transferred per second by a communications channel and is measured in bytes per second (Bps)
- bandwidth; which is the difference between the highest and lowest frequencies of a communications channel, i.e. the width of its allocated band of frequencies, and is measured in MHz or GHz as appropriate

When reducing the UAV system dimensions, the major challenges for the communications parts are represented by the weight and size of the antennas, filters, and resonators. Antenna shape strongly depends on the operating frequencies and, thus, will depend on external factors, such as application (military frequencies are different from civilian frequencies), distances, bit rate, and other factors. This requires the application specific antenna design. [24]

The loss of communication during operations may result from: [5]

- failure of all or part of the system due to lack of reliability
- loss of line-of-sight (LOS) due to geographic features blocking the signals
- weakening of received power due to the distance from the UAV to the control station becoming too great
- intentional or inadvertent jamming of the signals

UAV system should have an ability to safely complete/cancel a mission in case the communication is lost, e.g. the UAV can fly to the base or to the last position where the communication was in order. It also must be ensured that the transient loss of communication will not affect the UAV functions.

E. Sensors

Sensors can roughly be divided into two categories. The first one contains the sensors that are necessary for flight control, the second is sensors that are a part of the payload and provide mission-specific information. [24]

A UAV system should theoretically be able to fly only with a 3-D accelerometer and a 3-D gyroscope. Ideally, if the initial position is known, all the later positions can be calculated only by integrating the resulting vector acceleration two times to find the position, while 3-D gyroscope signal is used to maintain flight stability. However, since all gyroscopes and accelerometers suffer from offsets and drifts, for instance with time and temperature, the accuracy of the calculated position will decrease over time. [24]

The currently popular method of position fixing and navigation between points is by use of the Global Positioning System (GPS). GPS is available as two services, the Standard Positioning System (SPS) for civilian users and the Precise Positioning Service (PPS) for military users. Both signals are transmitted from all satellites. The accuracy of GPS position fixes varies with the receiver's position and the satellite geometry. Height is also available from GPS, but to a lower accuracy. [5]

The other class of sensors is the data-collecting sensors that provide useful information for the users. Examples are cameras, microphones, gas sensors, biological sensors, radiation sensors, and other sensors. [24]

IV. The Modelling and Simulation of a UAV

As known, flight modelling and simulation have many advantages; for instance, energy and finance saving, security, and no limitation of locality and weather. Moreover, some hard and risky tasks can be simulated. [34]

However, the accurate modelling and simulation of a UAV is not an easy task, due to the need to calculate many parameters either by physical measurements, experiments, or estimation from available data of similar UAV or by software tools. One of the big challenges is calculating aerodynamic coefficients. Aerodynamic coefficients characterize the response of the proposed vehicle based on its geometry. [1] [3]

UAVs typically consist of sets of sophisticated and different entities including several categories of human personnel. A comprehensive simulation environment must model all these components and include specific characteristics related to system intelligence, complexity, autonomous operation, and collaborative operation. [25]

Several major assumptions are often made for the modelling and simulation of the aircraft. First, the aircraft is rigid. Although aircraft are truly elastic in nature, modelling the flexibility of the UAV should not contribute significantly to the research. Second, the earth is an inertial reference frame. Third, aircraft mass properties are constant throughout the simulation. For UAV modelling, it can be assumed the aircraft has constant mass over a flight. Finally, the aircraft has a plane of symmetry. The first and third assumptions allow for the treatment of the aircraft as a point mass. This assumption is a satisfactory approximation for UAV models. [1] [2]

For the modelling and simulation of a UAV at least the following items must be created: [3] [34]

- A Flight Dynamics Model (FDM)
- A UAV mathematical model
- A 3D graphical model (only if the 3D visualization of the UAV is needed)
- A control system
- A flight route identification
- Autonomous flight simulation

FDM is the physics/math model which defines the movement of an aircraft under the forces and moments applied to it using the various control mechanisms and from the forces of nature. FDM includes development of a physical, inertial, and aerodynamic model representing the UAV. FDM processes parameters from all input information. By manipulating input variables mathematically, FDM predicts the future states of an aircraft. The FDM accuracy determines the fidelity of the simulator. [1] [2] [3] [34]

With a generic FDM implementation in mind, the Aerodynamic Coefficients are not provided by the FDM and hence need to be determined by other ways. As long as the aerodynamic coefficients are available, the FDM may model the motion of any vehicle configuration, from a ball to a transonic fighter. [1]

FDM, like any other dynamics model, is a data driven program. Hence the accuracy of its outputs depends on the quality of the input information supplied. FDM takes initial conditions of the vehicle, and other inputs including aircraft properties (e.g. inertia and gravity), aerodynamic coefficients, control inputs and relative wind conditions. After calculations, FDM sends the vehicle dynamic responses to the output. Fig. 9 illustrates the internal data flow of FDM. [1]

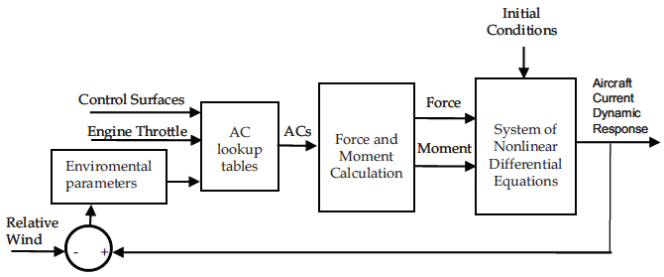

Fig. 9. The schema of a flight dynamics model [1]

As indicated, wind modelling has a significant role in the design and certification of aerial vehicles. It helps to study and analyze the behavior of the aircraft facing the wind. [40] [1]

The best equations to use to completely and accurately model a UAV's true motion are nonlinear fully coupled ordinary differential equations. With these equations of motion,

UAV response to any commanded inputs or wind disturbances is accurately modeled. [2] [33]

However, a software model developed from first principles has unknown accuracy. The accuracy of model is ensured through a verification and validation process. [1] [25] [26]

Verification is the process of determining that a model implementation accurately represents the developer's conceptual description of the model and the solution to the model. [26] [27]

Validation is the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model. [26] [27]

The 3D graphical model is necessary only if 3D visualization of the UAV is needed. 3D Visualization can give us a better view of the simulation than numbers and graphs alone. [3]

UAVs promise greater precision; however, autonomous flight and stability of the UAV depend on the control system. The ability to test control systems in a virtual environment is significant for development. A reliable UAV simulation process which can be adapted for different aircraft would provide a platform for developing control systems with reduced dependence on expensive field trials. In many cases, testing newly developed control systems in a virtual environment is the only way to guarantee absolute safety. [1] [2] [4] [34]

The way to identify flight route means to simulate system similar to the GPS or the GPS itself. [3]

Autonomous flight simulation depends on all things above; in addition, use cases and activity diagrams of simulation should be designed to define what should be found and how it should be found. Next, the settings of simulation parameters (e.g. density, gravity, airspeed, and altitude) must be defined and then the simulation itself can be run with or without visualization. [2] [3]

An example of the hierarchy of a UAV simulation system can be seen in Fig. 10. This UAV simulation system is divided into three layers based on function. In the simulation system, the objects on top layer have more responsibilities and manage the objects on low layer; for instance, they provide the instantiation of object, the calling of methods, communication and operation management. In contrast, the objects on low layer are focused more on functionality. [34]

The simulation system includes an environment model, an aircraft system model and an equipment model. Each subsystem model has the independent function; for example the aircraft system model is the core of the system and is formed of flying control model, engine model, kinematics model, sensor model, data channel model and other models. [34] [25]

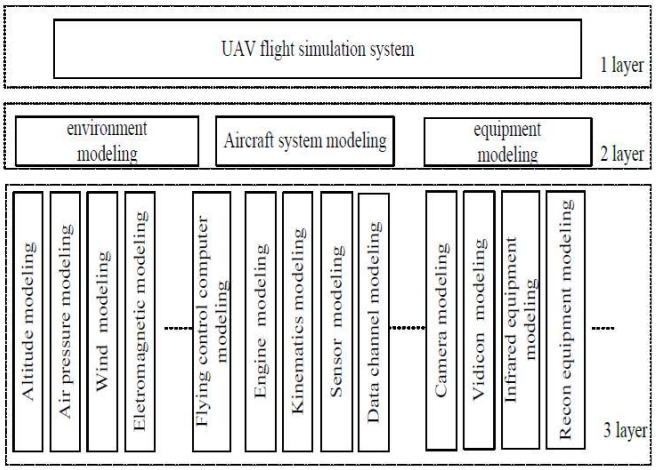

Fig. 10. The structure of a UAV simulation system [34]

V. FREE SOFTWARE

Modelling and simulation software has been developed to assist in the design, development, test, and validation of complex aircraft systems. The fidelity and precision of the software ensure the reliability and efficiency of flying simulation system and can decrease the time and costs needed to development of any UAV. [39] [34]

Simulation program can run in two modes: [34]

- real-time mode (used in real-time systems)
- script mode (used in test process)

A large amount of freeware and open-source software for the modelling and simulation of aircraft exists on the Internet. Fortunately, much of the software can also be used for UAVs. As noted, this section probably does not include all of the free software, but tries to mention, describe, or analyze the most interesting applications.

A. JSBSim Flight Dynamics Model

JSBSim is an open-source 6-DOF nonlinear flight dynamics model which is used as the default FDM for FlightGear Flight Simulator and for OpenEaagles Simulation Framework. JSBSim is generally considered as a very accurate. [1] [3] [12] [13] [14] [52] [55] [56]

The accuracy of JSBsim has been proved by a large number of studies; for example, it was tested on Cessna-182 UAV by University of Sheffield in the United Kingdom [3], and on Shadow UAV by Purdue University in Indiana, USA [58]. Moreover, a full 6-DOF simulator for flight simulation and pilot training was constructed at the University of Naples using JSBSim as its physics engine. JSBSim is also used to drive the motion-based research simulators in the Institute of Flight System Dynamics and Institute of Aeronautics and Astronautics at RWTH Aachen University in Germany. Furthermore, U.S. Department of Transportation developed a human pilot math model by using JSBSim as the 6-DOF simulation core. [1] [3] [12] [57] [52] [56]

JSBSim is written in the C++ programming language and can be run as a stand-alone application, or as an integrated part of the flight simulator which provides visual output. It also supports many data output formats such as socket and file. [3] [12] [13] [55] [56]

JSBSim can model aircraft, missile, rotorcraft, and lighter-than-air systems, and may take into consideration the rotational earth or wind effects on the equation of motion. Particular aircraft flight control systems, propulsion, aerodynamics, landing gears, and autopilot are defined in eXtensible Markup Language (XML) format files. [3] [12] [13] [59] [55] [56]

JSBSim allows algebraic functions (e.g. sum, random, average, difference, sinus, power and other operations) to be defined in configuration files. All currently supported operations are listed in [57].

Despite JSBSim uses imperial units (e.g. feet, pounds, etc.) for internal calculations almost exclusively, it is also possible to use international units (e.g. meters, kilograms, etc.). In fact, to avoid confusion, the unit should always be specified using the "unit" attribute as shown in Fig. 11. [57]

```
<wingspan unit="M">2</wingspan>
```

Fig. 11. An example of the unit attribute

1) Configuration Files and Aeromatic

JSBSim requires creating JSBSim aircraft configuration files to model and simulate an autonomous UAV; for instance, by using the Aeromatic. Aeromatic is a free PHP-based web application and is also included in source code of JSBSim. However, Aeromatic uses templates for the generation of the configuration file (e.g. the template for the glider, light twin, subsonic racer/aerobatic, etc.) and provides only rough values. As a result, the next step is to perform educated guesses to improve important sections in the created configuration files with the assistance of available data of a similar UAV or with the assistance of appropriate software tools. [3] [13]

Aeromatic takes input conditions from the user i.e. used units (imperial/international), a name, type, and length of the aircraft, maximum take-off weight, wingspan, wing area, a landing gear layout, a number of engines, an engine type, and an engine layout. Some values can be estimated by Aeromatic, e.g. wing chord, wing area, inertia, and other parameters. A part of Aeromatic application can be seen in Fig. 12. [3] [13]

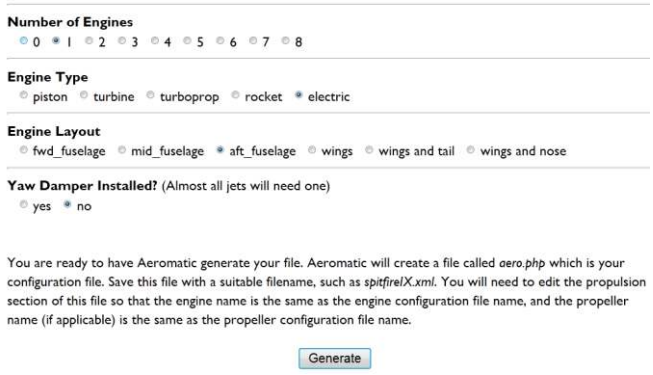

Fig. 12. The aircraft configuration file creation by using Aeromatic

For the test of Aeromatic, the glider type was used, the metric system of measurement, a maximum take-off weight of 2 kilograms, a wingspan of 2.5 meters, a length of 1 meter, one electric motor, and an after-fuselage engine layout. The other parameters were automatically estimated by Aeromatic, for example a wing chord was calculated as 0.0243 m, a wing area as 0.0585 m^2, and wing loading as 34.177 kg/m^2.

However, because the wing loading seemed to be extremely high, it was necessary to set the wing chord and wing area to higher values i.e. to 0.25 meters and 0.58 m^2. After this modification, the wing loading decreased to 3.467 kg/m^2 which should be acceptable. In conclude, the estimated values may not always be suitably calculated by Aeromatic; thus educated guesses must be performed.

Aircraft's metrics, airframe geometry, mass and inertia properties, landing gear positions and their ground reactions, flight control system, and aerodynamic characteristics are specified in the aircraft configuration file whose structure is shown in Fig. 13. [3] [57] [56]

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="http://jsbsim.sourceforge.net/JSBSim.xsl"?>
<fdm_config name="unnamed" version="2.0" release="ALPHA"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://jsbsim.sourceforge.net/JSBSim.xsd">

 <fileheader>
  <author> Aeromatic v 0.95 </author>
  <filecreationdate>2014-06-11</filecreationdate>
  <version>$Revision: 1.15 $</version>
  <description> Models a unnamed. </description>
 </fileheader>

 <metrics>
  ...
 </metrics>

 <mass_balance>
  ...
 </mass_balance>

 <ground_reactions>
 ...
 </ground_reactions>

 <propulsion>
  <engine file="unnamed_engine">
   ...
   <thruster file="direct">
    ...
   </thruster>
  </engine>
   ...
 </propulsion>

 <flight_control name="FCS: unnamed">
 ...
 </flight_control>

 <aerodynamics>
 ...
 </aerodynamics>

 <external_reactions>
 </external_reactions>

</fdm_config>
```

Fig. 13. The structure of the aircraft configuration file

The file-header section of the configuration file includes the author's name, date of creation, a version of the model, a license type, references, notes, and limitations. [57]

The metrics section of the configuration file defines the characteristic measurements of the vehicle and the locations of key points. A Vehicle Reference Point (VRP) is usually placed at the nose of the aircraft; X-axis is along the aircraft body (positive towards the tail), Y-axis is along the wings (positive towards the right wing tip), and Z-axis is in the vertical axis (positive downward). The VRP is an agreed upon point on the aircraft, for which the flight model will provide the latitude/longitude/altitude. The VRP is not so important to flight dynamics but it is very important to 3D visualization. It helps with the placement of the 3D aircraft model where the flight model exactly specifies aircraft components. [3] [57]

The mass-balance section specifies the empty weight of the aircraft, the moments and products of inertia, the location of the center of gravity, and definitions for any point masses that are included such as Payloads. [57]

Hot air balloons, buoyancy-assisted vehicles, and zeppelins can be modeled through the use of gas cells and ballonets. The buoyant-forces section has to be added for these types of systems. The type of gas contained in the gas cell can be 100% pure hydrogen, helium or air. The initial fullness fraction of the cell is normally in the interval (0-1); the fullness value greater than 1.0 initialize the cell at higher than ambient pressure. The maximum allowed cell overpressure with respect to the surrounding atmosphere, the capacity of the manual valve, and the heat flow from the atmosphere and surrounding environment into the gas cell can also be defined. However, automatic valves with limited capacity, cell failure from overpressure, more realistic center of buoyancy, and more realistic and complete inertia moment are still missing. [57] [59]

Ground reactions section specifies the wheels location, and the coefficients associated with each wheel. Contacts between the aircraft and the ground can be modelled so that the aircraft can realistically take-off and land. Contacts can also be used to model the interaction between the ground and any part of the aircraft structure such as the wing tip. JSBSim does not make any guess for the contact points location from the geometry data provided in the metrics section. It is advisable to have at least 3 unaligned contacts so that the aircraft can have a stable position when resting on the ground. [3] [57]

JSBSim can model two types of contacts: [57]

- BOGEY which is used for landing gears
- STRUCTURE which is used for any location on the aircraft other than the landing gears (typically wing tips, nose and tail)

Both of these contact types basically result in a force which resists the penetration of the ground by the aircraft. The main difference between the two types of contacts is how the ground reaction force is computed. Furthermore the BOGEY type includes features which are typical to landing gears such as brake and steering. [57]

The ground reactions are computed as forces that support the aircraft above the ground, and affect the motion over the ground; thus, these forces can be split into two components: [57]

- the ground normal reaction (computed by a spring/damper model)
- the ground tangential reaction (computed by the Coulomb friction law)

Aerodynamic forces and moments in JSBSim are defined in the aerodynamics section of the aircraft configuration file, or in a separate file. Within the aerodynamics section there are six axis sections corresponding to three translational and three rotational axes. Many factors affect each of the forces and moments. The total force or moment is the sum of the individual effect. [3] [57] [55] [56]

Optionally two other coordinate systems may be used: [57]

- body coordinate system (x, y, z)
- axial-normal coordinate system (axial, normal, side)

JSBSim can model externally or arbitrarily applied forces and moments. Such a capability might be needed to model a catapult, hook and wire capture device, tow rope, or parachute. External forces are defined in the external reactions section. [57]

JSBSim can model different types of engines i.e. electric, turbine, turboprop, piston, and rocket engines. Nowadays, there is no battery model available for electric engine; thus it does not consume any energy in simulations. [3] [13]

The aircraft's propulsion system is specified in two files, one for an engine and the other for a thruster. This technique allows researchers to assign different kinds of engines and thrusters to the aircraft. These files are referred to in the propulsion section of the aircraft configuration file. Additionally, other parameters (e.g. the location, orientation, and energy consumption) of the engine, thruster, and fuel tank are specified in this file. [3] [13] [55] [56] [58]

For the creation of an engine or propeller, Aeromatic takes the engine type, engine power or thrust, maximum engine RPM, pitch condition, and propeller diameter from the user while the orientations of engines and propellers have to be edited in file if it is needed. [3] [13] [58]

Aeromatic does not have the ability to create an electric engine; however, there is no problem to make the file manually because of small complexity. An example of the electric engine file is shown in Fig. 14.

```
<?xml version="1.0"?>

<electric_engine name="electric_2kW">
  <power unit="WATTS">2000.0</power>
</electric_engine>
```

Fig. 14. The configuration file of the 2 kW electric engine

It was found that Aeromatic is not appropriate for creating the propeller of small measurements because, for example, the rotational inertia (<ixx> element in file) of generated propeller

is almost always zero. The problem is probably in the rounding of the value to 2 decimal places. However, after appropriate editing, the generated file may be usable.

JSBSim provides components which can be connected together to model a flight control system for an aircraft. The control of the channels such as Pitch, Roll, Yaw, Flaps, Landing Gear, and Speed Brake can be performed in the flight control section. The flight control surfaces are elevator, right and left ailerons, and rudder. [3]

Mechanization of components can be defined by properties. Properties are like variables which are categorized into a tree structure, and accessible from the configuration file. Properties refer to various values within the simulation which represent such physical parameters as roll rate, air density, drag force, and others. [57]

Almost all control system components have some common features. Unless otherwise specified, the common elements are: input, output, delay, and clipto (the last one permits limiting of the output of a component). The possible components to use are: Filter, Switch, Sample and Hold, Sum (named as "summer"), Gain, Scheduled Gain, Aero-surface Scaling, Deadband, Limiter, Positive or Negative Value, Absolute Value, Kinematic, FCS (Flight Control System) Function, Actuator, Sensor, Translational Accelerometer, and PID (Proportional-Integral-Derivative) controller. [57]

JSBSim can be scripted to run automatically by using a script file in XML format. Commands are specified using the scripting directives for JSBSim. A test condition (or conditions) can be set in an event in a script and when the condition evaluates to true, the specified action (or actions) is taken. An event can be persistent, which means that at all times when the test condition evaluates to true the specified actions are executed. When the set of tests evaluates to true for a given condition, an item may be set to another value. This value may be a value, or a delta value, and the change from the current value to the new value can be either via a step function, a ramp, or an exponential approach. The speed of a ramp or approach is specified via the time constant. The basic structure of the script file is shown Fig. 15. [57] [55] [56]

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="http://jsbsim.sf.net/JSBSimScript.xsl"?>
<runscript xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="http://jsbsim.sf.net/JSBSimScript.xsd"
   name=".....">
  <description>.....</description>
  <use aircraft="....." initialize="....."/>
  <run start="0.0" end="100" dt="0.00833333">
    <event name=".....">
      <condition>.....</condition>
      <set name="....." value="....." action="....." type="....." tc="....."/>
    </event>
  </run>
</runscript>
```

Fig. 15. The basic structure of the script file

B. *FlightGear Flight Simulator*

FlightGear is an open-source flight simulator, written in the C++ programming language, to model and simulate a wide variety of aircraft and, reportedly, also a soft-wing and flapping-wing vehicles (ornithopters). FlightGear has probably the ability to model all these kind of vehicles and their hybrids. [1] [3] [14] [15] [60] [45] [47] [48] [49] [52] [58]

Data visualization is another aspect considered while building the flight dynamics model. FlightGear supports many different 3D file formats, for example VRML1, DXF and AC3D. AC3D is the standard used in most FlightGear models. FlightGear can produce a 3D graphic animation in real time and is connected to FDM. The animation facility allows the UAV to be viewed from any angles, and provides absolute visual information on the UAV attitude and stability. Moreover, it models real world instrument behavior, and system failures. [1] [3] [14] [15] [60] [62] [45] [47] [50] [58]

FlightGear allows the user to access the internal properties and monitor any of its internal state variables. By editing configuration files, it is possible to create sound effects, model animations, instrument animations and network protocols for nearly any situation. FlightGear can communicate with external flight dynamics models, GPS receivers, external autopilot, control modules, other instances of FlightGear, and other software. FlightGear could also be used for the simulation of multi-agent cooperation. [3] [15] [62] [47] [48] [49] [58]

FlightGear contains many special features, some of which are not obvious to the new user, e.g. aircraft carrier with launching of aircraft from a catapult, Atlas ("moving map" application), multiple displays, multiple computer, recording and playback, and air-air refueling. [60]

It is possible to choose between three primary FDMs: JSBSim (described in previous subchapter), YASim, and UIUC. It is also possible to add new dynamics models or even interface to external proprietary flight dynamics models. [15] [16] [60] [61] [62]

YASim is an integrated part of FlightGear and uses a different approach than JSBSim by simulating the effect of the airflow on the different parts of an aircraft. The advantage of this approach is that it is possible to perform the simulation based on geometry and mass information combined with more commonly available performance numbers for an aircraft. This allows for quickly constructing a plausibly behaving aircraft which matches published performance numbers without requiring all the traditional aerodynamic test data. [15] [60]

UIUC is based on LaRCsim originally written by the NASA. UIUC was initially geared toward modelling aircraft in icing conditions, but now encompasses nonlinear aerodynamics, which result in more realism in extreme attitudes, such as stall and high angle of attack flight. [15] [16] [60] [61]

FlightGear tries to replicate the real navigation system around the world; thus a flight path which contains a number of waypoints can be constructed. FlightGear uses fixed waypoints such as airports and navigation aids such as radio stations for navigation. In order to use the selected waypoints with FlightGear navigation system, a unique ID can be assigned to each waypoint, and the FlightGear database can be altered to include the new waypoints with their IDs. [3]

The fixed waypoints are determined by latitude and longitude. When a waypoint is entered in the aircraft route

during the simulation time, FlightGear checks the database to see if it is a valid fixed point or not. This database is stored in the compressed file called fix.dat which can be found in the directory FG_ROOT\ FlightGear\data\Navaids. This file can be edited by using, for example, Notepad++. [3]

FlightGear was tested, for example, on Cessna-182 UAV [3], Shadow UAV [58], Pioneer UAV [61], and ArduCopter [63].

1) Configuration Files

Except of FDM configuration files, other files are required for use with FlightGear which include the electric-system file, autopilot file, and 3D graphical model specification file. The final required file ties the previous files together. [3] [47]

The electrical-system file specifies the battery characteristics, the lights and other parameters. A part of an example of an electrical-system configuration file can be seen in Fig. 16. [3]

```
<?xml version="1.0"?>

<PropertyList>

 <!-- Supplier list -->

 <supplier>
  <name>Battery 1</name>
  <prop>/systems/electrical/suppliers/battery[0]</prop>
  <kind>battery</kind>
  <volts>36</volts>
  <amps>35</amps>
 </supplier>

 <supplier>
  <name>Alternator 1</name>
  <prop>/systems/electrical/suppliers/alternator[0]</prop>
  <kind>alternator</kind>
  <rpm-source>/engines/engine[0]/rpm</rpm-source>
  <volts>36</volts>
  <amps>35</amps>
 </supplier>
```

Fig. 16. A part of the electrical-system configuration file

To fly the modelled UAV autonomously, a tuning process should be made for the built-in PID autopilot which has the ability to hold aircraft velocity, vertical aircraft speed, altitude, pitch angle, angle of attack, bank angle, and true heading. [3]

FlightGear implements a PID algorithm in a flexible way which makes it reusable with similar aircraft. Typically a PID controller manipulates one control output to force a current value (or process value) towards a target value (or reference point). However, any number of PID controllers can be defined in the autopilot configuration file. Moreover, a process value, reference point, any number of output values, and other tuning constants can be assigned to each controller. Cascading controllers can be implemented by specifying multiple PID controllers in which the output of the current stage is used as the input to the next stage. [3]

To construct an autopilot configuration file (whose example can be seen in Fig. 17) for a modelled aircraft, an suitable idea may be to copy an autopilot configuration file from an existing, similar aircraft, and tuning the autopilot parameters to adapt to the modelled aircraft. The most basic method of tuning autopilot parameters is the trial and error method. In this method, the proportional gain, integral time, and derivative time are adjusted until the performance is acceptable. [3]

```
<?xml version="1.0"?>

<!-- Generic Autopilot Configuration -->
<!-- Each component is evaluated in the order specified.  You can make up -->
<!-- property names to pass the result of one component on to a subsequent -->
<!-- component. -->

<PropertyList>
 <!-- ================================================================== -->
 <!-- Roll Axis Modes                                                    -->
 <!-- ================================================================== -->
 <!-- Wing leveler -->
 <pid-controller>
  <name>Wing Leveler (Turn Coordinator based)</name>
  <debug>false</debug>
  <enable>
   <prop>/autopilot/locks/heading</prop>
   <value>wing-leveler</value>
  </enable>
  <input>
   <prop>/orientation/roll-deg</prop>
  </input>
  <reference>
   <value>0.0</value>
  </reference>
  <output>
   <prop>/controls/flight/aileron</prop>
  </output>
  <config>
   <Kp>0.05</Kp>     <!-- proportional gain -->
   <beta>1.0</beta>   <!-- input value weighing factor -->
   <alpha>0.1</alpha>  <!-- low pass filter weighing factor -->
   <gamma>0.0</gamma>  <!-- input value weighing factor for -->
```

Fig. 17. A part of the autopilot configuration file

In order to perform a visual simulation, a 3D graphical model should be specified. The animated control surfaces and their kind of animation are specified in a graphical-model configuration file as shown in Fig. 18. [3] [47] [62]

```
<?xml version="1.0"?>

<PropertyList>

 <path>Rascal110-000-013.ac</path>

 <model>
  <path>Aircraft/Rascal/Models/smokeW.xml</path>
  <offsets>
   <x-m> 2.0</x-m>
   <y-m> 0.0</y-m>
   <z-m> 0.0</z-m>
   <roll-deg>  0</roll-deg>
   <pitch-deg>  0</pitch-deg>
   <heading-deg>0</heading-deg>
  </offsets>
 </model>

 <animation>
  <type>rotate</type>
  <object-name>L_Aileron</object-name>
  <property>/surface-positions/left-aileron-pos-norm</property>
  <factor>20.0</factor> <!-- fixme -->
  <center>
   <x-m>0.735</x-m>
   <y-m>-0.450</y-m>
   <z-m>0.139</z-m>
  </center>
  <axis>
   <x>0.037</x>
   <y>1.0</y>
   <z>-0.029</z>
  </axis>
 </animation>
```

Fig. 18. A part of the graphical-model configuration file

The set configuration file ties all the previous files together by specifying their names and paths and is the first processed file in the simulation. Fig. 19 illustrates an example of the set configuration file. [3] [50] [62]

```
<?xml version="1.0"?>
<!--
************************************************************
Rascal 110 R/C airplane config. This files ties together all the components
used by FGFS to represent the Rascal 110 (by Sig Mfg) including the flight
dynamics model, and external 3D model.
************************************************************
-->

<PropertyList>
 <sim>

  <description>Rascal 110 (R/C)</description>
  <author>Lee Elliot (3D) Dave Culp (JSBsim dynamics) and Curt Olson</author>
  <aircraft-version>0.1</aircraft-version>
  <model-hz>400</model-hz>

  <startup>
   <splash-texture>Aircraft/Rascal/Rascal110-splash.rgb</splash-texture>
  </startup>

  <flight-model>jsb</flight-model>
  <aero>Rascal110-JSBSim</aero>
  <fuel-fraction>0.8</fuel-fraction>

  <systems>
   <autopilot>
    <path>Aircraft/Rascal/Systems/110-autopilot.xml</path>
   </autopilot>
   <electrical>
    <path>Aircraft/Rascal/Systems/electrical.xml</path>
   </electrical>
  </systems>
```

Fig. 19. A part of the set configuration file

C. *OpenEaagles Simulation Framework*

OpenEaagles is an open-source multi-platform simulation framework targeted to help simulation engineers and software developers rapidly prototype and build robust, scalable, virtual, constructive, stand-alone, and distributed simulation applications. OpenEaagles is written in the C++ programming language and has been used extensively to build applications which demand deterministic real-time performance or execution as fast as possible. This includes applications used to conduct human factor studies, operator training, or the development of complete distributed virtual simulation systems. OpenEaagles has also been used to build stand-alone and distributed constructive applications oriented at system performance analysis. Constructive-only simulation applications which do not need to meet time-critical deadlines can use models with even higher levels of fidelity. [17] [18] [57]

It should be emphasized that OpenEaagles is a cycle or frame-based system, not a discrete-event simulator. This approach satisfies the requirements for which it is designed; namely, support for models of varying levels of fidelity including higher-level physics-based models, digital signal processing models and the ability to meet real-time performance requirements. Model state can be captured with state machines and state transitions can use the message passing mechanisms provided by the framework. [19] [20] [57]

The framework embraces the Model-View-Controller (MVC) software design pattern by partitioning functional components into packages as can be seen in Fig. 20 (packages with white/clear background indicate the use of a third party open-source tool). This concept is taken a step further by providing an abstract network interface; thus, custom protocols can be implemented without affecting system models. The framework supports a number of other third party open-source tools such as FLTK, Fox and wxWidgets for cross-platform GUI applications, and JSBSim as a high quality flight dynamics model. [18] [19] [20]

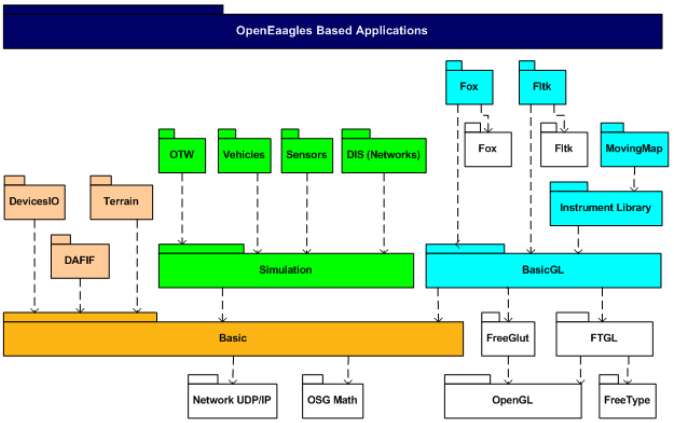

Fig. 20. OpenEaagles Package Hierarchy [18]

The simulation section provides a wealth of capabilities including abstract classes for representing a variety of entity types such as aircraft, ships, tanks, ground vehicles, space vehicles and even lifeforms. Moreover, a complete radar modelling environment is included. [20]

The graphics hierarchy provides a collection of classes which can be used to render instruments which are commonly used in operator-vehicle interface displays. The available instruments include, for example, analog dials for altimeters, dials for direction finders, speedometer dials, and landing gear indicators. [20]

Distributed applications can interoperate with other systems and simulations through Distributed Interactive Simulation (DIS) and/or High Level Architecture (HLA) interfaces. Numerous DIS compliant distributed simulation applications have been built using this framework as the foundation. [17] [18] [19]

Specific applications using the framework to support simulation activities include a UAV ground control station (Predator MQ-9), representative F-16 cockpits, Integrated Air Defense Systems (IADS) and a futuristic battle manager. OpenEaagles is also suitable for use in multi-agent applications. [19]

The framework is routinely compiled with Microsoft Visual Studio for the Windows environment and GCC for Linux. Applications probably perform best when they are executed on dual-core or dual-CPU systems. [19]

However, the project files for Codelite and Codeblocks development environments can be generated by OE_source_ROOT\build\premake\make.bat file. For the generating of the Codelite project files, “%Premake% codelite” has to be added to the end of the make.bat file.

D. *MIT and ESOTEC Software*

This chapter contains an analysis of MIT (Massachusetts Institute of Technology) and ESOTEC (Esoteric Technology) open-source software. ESOTEC software, written by Carter [134], is an extension of MIT software, which was developed mainly by Drela and Youngren [135].

In addition to described software, it is also worth mentioning the Transport Aircraft System OPTimization (TASOPT). TASOPT is a program for optimizing the airframe of a wing-tube transport aircraft, together with the engine parameters and operating parameters. However, despite there may be found some parts which can be useful, this software as a whole is probably inappropriate for UAVs. [136]

1) Athena Vortex Lattice (AVL)

AVL (whose logo is illustrated in Fig. 21) is a program for the aerodynamic and flight-dynamic analysis of rigid aircraft of arbitrary configuration. It employs an extended vortex lattice model for the lifting surfaces, together with a slender-body model for fuselages and nacelles. [132] [109] [53]

General nonlinear flight states can be specified. The flight dynamic analysis combines a full linearization of the aerodynamic model about any flight state, together with specified mass properties. [132]

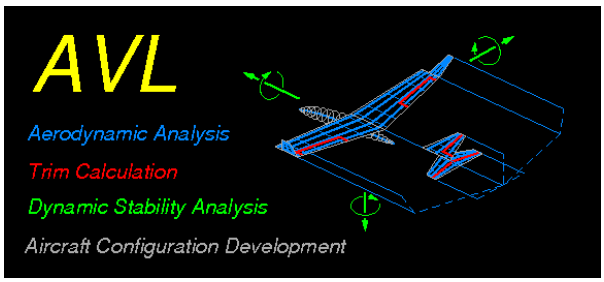

Fig. 21. The logo of AVL [132]

AVL has a large number of features intended for rapid aircraft configuration analysis. The major features are as follows: [133] [132] [53]

- Aerodynamic Components (Lifting surfaces, and Slender bodies)
- Configuration definition (Keyword-driven geometry input file, Defined sections with linear interpolation, Section properties, Scaling, translation, rotation of entire surface or body, and Duplication of entire surface or body)
- Singularities (Horseshoe vortices - surfaces, Source plus doublet lines - bodies, and Finite-core option)
- Discretization (Uniform, Sine, Cosine, and Blend)
- Control deflections (Via normal-vector tilting, Leading edge or trailing edge flaps, and Flaps independent of discretization)
- General Freestream description (alpha, beta flow angles; p, q, r aircraft rotation components; Subsonic Prandtl-Glauert compressibility treatment)
- Aerodynamic outputs (Aerodynamic forces and moments, in body or stability axes, Trefftz-plane induced drag analysis, and Force and moment derivatives w.r.t. angles, rotations, controls)
- Trim Calculation (Operating variables such as alpha, beta, p, q, r, and control deflections, Constraints such as direct constraints on variables and indirect constraints via specified C_L and moments, and multiple trim run cases which can be defined, saved, recalled)
- Optional mass definition file for trim setup and Eigenmode analysis (User-chosen units, and Itemized component location, mass, inertias)
- Trim setup of constraints (level or banked horizontal flight, and steady pitch rate (looping) flight)
- Eigenmode analysis (Predicts flight stability characteristics, Rigid-body, quasi-steady aero model, Eigenvalue root progression with a parameter, Display of Eigenmode motion in real time, and Output of dynamic system matrices)

AVL utilizes the extended vortex lattice method to determine the aerodynamic loads along the span of all aerodynamic surfaces, interprets the geometric configuration of the aircraft, and discretizes the wing into a finite element mesh. An AVL aircraft model and the corresponding static lift distribution can be seen in Fig. 22. [31] [53]

AVL allows the user to obtain individual lift forces for each element in the mesh. The total lift load on each segmented portion of the wing is calculated by summing the elemental lift forces that are located within the geometric bounds defined by the segmented plate control surfaces. [31]

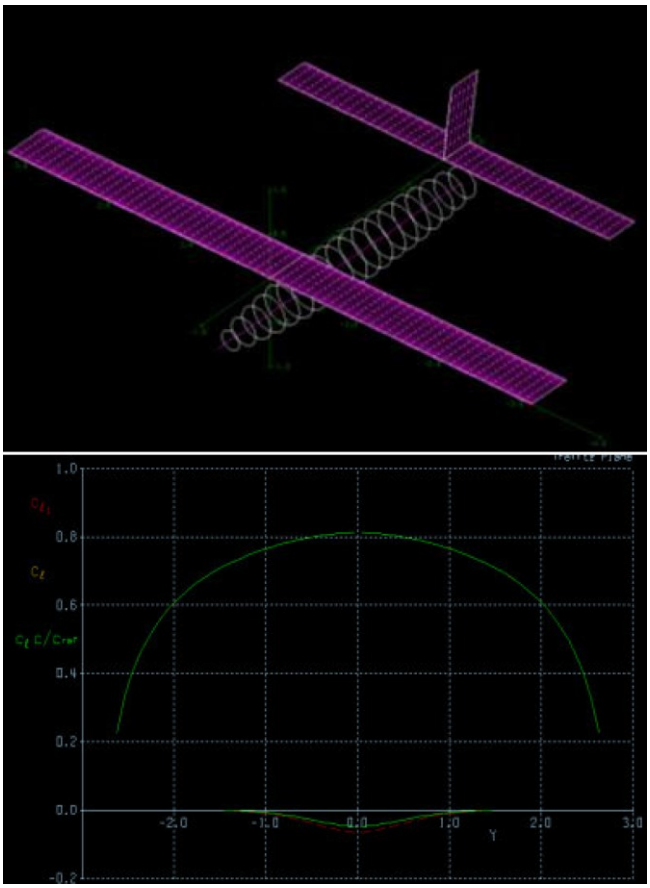

Fig. 22. An AVL model and the lift distribution of an aircraft [31]

AVL works with three input files, all in plain text format with the following extensions: [133] [109]

- .avl - required main input file defining the configuration geometry
- .mass - optional file giving masses and inertias, and dimensional units
- .run - optional file defining the parameter for some number of run cases

AVL was tested, for example, on Odyssey UAV [31], and MAV prototype in [109] and [53].

However, like any computational method, AVL has some limitations. A vortex-lattice model like AVL is best suited for aerodynamic configurations which consist mainly of thin lifting surfaces at small angles of attack and sideslip. These surfaces and their trailing wakes are represented as single-layer vortex sheets, discretized into horseshoe vortex filaments, whose trailing legs are assumed to be parallel to the x-axis. [133] [53]

AVL provides the capability to model also slender bodies such as fuselages and nacelles via source plus doublet filaments. The resulting force and moment predictions are consistent with slender-body theory, but the experience with this model is relatively limited, and hence modelling of bodies should be done with caution. If a fuselage is expected to have little influence on the aerodynamic loads, it is simplest to just leave it out of the AVL model. However, the two wings should be connected by a fictitious wing portion which spans the omitted fuselage. [133]

AVL assumes quasi-steady flow, meaning that unsteady vorticity shedding is neglected. More precisely, it assumes the limit of small reduced frequency, which means that any oscillatory motion (e.g. in pitch) must be slow enough so that the period of oscillation is much longer than the time it takes the flow to traverse an airfoil chord. This is true for virtually any expected flight maneuver. Also, the roll, pitch, and yaw rates used in the computations must be slow enough so that the resulting relative flow angles are small. This can be judged by the dimensionless rotation rate parameters, which should fall within the following practical limits: [133] [53]

- -0.10 < pb/2V (Roll rate) < 0.10
- -0.03 < qc/2V (Pitch rate) < 0.03
- -0.25 < rb/2V (Yaw rate) < 0.25

These limits represent violent aircraft motion and are unlikely to be exceeded in any typical flight situation, except perhaps during low-airspeed aerobatic maneuvers. However, if any of these parameters falls outside of these limits, the results should be interpreted with caution. [133]

Compressibility is treated in AVL using the classical Prandtl-Glauert (PG) transformation, which converts the PG equation to the Laplace equation, which can then be solved by the basic incompressible method. This is equivalent to the compressible continuity equation, with the assumptions of the state without rotation, and linearization about the freestream. The forces are computed by applying the Kutta-Joukowsky relation to each vortex, which remains valid for compressible flow. [133]

The linearization assumes small perturbations (thin surfaces) and is not completely valid when velocity perturbations from the free-stream become large. The relative importance of compressible effects can be judged by the PG factor (8), where M is the freestream Mach number. A few values are given in Table II, which shows the expected range of validity. [133]

$$\frac{1}{B} = \frac{1}{\sqrt{1 - M^2}} \tag{8}$$

TABLE II. THE EXPECTED RANGE OF THE VALIDITY OF PRANDTL-GLAUERT TRANSFORMATION [133]

M	*1/B*	*Validity*
0.0	1.000	PG expected valid
0.1	1.005	
0.2	1.021	
0.3	1.048	
0.4	1.091	
0.5	1.155	
0.6	1.250	
0.7	1.400	PG suspect
0.8	1.667	PG unreliable
0.9	2.294	PG hopeless

For swept-wing configurations, the validity of the PG model is best judged using the wing-perpendicular Mach number in (9). Since $M_{perp} \leq 0.6$, swept-wing cases can be modeled up to higher M values than unswept cases. [133]

$$M_{perp} = M \cdot \cos(sweep) \tag{9}$$

For instance, a 45° swept wing operating at freestream M = 0.8 has M_{perp} = 0.8 * cos(45) = *0.566*, which is still within the expected range of PG validity in the Table II; thus, reasonable results may also be expected for this case. [133]

2) XFOIL

Most of the second-generation small UAVs have used airfoil sections designed specifically for their application. The two methods most often used to design airfoils at low Reynolds numbers are attributed to Eppler (implemented in the code described in the chapter V.E.1)) and Drela (XFOIL). [28]

XFOIL is an interactive program for the design and analysis of subsonic isolated airfoils. The main goal was to combine the speed and accuracy of high-order panel methods

with the new fully-coupled viscous/inviscid interaction method. XFOIL consists of a collection of menu-driven routines which perform various useful functions such as: [21] [130] [32]

- Viscous (or inviscid) analysis of an existing airfoil, allowing forced or free transition, transitional separation bubble(s), limited trailing edge separation, lift and drag predictions just beyond C_{Lmax}, and Karman-Tsien compressibility correction
- Airfoil design and redesign by interactive specification of a surface speed distribution via screen cursor or mouse. Two such facilities are implemented: Full-Inverse, based on a complex-mapping formulation, and Mixed-Inverse, an extension of XFOIL's basic panel method
- Airfoil redesign by interactive specification of new geometric parameters such as new max thickness and/or camber, new LE radius, new TE thickness, new camber line via geometry specification, new camber line via loading change specification, flap deflection, and explicit contour geometry (via screen cursor)
- Blending of airfoils
- Drag polar calculation with fixed or varying Reynolds and/or Mach numbers
- Writing and reading of airfoil geometry and polar save files
- Plotting of geometry, pressure distributions, and polars

XFOIL uses a text x- and y-coordinate file to model two-dimensional airfoils. The user may input an airfoil from a file or select a NACA four- or five-series airfoil and XFOIL will build the appropriate coordinate file. The user can then make changes to inviscid/viscous properties such as Mach number and Reynolds number (Re). XFOIL will then use the user data to simulate flight at many angles of attack, to return lift coefficient (C_L), drag coefficient (C_D), and moment coefficient (C_M) in the form of a saved polar file, and to generate C_L versus α, and C_L versus C_D plots. [32]

XFOIL stores all its data in RAM during execution. Saving of the data to files is NOT normally performed automatically; thus the user must be careful to save work results before exiting XFOIL. [130]

XFOIL gives results much more quickly than more advanced CFD programs and still provides results accurate enough to be a good design tool. However, XFOIL works only for two dimensions and is only effective at low Reynolds numbers and incompressible flows. [32]

XFOIL was used in design of camber-controlled morphing UAVs in [32]; the results are:

- XFOIL follows the wind tunnel data well
- XFOIL is predicting both lift and drag coefficients within an acceptable range or accuracy
- XFOIL provides accurate simulation of flap addition on airfoils
- XFOIL can only model plain flaps with a sealed gap

Moreover, following the validation tests, a series of aerodynamic simulations using XFOIL were also performed on the LEEUAV airfoil in [131].

However, independently of any accuracy, the following situations may cause problems strictly due to numerical rounding off: [130]

- Excessively small panel(s) somewhere on the airfoil
- Airfoil is located too far from origin
- Airfoil is too thin

These situations will rarely result in an arithmetic failure, but will typically result in a rough C_p distribution. [130]

3) QPROP/QMIL

Same as aerodynamic design, the design of propeller is very important. When the inefficient or inappropriate propeller is used in UAV, all advantages of excellent-designed aerodynamics may remain underutilized. Modern propeller theory is analogous to wing theory in which the propeller blade is considered to be a lifting surface about which there is a circulation associated with the bound vorticity and a vortex sheet is continuously shed from the trailing edge. [54]

QPROP (whose logo is illustrated in Fig. 23) is an analysis program for predicting the performance of propeller-motor or windmill-generator combinations. The same formulation applies to the companion propeller/windmill design program QMIL, which generates propeller geometries for the Minimum Induced Loss (MIL) condition, or windmill geometries for the MIL or Maximum Total Power (MTP) conditions. [122] [123]

Fig. 23. The logo of QPROP [122]

QPROP and QMIL use an extension of the classical blade-element/vortex formulation, developed originally by Betz [124], Goldstein [125], and Theodorsen [126], and reformulated somewhat by Larrabee [127]. The extensions include: [123] [53] [54]

- Radially-varying self-induction velocity which gives consistency with the heavily-loaded actuator disk limit
- Perfect consistency of the analysis and design formulations
- Solution of the overall system by a global Newton method, which includes the self-induction effects and power-plant model

- Formulation and implementation of the Maximum Total Power (MTP) design condition for windmills

QPROP has a relatively sophisticated and accurate prop aerodynamic model, and a general motor model which can be implemented via a user-supplied subroutine if necessary. [128] [109] [53]

The enhancement in the classical blade-element/vortex formulation is primarily in the correct accounting of the propeller's self-induction, which makes QPROP accurate for very high disk loadings, all the way to the static-thrust case. The blade airfoil lift characteristic is assumed to be a simple linear $C_{L(alpha)}$ line with C_{Lmax} and C_{Lmin} stall limiting. The profile drag characteristic is a quadratic $C_D(C_L)$ function, with an approximate stall drag increase, and a power-law scaling with Reynolds number. The model applies equally well to propellers and windmills. [128]

QPROP requires a fairly detailed description of the propeller geometry and blade airfoil characteristics. The specification of the diameter and pitch of a propeller is in general insufficient to accurately capture the propeller performance. [128] [53]

The default motor type 1 corresponds to a brushed DC motor, and is modeled using the fairly standard approach with an RPM/Volt motor constant K_v, an electrical resistance R, and a constant rotational friction described by the zero-load current I_0. [128]

Motor type 2 corresponds to a brushed DC motor, and is a more accurate extension of the type 1 model above. The extensions are improved models of the frictional torque, temperature-dependent resistance, and magnetic lags. [128]

Any other motor model can be coded in SUBROUTINE MOTORQ (in motor.f), as a $Q(w,V)$ function. The derivatives dQ/dw and dQ/dV must also be returned. The subroutine source header fully describes the inputs and outputs. [128] [109]

For non-electric motors, the voltage (V) passed to MOTORQ can represent any suitable power-control variable, e.g. throttle setting, fuel flow rate, etc. [128] [109]

QPROP and QMIL have identical theoretical formulations and very similar input files. This is described in detail in [128], [129], and in the theory document [123]. Their only difference is the variables which are treated as knowns and unknowns (the variables are swapped). [129]

QPROP's output is entirely in tabular text format, and is typically used in conjunction with the user's own plotting programs. It is intended for large-scale parametric sweeps, driven manually or via batch execution. [122]

QPROP was used to find an optimal propeller, motor, and energy requirement, for example, on MAV prototype in [109] and [53].

4) XROTOR

XROTOR is an interactive program for the design and analysis of ducted and free-tip propellers and windmills. It consists of a collection of menu-driven routines which perform various useful functions such as: [115] [116]

- Design of minimum induced loss rotor (propeller or windmill)
- Prompted input of arbitrary rotor geometry
- Interactive modification of rotor geometry
- Twist optimization of an arbitrary rotor for minimum induced loss
- Analysis of a rotor with a wealth of choices of operating parameters
- Incoming slipstream effects (from an upstream propeller, viscous wake, etc.)
- Multi-point parameter display
- Structural analysis and corrections for twist under load
- Acoustic analysis with dB noise footprint predictions
- Interpolation of geometry to radii of interest
- Plotting of geometry, aerodynamic parameters, and performance maps

The design procedure of a rotor design allows calculation of a rotor chord and blade angle (c/R, beta) distributions to achieve a Minimum Induced Loss (MIL) circulation distribution. This can be either the i) Betz-Prandtl distribution (Graded-Momentum Formulation), or ii) Goldstein distribution (Potential Formulation), depending on the state of the FORM toggle. [116]

The design of a new rotor is typically begun with the INPU command, which prompts the user for all required design-parameter inputs, and then follows by displaying the input-modification menu. [116]

All the design parameters will retain their values for the length of the XROTOR session. Hence the designed rotor can be analyzed in the other menus, and can then be further redesigned in DESI just by invoking EDIT again. [116]

QPROP/QMIL has almost the same theoretical formulation as XROTOR, and they are also extensively documented; thus, their documentation can be used as the basic documentation for XROTOR. [115]

In [117], XROTOR and CROTOR (described below) were used to determine the 10 best propellers with optimal efficiency for a multi-mission micro aerial vehicle. [117]

a) CROTOR

CROTOR v755-ES is all of XROTOR 7.55 along with a couple of features ported from DFDC v070-ES (described in the chapter V.D.5)a)). With the source directory growing by some 70 percent (including some XFOIL code), the package is dubbed CROTOR to distinguish it from its parent. [118]

CROTOR added the following features: [118]

- Counter-Rotation Facility

- Multi-Axis Parametric Analysis
- Blade Lofting Facility
- Multi-Re Plotting in AERO

Subroutine CROTOR automates the design and analysis of converging counter-rotating rotors while providing an effective user interface and reporting. The result is a flexible and robust counter-rotation design/analysis facility. [118] [119]

Rotors can be designed directly in CROTOR or imported from XROTOR. When rotors are designed or loaded into CROTOR, the geometries, names and imposed slipstreams are stored in CROTOR to be loaded into XROTOR as required. [119]

When operating parameters and geometries of two rotors have been defined, the code converges the dual-rotor system by alternately loading and analyzing of the forward and aft rotors. When the thrust of each rotor converges, the iteration stops, and the output for both rotors is displayed. Each rotor can then be run independently in the converged slipstream for closer inspection. [119]

To reduce the complexity of working with multiple rotors and input, a Default Input system is used which allows the user to progress efficiently through a design study. [119]

Subroutine ESPARA is one half of a parametric analysis system originally developed for a propeller company using third party blades in their electric constant speed hubs. The problem was to select appropriate blades for a particular engine/airframe. ESPARA makes multi-axis parameter sweeps and stores data in a multi-rotor database to be displayed and plotted with great flexibility in a standalone utility called ESPROP. This facility is intended for practical applications in the field, selecting propellers for real world applications by directly comparing them over any operating range within the database bounds. Currently only variable pitch props are supported although support for fixed pitch is planned by calculating on the same database. [118]

The ESPROP system has two main applications: [120]

- The propeller designer seeking to directly compare competing designs over a wide range of operating conditions.
- The propeller manufacturer seeking to select the optimum existing blade for a specific application.

ESPROP databases can be built from any blade geometry which has been loaded into XROTOR, over any range of operating parameters for which XROTOR will converge solutions. [120]

Subroutine ESLOFTX is a port and further development of ESLOFT for DFDC (described in the chapter V.D.5)a)), allowing rotor designs to be explicitly defined in three dimensions and exported into CAD for 3D modelling, meshing or manufacture. Support has been added for round tips, along with splined thickness or thickness/chord distributions (in addition to linear and parabolic distributions). Version ES1.1 adds circular root blends, important for windmills and constant speed propellers. [118]

The definition of blade thickness plays a main role in ESLOFTX. A smooth thickness distribution is desired for both aerodynamic and structural reasons. Since chord is defined by the blade design, defining the distribution of thickness (t) or thickness/chord (t/c), the blade allows for section t/c to be determined at any station radius. Sections are interpolated to the required t/c from parent airfoils bounding the t/c (or extrapolated as necessary). [121]

Because the setting of the REexp parameter in AERO has sometimes seemed like guesswork, the improvement has been implemented. In addition to polar plots at multiple Mach numbers, AERO now supports plotting at multiple Reynolds numbers, providing precise feedback on the effects of REexp. [118]

5) Ducted Fan Design Code (DFDC)

DFDC is an analysis code for axisymmetric ducted rotor design and analysis. Wall Boundary Layer (BL) analysis solves the wall shear forces with a BL calculation (not interacted with the inviscid flow, as is done in XFOIL). The 0.70 version adds analysis and design for a ducted rotor with stator. [112]

There is also a Win32 GUI wrapper for DFDC, Whirlwind. However, it does not support all DFDC functionality. [113]

The logo of DFDC is shown in Fig. 24. DFDC has a number of features intended for rapid duct and ducted rotor design and analysis, e.g. axisymmetric components, aerodynamic outputs, analysis capability, rotor/stator design capability, geometric redesign of duct walls (XFOIL-like geometry modification), BL analysis of duct and center-body walls (viscous forces), aerodynamic redesign of duct walls to specified pressures, and an input file. [112] [113]

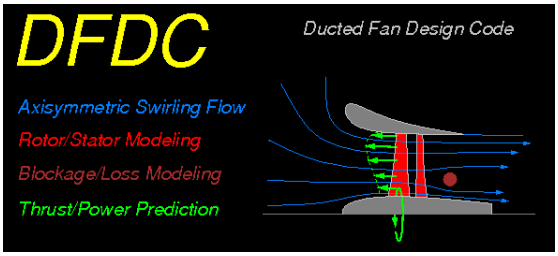

Fig. 24. The logo of the Ducted Fan Design Code [113]

DFDC works with a single input file in plain text format. This file contains the duct case data, the duct geometry data, actuator disk or rotor blade data and drag source data. The sections of this file are separated by keywords and can be input in any order. [112]

The operating point section contains the flow condition and operating point data for the case (e.g. the freestream velocity, reference velocity, RPM, fluid properties, etc.). The aero properties section contains aerodynamic data which is used for each blade element in a rotor analysis or design. The actuator disk section contains a specification for an actuator disk to model the duct rotor. The rotor section contains a specification for a bladed disk in the ducted fan. The drag object section contains a CDA (drag area) and XY (XR in axisymmetric system) coordinates of the drag line. The geometry section contains the center-body and duct wall coordinates. [112]

Optionally, a second file can be used, containing information to redistribute points on the duct and center-body walls. This paneling file has the extension of .pan. However, the paneling information can also be put directly into the case file in the paneling data section. Moreover, user-specified paneling information is normally not needed; the automatic point distribution scheme works for most inputs without further interaction. [112]

a) DFDC v070-ES

Because the development of DFDC appears to be halted, Esotec Developments started to upgrade the code. The new code fixed most known bugs, added enhancements to AERO, plotting, reporting, multi-rotor, blade editing, an experimental blade blockage model, and ESLOFT. [114]

ESLOFT is a tool for getting rotor and stator blade designs out of DFDC and into CAD for subsequent 3D modelling, meshing or manufacture. This is accomplished by calculating on the rotor/stator geometry in conjunction with airfoil geometries (parent airfoils) and blade thickness distributions (user controlled) to generate point files which can be imported into 3D CAD systems. ESLOFT is fully integrated with DFDC and generates accurate output with minimum user input. [114]

6) Dynamic Soaring simulation and optimization program (DSOPT)

Dynamic soaring is a flying technique used to gain energy by utilizing wind shear over altitude to reduce energy consumption and extend flight duration. Because the wind shear gradient is persistently distributed in the boundary layer above the ocean surface, dynamic soaring can be widely used in UAVs and may have the potential to support almost perpetual flight. [110]

DSOPT uses the "Inverse Dynamics" approach to simulating a Dynamic Soaring orbit. The code is two nested loops, with an inverse-dynamics integrator on the inside and requires Fortran 77 compiler. [111]

The bulk of the calculations are performed in SUBROUTINE ORBIT1, which assumes that the path shape in xyz space is known, and is prescribed to be an ellipse of specified size, oriented at some tip and lean angles within the atmospheric wind shear layer. SUBROUTINE ORBIT1 is also provided with an initial velocity V_i at the first point in the orbit, and then integrates the equations of motion along the known trajectory, using simple representations of the airplane's C_L and C_D. One output is the final velocity V_f at the last point in the orbit (which is at the same spatial location as the first point). In general V_i and V_f will not match, in which case V_i is modified and the orbit is recalculated again. When V_f matches V_i to within some small tolerance the orbit is converged. This means that it is periodic, and hence represents sustained Dynamic Soaring. [111]

The output quantities are, for example, time, position, ground speed, air speed, lift coefficient and a number of others. [111]

The code is heavily commented, and it is fairly easy to locate the formulas used for the various models for C_L, C_{Di}, C_{Dp}, C_{Dw}, wind shear field, etc. The dynamic equations which are integrated are simply $F=ma$, with three separate components in the xyz directions. [111]

The outermost OPTIMIZER loop is optional and adjusts the selected parameters of the formulation via gradient-descent steps to maximize the speed $V_{measured}$ at a selected point in the Dynamic Soaring orbit. [111]

E. Public Domain Aeronautical Software (PDAS)

For many years the Air Force, Navy, NASA, and educational institutions have sponsored the development of computer software which is useful to aeronautical engineers, airplane designers, and aviation technicians. [8]

Public Domain Aeronautical Software (PDAS) was founded to make this valuable software available to the aeronautical community for use on desktop computers. These programs include descriptions and complete public domain source code (written mostly in the FORTRAN programming language). The source code is not copyrighted and may be used in whole or part in any of aeronautical studies. Moreover, many programs have sample cases (both input and output). However, some of the programs are noted as "work-in-progress", indicating that they are lacking in instructions or documentation or do not run properly. [8] [9]

The list of the useful software for development of UAVs which is included in this subchapter may not be comprehensive. However, other interesting applications can be found in [9], for instance:

- *Atmosphere (ATMOS)* - characterizes the 1976 standard atmosphere to 1000 km altitude, including nonstandard atmosphere routines (hot, cold, polar, tropical). [64]
- *Real Gas Properties (GASP)* - computes real gas properties of ten important gases over a wide range of temperatures and pressures. Covers cryogenic regions and saturated liquid/gas regions. [65]
- *Thermodynamic and Transport Properties of Fluids (FLUID)* - a companion program to GASP computes thermodynamic and transport properties of many gases. Treats air and steam as well as pure fluids. [66]
- *A Compressible Flow Calculator (VuCalc)* - performs calculations in compressible fluid dynamics. There are six different classes of calculations: Isentropic Flow, Normal Shock, Oblique Shock, Standard Atmosphere, Rayleigh Flow, Fanno Flow. [67]
- *Turbulent Skin Friction by the Reference Temperature Method of Sommer and Short (TURBSF)* – contains the formula developed by Sommer and Short with including the temperature of the flow as a variable. As a result, the created function has three arguments, Reynolds number, Mach number and freestream temperature. For the great majority of flight problems, the variation of friction with temperature is of little importance. This effect is small at subsonic speeds but becomes appreciable for supersonic and hypersonic aircraft. [82] [105]

- *A segmented mission analysis program for low and high speed aircraft (NSEG)* - was developed to perform rapid aircraft mission analyses which is based upon the use of approximate equations of motion whose form varies with the type of flight segment (e.g. take-off, accelerations, climbs, cruises, descents, decelerations, and landings). There are three main atmosphere options available: the 1962 U.S. Standard atmosphere, a stratified atmosphere model, and an external atmosphere model supplied by the user. [68]
- *Conical relaxation program for supersonic wing design and analysis (COREL)* [69] and *supersonic wing design and analysis program (W12SC3)* [70] - can be run also at subsonic speeds despite their titles.
- *Two-dimensional grids about airfoils and other shapes by the use of Poisson's Equation (GRAPE)* - can provide the aerodynamic analysis with an efficient and consistent means of grid generation and should be numerically stable and computationally fast. [71]
- *NASA-AMES WingBody Panel Code (WINGBODY)* - The classic NASA program for computing subsonic and supersonic aerodynamics of a wing-body combination by using Panel Code [80]
- *V/STOL Aircraft Sizing and Performance (VASCOMP II)* - developed to aid in the comparative design study of V/STOL (vertical/short take-off and landing) aircraft systems by rapidly providing aircraft size and mission performance data. Generality and flexibility were maintained during formulation of the program in order to permit an accurate simulation of virtually any V/STOL configuration. [83] [9]

1) The Eppler Airfoil Code (PROFILE)

PROFILE is one of computer programs for low-speed (incompressible) airfoils. The program has been successfully applied at Reynolds numbers from 20 thousand to 100 million. [92] [72]

A conformal-mapping method for the design of airfoils with prescribed velocity distribution characteristics, a panel method for the analysis of the potential flow about given airfoils, and a boundary-layer method have been combined. With this combined method, airfoils with prescribed boundary-layer characteristics can be designed and airfoils with prescribed shapes can be analyzed. [72]

The flow about an airfoil in free air can be described approximately by a boundary-layer flow near the surface of the airfoil and by a potential flow everywhere else. Boundary-layer theory can be applied to the flow about an airfoil in two ways. First, the boundary-layer development can be determined for a given potential flow velocity distribution. This is the direct or analysis problem. Second, the potential-flow field, or at least some of its properties, can be determined for a given boundary-layer development. This is the inverse or design problem. [92] [72]

The potential flow inverse problem still plays a major role in airfoil design. This problem has been solved exactly by means of conformal mapping which is similar to the method of Lighthill; it is direct, and solves most multipoint design problems in a very simple manner. A potential-flow analysis method is also required for comparison with wind tunnel tests of given airfoils and for analyses of airfoils generated by the design method and then modified by a flap deflection. The airfoil analysis problem is solved using a distributed surface singularity method. The boundary-layer method uses integral momentum and energy equations. The present method does not contain boundary-layer displacement iteration. [92] [72]

Although this program is of great historical importance and current papers which refer to calculations performed with PROFILE can still be found, it is not the program of choice for someone learning about airfoil plus boundary layer calculations. For this kind of interest, XFOIL is recommended. XFOIL has been described in the chapter V.D.2). [72]

2) Minimum Drag Camber Surface by Vortex Lattice (VLMD)

This program represents a subsonic aerodynamic method for determining the mean camber surface of trimmed noncoplanar planforms with minimum vortex drag. With this program, multiple surfaces can be designed together to yield a trimmed configuration with minimum induced drag at some specified lift coefficient. [84]

The method uses a vortex-lattice and overcomes difficulties with chord loading specification. A Trefftz plane analysis is used to determine the optimum span loading for minimum drag. The program then solves for the mean camber surface of the wing associated with this loading. Pitching-moment or root-bending-moment constraints can be employed at the design lift coefficient. [84]

Sensitivity studies of vortex-lattice arrangements have been made with this program and comparisons with other theories show generally good agreement. The program is very versatile and has been applied to isolated wings, wing-canard configurations, a tandem wing, and a wing-winglet configuration. [84]

The design problem solved with this code is essentially an optimization one. A subsonic vortex-lattice is used to determine the span load distribution(s) on bent lifting line(s) in the Trefftz plane. A Lagrange multiplier technique determines the required loading which is used to calculate the mean camber slopes, which are then integrated to yield the local elevation surface. The problem of determining the necessary circulation matrix is simplified by having the chordwise shape of the bound circulation remain unchanged across each span, though the chordwise shape may vary from one planform to another. The circulation matrix is obtained by calculating the spanwise scaling of the chordwise shapes. A chordwise summation of the lift and pitching-moment is utilized in the Trefftz plane solution on the assumption that the trailing wake does not roll up and that the general configuration has specifiable chord loading shapes. [84]

3) Induced Drag from Span Load Distribution (INDUCED)

The induced drag may be computed from the span load distribution on a planar wing. Most books on aerodynamics show how to do this if the analytical form of the loading

function is known. This algorithm enables to solve the same problem when only a few sparse values of the loading function are known. [74]

A simple algorithm for computing a curve which in one sense is the smoothest which exactly matches the data points and produces the Fourier sine coefficients as part of the solution is described in [93] by Lundry. This technique and other similar algorithms are widely used by specialists. The routines given in this program are a coding of Lundry's equations 3 and 5 for asymmetric and symmetric loadings. [74]

The coefficients are computed with a call to subroutine "ComputeFourierCoefficients". The drag may then be computed from the coefficients by use of the function "DragFromCoefficients". Moreover, lift coefficient may be computed by (10). [74]

$$C_L = \frac{PI \cdot span \cdot span \cdot coeff(1)}{sref} \tag{10}$$

If the induced drag without the coefficients is needed to be calculated, the following two functions can be used: [74]

- AsymmetricLoadingInducedDrag
- SymmetricLoadingInducedDrag

These functions take the loadings and return D/q (these variables have been described in the chapter II.B). If drag coefficient is required, the result has to be divided by the reference area (sref). [74]

Once the module is compiled, there is access to any of the routines by inserting the statement "USE InducedDrag" in application programs. [74]

4) Flutter Analysis by Strip Theory (FLUTTER)

A modified strip analysis has been developed for rapidly predicting flutter of finite-span, swept or unswept wings at subsonic to hypersonic speeds. The method employs distributions of aerodynamic parameters which may be evaluated from any suitable linear or nonlinear steady-flow theory or from measured steady-flow load distributions for the underformed wing. The method has been shown to give good flutter results for a broad range of wings at Mach number from 0 to as high as 15.3. [73]

Flutter characteristics have been calculated by the modified strip analysis and compared with results of other calculations and with experiments for Mach numbers up to 15.3 and for wings with sweep angles from 0 degrees to 52.5 degrees, aspect ratios from 2.0 to 7.4, taper ratios from 0.2 to 1.0, and center-of-gravity positions between 34% chord and 59% chord. These ranges probably cover the great majority of wings which are of practical interest with the exception of very low-aspect-ratio surfaces such as delta wings and missile fins. [73]

5) Mean Aerodynamic Chord of a Wing (GETMAC)

GETMAC computes the mean aerodynamic chord (MAC) of a wing of arbitrary planform. GETMAC reads the definition of a wing from an arbitrary number of chords, each defined by its spanwise location, longitudinal position of its leading edge, and its length. [75]

The program reads the input file and prints the area of wing, length of MAC, y of MAC, x_{LE} (x of Leading Edge) of MAC, x_{TE} (x of Trailing Edge) of MAC, and x of $c/4$ of MAC. [75]

A following example illustrates a wing similar to the B-2 airplane. The chords are 1300, 500, 500, and 0 in length. The leading edges are at 0, 400, 750, and 1000 with y equals 0, 480, 900, and 1200. The projection of these points can be seen in Fig. 25. The input to GETMAC for the right wing is shown in Fig. 26. And finally, Fig. 27 presents the resultant MAC. [94] [95] [96]

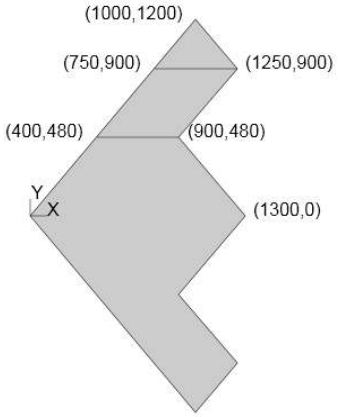

Fig. 25. The projection of the wing inserted to GETMAC [95]

```
&chord y=0,   xle=0,   c=1300/
&chord y=480, xle=400, c=500/
&chord y=900, xle=750, c=500/
&chord y=1200, xle=1000, c=0/
```

Fig. 26. The input file to GETMAC for the right wing [95]

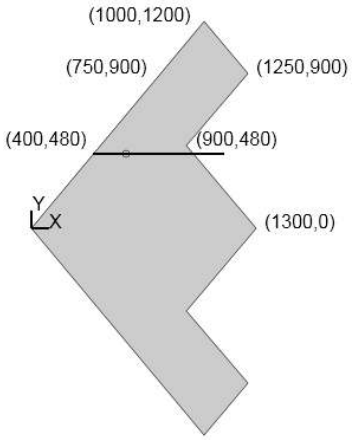

Fig. 27. The projection of the resultant MAC [96]

It is evident that the use of a homogeneous material is a necessary condition to achieving this MAC. Moreover, the shape of an elliptical wing is more difficult to define and its real MAC may be slightly different.

6) NACA Airfoil Coordinates (NACA456)

A large number of NACA airfoil shapes have been successfully used over years as wing sections or tail sections for general aviation and military aircraft, as well as propellers and helicopter rotors. The ordinates for numerous specific airfoils of these families at a coarse set of data points were published in a series of NACA reports. However, when performing parametric studies on effects of such variables as thickness, location of maximum thickness, leading-edge radius, location of maximum camber and others, it is not always easy to obtain the ordinates of the desired shapes rapidly and accurately. To remedy this problem the NASA Langley Research Center sponsored the development of computer programs for generation of ordinates of standard NACA airfoils. [77]

NACA 4-digit, 4-digit modified, 5-digit, and 16-series airfoils are defined by algebraic equations. These thickness families are combined with appropriate mean lines to produce the final thick cambered airfoil. [77] [101]

NACA 6-series and 6A-series airfoils are not defined by algebraic equations, but use conformal mapping of a circle into an airfoil shape. These thicknesses are combined with 6-series mean lines to produce the final thick cambered airfoil. [77] [101]

The coordinates of 4-digit, 4-digit-modified, 5-digit, 6-series, and 16-series airfoils may be accurately calculated by NACA456. It is modified to present upper and lower surface points at the same x-coordinate. All NACA airfoils are produced by combining a thickness distribution and a mean line into a definition of the upper and lower surfaces of the airfoil. [9] [100]

NACA456 is a complete revision of the NASA Langley programs for computing the coordinates of NACA airfoils. The NASA 1996 program was used as a guide for the development of a program which is highly modular and contains several features which were requested by user of the older programs. This program is a console application for which the user prepares an input file. An output file containing the airfoil geometry is generated by the program. Moreover, a file for graphical examination is produced. [100]

a) AirfoilTools.com

A similar online version of the NACA 4 digit generator, the NACA 5 digit generator, an airfoil database, and other airfoil tools can be found in [99].

7) Mass Properties of a Rigid Structure (MASSPROP)

MASSPROP was developed for the rapid computation of the mass properties of complex rigid structural systems and provides a designer with a simple technique which requires minimal input to calculate the mass properties of a complex rigid structure and should be useful in any situation where one needs to calculate the center of gravity and moments of inertia of a complex structure. [76] [97]

This program is based on the premise that complex systems can be adequately described by a combination of elemental structural shapes. Thirteen widely used structural shapes are available in this program. They are as follows: Discrete Mass, Cylinder, Truncated Cone, Torus, Beam (arbitrary cross section), Circular Rod (arbitrary cross section), Spherical Segment, Sphere, Hemisphere, Parallelepiped, Swept Trapezoidal Panel, Symmetric Trapezoidal Panels, and a Curved Rectangular Panel. [97] [76]

Rigid body analysis is used to calculate mass properties. Mass properties are calculated about component axes which have been rotated to be parallel to the system coordinate axes. Then the system center of gravity is calculated and the mass properties are transferred to axes through the system center of gravity by using the parallel axis theorem. System weight, moments of inertia about the system origin, and the products of inertia about the system center of mass are calculated and printed. From the information about the system center of mass the principal axes of the system and the moments of inertia about them are calculated and printed. [76] [97]

Geometric data describing size and location of each element and the respective material density or weight of each element are the only required input data. [97] [76]

8) Predicting Subsonic or Supersonic Linear Potential Flows about Arbitrary Configurations Using a Higher Order Panel Method (PANAIR)

PANAIR is the definitive subsonic/supersonic panel method based on linear aerodynamic theory. PANAIR calculates flow properties about arbitrary three-dimensional configurations. The program uses a higher-order panel method to solve the linearized potential flow boundary-value problem at subsonic and supersonic Mach numbers. [78] [102] [103]

Generally speaking, a panel method solves a linear partial differential equation numerically by approximating the configuration surface by a set of panels on which unknown "singularity strengths" are defined, imposing boundary conditions at a discrete set of points, and thereby generating a system of linear equations relating the unknown singularity strengths. These equations are solved for the singularity strengths which provide information on the properties of the flow. [78]

A "higher order" panel method means that the singularity strengths are not constant on each panel. The potential for numerical error is greatly reduced in the PANAIR program by requiring the singularity strength to be continuous. It is also this "higher order" attribute which allows PANAIR to be used to analyze flow about arbitrary configurations. PANAIR can handle the simple configurations considered in the preliminary design phase and later serve as the "analytical wind tunnel" which can analyze the flow about the final detailed, complex configurations. [78]

PANAIR includes the following capabilities: [78]

- the ability to handle, within the limitations of linear potential flow theory, completely arbitrary configurations, using either exact or linearized boundary conditions
- the ability to handle asymmetric configurations as well as those with one or two planes of symmetry

- the ability to handle symmetric configurations in either symmetric or asymmetric flow
- the ability to calculate pressures, forces, and moments using a variety of pressure formulas (such as isentropic, linear, etc.), including the forces and moments due to flow through the surface

The aerodynamic solution provides surface flow properties (flow directions, pressures, and Mach number), configuration forces and moments, sectional forces and moments, and pressures. In addition, PANAIR calculates flow properties in the flow-field points and flow-field streamlines. Results are limited to subsonic and supersonic cases (transonic cases are excluded) with attached flow. In other words, PANAIR offers a comprehensive aerodynamic analysis and design capability for nearly arbitrary configurations in subsonic and supersonic flows. [102] [103]

Most problems can be modeled with a minimum of user input. In general, the aircraft surface is partitioned into several networks of surface grid points, such as a fore-body network, a wing network, etc. The coordinates of the input grid points must be computed and entered by the user; PANAIR does not generate grid point coordinates. PANAIR connects the grid points in each network with piecewise flat panels. The user also supplies information concerning the free-stream onset flow, the angle of attack, and the angle of sideslip. Numerous flow quantities are computed at points on the vehicle surface and at points in space. These include pressure coefficients, total and perturbation values of velocity and mass flux components, total and perturbation potential, local Mach number, and vacuum pressure coefficient. The pressure coefficients on individual panels are fitted with two dimensional quadratic splines and integrated to obtain the six components of force and the moment coefficients. These coefficients may be output for each panel, for columns of panels, for each network, or for any combination of networks. [78]

Panel codes generally generate the solution to problems in aerodynamics by superposition of elementary solutions. Panel codes have been superseded by Computational Fluid Dynamics (CFD) codes solving Euler or Navier-Stokes equations. However, there is still a tradeoff between the time spent setting up the input for a computational technique versus the accuracy of the method. [80]

a) PANAIR input pre-processor (PANIN)

Because the creation of an input file for PANAIR is error-prone, PANIN accepts a free-form file and creates a properly formatted input file for PANAIR. [79]

The input to PANAIR is described in [102]. The input data is organized in specific columns. The PANIN program was written to enable a user to select the flow properties and all other program options by editing a short free-format file called an auxiliary file. [79]

One entry in the auxiliary file is the name of a file which contains the geometrical information. The geometrical file is in the NASA standard format for wireframe geometry, i.e. in the LaWGS (Langley Wireframe Geometry Standard) format. The program reads the various items of control information from the auxiliary file and combines this information with the panel geometry in the LaWGS file to produce a combined file which is a properly formatted input file for PANAIR. [79]

The first and most difficult part of preparing a case for PANAIR or any panel code is the definition of the surface geometry as a set of quadrilateral panels. A variety of techniques exist for the creation of this data; for example a program named MAKEWGS (described in the chapter V.E.11)) is usable for definition of simple geometries. The ultimate solution lies in the use of a Computer-Aided-Design (CAD) system which has a wireframe output option. There is also a program named 2WGS (described in the chapter V.E.13)) which can convert PANAIR input to LaWGS. [79]

Once the LaWGS geometry file and the auxiliary file are created and stored, the PANAIR input file can be generated by invoking the program PANIN. [79]

After PANIN completes the execution, two files are produced. These files take their names from the data in the NAME record in the auxiliary file with extensions of .SH and .IN. The .IN file is the PANAIR input data. [79]

Although a considerable effort has been expended in making the program free of errors or omissions, the user should inspect the input file and script carefully. There are many problems in forcing data to fit the fixed field format and inadvertent round off may result. In this case, the PRECISION keyword can be helpful, but there are many potential sources of error. The principal problem area is overflow of an output field when one is trying to keep the geometric accuracy high by printing many decimals. Two popular places for this error to occur are the x-coordinates of the trailing edges of wakes and the value of reference area. It is usually useful to scan the PANAIR input file for asterisk characters. [79]

There are many options in the PANAIR input file and it would be virtually impossible to incorporate all of them in this program. For example, in PANAIR different reference lengths for yawing moment and rolling moment can be selected but PANIN simply requests a span. [79]

9) Digital Datcom

The Stability and Control Data Compendium (Datcom) provides a systematic summary of methods for estimating static stability, high lift and control, and dynamic derivative characteristics of a wide variety of aircraft and aircraft configurations. The estimation includes, for example, lift coefficients, drag coefficients, side force coefficients, roll moment coefficients, pitch moment coefficients, yaw moment coefficients, changes in coefficients due to power effects, and changes in coefficients due to ground effects. For any given flight condition and configuration the complete set of derivatives can be determined without resort to outside information. [2] [10] [106] [108] [86] [109] [52]

The program contains a trim option which computes control deflections and aerodynamic increments for vehicle trim at subsonic Mach numbers. Furthermore, user oriented features of the program include minimized input requirements, input error analysis, and various options for application flexibility. An interesting feature is also roughness factors for various types of surface such as polished metal, wood, natural

sheet metal, camouflage paint and other surfaces. [106] [108] [109]

The program has been developed as a modular basis. These modules correspond to the primary building blocks referenced in the program executive. The modular approach was used because it simplified program development, testing, and modification or expansion. [10] [107] [108]

Primarily intended for preliminary use, ahead of test data, it is designed to give an initial look at the stability performance of an aircraft design. By interfacing Datcom with the FDM, an aircraft model for any fixed-wing UAVs can be rapidly developed without wind tunnel testing. This feature significantly increases the repeatability of flight simulation and is found very useful for UAV preliminary designs where only a rough estimate of the vehicle's stability is required. However, program should not be intended for use instead of wind tunnel or flight test. [1] [2]

Digital Datcom is used to calculate aerodynamic coefficients from first principles. Datcom produces an output file with aerodynamic coefficients based on an input file containing all essential geometries of an aircraft. The coefficients in the six degrees of freedom are drag, lift, side, pitching moment, rolling moment, and yawing moment coefficient. The location of the center of gravity is uncertain because the material of the entire solid model is considered to be homogeneous. However, almost all similar programs have the same problem. [1] [10] [30]

Inputs to Datcom include desired flight conditions, aircraft attitudes, physical geometry, and desired outputs. Datcom treats inputs which represent a traditional wing-body-tail configuration and any control or high lift devices. However, some nonstandard geometry can be treated as well. Datcom inputs were assumed for straight-tapered and nonstraight-tapered wings including effects of sweep, taper, and incidence. For the longitudinal characteristics, the program assumes a mid-wing configuration. [2] [10] [11]

The effect of linear twist can be treated at subsonic Mach numbers. Dihedral influences are included in lateral-directional stability derivatives and wing wake location used in the calculation of longitudinal data. Airfoil section characteristics are a required input, although most of these characteristics may be generated using the Airfoil Section Module. [10] [108]

Users are advised to be mindful of section characteristics which are sensitive to Reynolds number, particularly in cases where very low Reynolds number estimates are of interest. A typical example would be pretest estimates for small, laminar flow wind tunnels where Reynolds numbers on the order of 100,000 are common. Datcom has low accuracy below this Reynolds number. [10] [109]

Users should be aware that Datcom employs turbulent skin friction methods in the computation of friction drag values. Estimates for cases involving significant wetted areas in laminar flow will require adjustment by the user. [10]

Datcom requires Mach number and Reynolds number to define the flight conditions. This requirement can be satisfied by defining combinations of Mach number, velocity, Reynolds number, altitude, and pressure and temperature. The speed reference is input as either Mach number or velocity, and the atmospheric conditions as either altitude or freestream pressure and temperature. The specific reference and atmospheric conditions are then used to calculate Reynolds number. [10]

The program may loop on speed reference and atmospheric conditions by using three different ways: [10]

- LOOP=1 - Mach and altitude changes. The program executes at the first Mach number and first altitude, then at the second Mach number and second altitude, and continues for all the flight conditions. In the input data, NMACH must equal NALT. NMACH flight conditions are executed. This option should be selected when the Reynolds number is input, and must be selected when atmospheric conditions are not input.
- LOOP=2 - Mach number changes at fixed altitude. The program executes using the first altitude and cycles through each Mach number in the input list, then using the second altitude and cycles through each Mach number, and continues until each altitude has been selected. Atmospheric conditions must be input for this option. NMACH times NALT flight conditions are executed.
- LOOP=3 - Altitude changes at fixed Mach number. The program executes using the first Mach number and cycles through each altitude in the input list, then using the second Mach number and cycles through each altitude, and continues until each Mach number has been selected. Atmospheric conditions must be input for this option. NMACH times NALT flight conditions are executed.

A diagnostic analysis module in Datcom scans all of the input data which listing is given and any errors are flagged. However, Datcom will attempt to execute all cases as input by the user even if errors are detected. [10]

The airfoil section module can be used to calculate the required geometric and aerodynamic input parameters for virtually any user defined airfoil section. This module substantially simplifies the user's input preparation. [10]

An airfoil section is defined by one of the following methods: [10]

- an airfoil section designation (for NACA, double wedge, circular arc, or hexagonal airfoils)
- upper and lower Cartesian coordinates
- mean line and thickness distribution

The airfoil section module uses Weber's method to calculate the inviscid aerodynamic characteristics. A viscous correction is applied to the lift curve slope, C_{L_alpha}. [10]

Five general characteristics of the module should be noted: [10]

- For subsonic Mach numbers, the module computes the airfoil subsonic section characteristics and the results can be considered accurate for Mach numbers less

than the crest critical Mach number. Near crest critical Mach number, flow mixing due to the upper surface shock will make the boundary layer correction invalid. Compressibility corrections also become invalid. The module also computes the required geometric variables at all speeds, and for transonic and supersonic speeds these are the only required inputs. Mach equals zero data are always supplied.

- Because of the nature of the solution, predictions for an airfoil whose maximum camber is greater than 6% of the chord will lose accuracy. Accuracy will also diminish when the maximum airfoil thickness exceeds approximately 12% of the chord, or large viscous interactions are present such as with supercritical airfoils.
- When section Cartesian coordinates or mean line and thickness distribution coordinates are specified, the user must adequately define the leading edge region to prevent surface curve fits which have infinite slope. This can be accomplished by supplying section ordinates at non-dimensional chord stations (x/c of 0.0, 0.001, 0.002, and 0.003).
- If the leading edge radius is not specified in the airfoil section input, the user must insure that the first and second coordinate points lie on the leading edge radius. For sharp nosed airfoils the user must specify a zero leading edge radius.
- The computational algorithm can be sensitive to the smoothness of the input coordinates. Therefore, the user should insure that the input data contains no unintentional fluctuations. Considering that Datcom procedures are preliminary design methods, it is at least as important to provide smoothly varying coordinates as to accurately define the airfoil geometry.

Several operational limitations exist in Datcom. These limitations and some pertinent operational techniques are listed below without extensive discussion or justification: [10]

- The forward lifting surface is always input as the wing and the aft lifting surface as the horizontal tail. This convention is used regardless of the nature of the configuration.
- Twin vertical tail methods are only applicable to lateral stability parameters at subsonic speeds.
- Airfoil section characteristics are assumed to be constant across the airfoil span, or an average for the panel. Inboard and outboard panels of cranked or double-delta planforms can have their individual panel leading edge radii and maximum thickness ratios specified separately.
- If airfoil sections are simultaneously specified for the same aerodynamic surface by an NACA designation and by coordinates, the coordinate information will take precedence.
- Jet and propeller power effects are only applied to the longitudinal stability parameters at subsonic speeds. Jet and propeller power effects cannot be applied simultaneously.
- Ground effect methods are only applicable to longitudinal stability parameters at subsonic speeds.
- Only one high lift or control device can be analyzed at a time. The effect of high lift and control devices on downwash is not calculated. The effects of multiple devices can be calculated by using the experimental data input option to supply the effects of one device and allowing Datcom to calculate the incremental effects of the second device.
- Jet flaps are considered to be symmetrical high lift and control devices. The methods are only applicable to the longitudinal stability parameters at subsonic speeds.
- The program uses the input names to define the configuration components to be synthesized. For example, the presence of name list HTPLNF causes Datcom to assume that the configuration has a horizontal tail.

Datcom was tested, for example, on Rascal UAV [2], Shadow UAV [58], and a MAV prototype [109].

a) Datcom Release 2 and OpenDatcom

Some aspects of Datcom are outdated; for example the user interface, the use of the DOS command prompt, and the input file written in a text format for which the standard rules of FORTRAN apply. With these restrictions, it is time consuming to compile and troubleshoot the input file. [86]

OpenDatcom and Datcom Release 2 (DR2) have been specifically developed to remove these outdated features. Both are written completely in Java SE and uses the Java Virtual Machine (JVM) to interface with Datcom. The interface is completely coded into OpenDatcom and DR2; as a result, no modification to the original Datcom code is needed. [86]

OpenDatcom and DR2 use basic GUI to allow the user to easily compile an input file, import an existing input file, and run Datcom without any knowledge of DOS or of the FORTRAN formatting of the input file. [86]

Another option which was added to DR2 is the compiling and export of stability and performance coefficients. This feature was added with the specific intention of compiling three dimensional stability tables which can be copied straight into the FlightGear (using the JSBsim). This allows that an aircraft can be analyzed in Datcom and its flight can be tested in FlightGear. [86]

However, DR2 is in BETA development stage and may have some bugs; as a result, DR2 should be used cautiously. [86]

The graphical user interface of OpenDatcom can be seen in Fig. 28, of Datcom Release 2 in Fig. 29, and of Datcom-to-JSBSim application in Fig. 30.

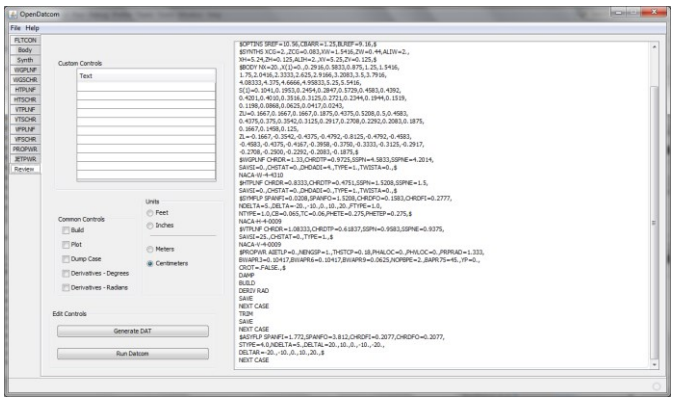

Fig. 28. The graphical user interface of OpenDatcom

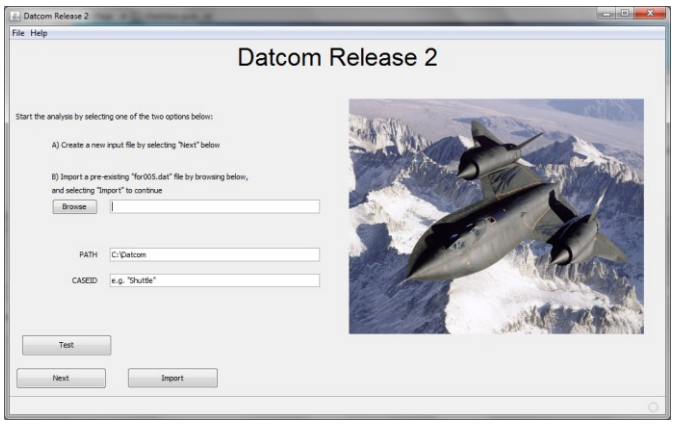

Fig. 29. The graphical user interface of Datcom Release 2

Fig. 30. The graphical user interface of Datcom-to-JSBSim

b) Datcom+

Datcom+ is an extension of the Datcom program and incorporates some tools to make it easier to use. Front-end and back-end is added to the original Datcom for user convenience. By adding a different format output section to the original program, the output data is in various formats: [85] [109]

- Free-format LFI tables, for plotting with LFIPLOT
- XML format, compatible with JSBSim
- AC3D Model

However, there are some known issues with DATCOM+. The first is the defining of airfoils manually with upper and lower surface points which does not provide any output for the AC3D picture. The second are fuselages which are not drawn correctly if defined as other than a circular cross-section. [109]

Datcom+ Pro is the next generation of this program, which has been made much more user-friendly. Visualization tools allow user to see his aircraft immediately, and coefficient data generated by the Datcom program is plotted on X-Y graphs for ease of interpretation and inclusion into reports. Additionally, Datcom+ Pro model can now be run in JSBSim and sample flight test scripts are provided to execute standard flight test maneuvers. However, Datcom+ Pro is not available for free. [85]

10) Aeroelastic Analysis for Rotorcraft in Flight or in a Wind Tunnel (ROTOR)

The testing of rotorcraft, either in flight or in a wind tunnel, requires a consideration of the coupled aero-elastic stability of the rotor and airframe, or the rotor and support system. Even if the primary purpose of a test is to measure rotor performance, ignoring the question of dynamic stability introduces the risk of catastrophic failure of the aircraft. [81]

This computer program was developed to incorporate an analytical model of the aero-elastic behavior of a wide range of rotorcraft. Such an analytical model is desirable for both pretest predictions and posttest correlations. The program is also applicable in investigations of isolated rotor aero-elasticity and helicopter flight dynamics and could be employed as a basis for more extensive investigations of aero-elastic behavior, such as automatic control system design. [81]

The program incorporates an analytical model which is applicable to a wide range of rotors, helicopters, and operating conditions. The equations of motion used in the model were derived using an integral Newtonian method, which provides considerable insight into the blade inertial and aerodynamic forces. The rotor model includes coupled flap-lag bending and blade torsion degrees of freedom, and is applicable to articulated, hinge-less, gimbaled, and teetering rotors with an arbitrary number of blades. The aerodynamic model is valid for both high and low inflow, and for both axial and non-axial flight. Rotor rotational speed dynamics, including engine inertia and damping, and perturbation inflow dynamics are included in the aerodynamic model. [81]

For a rotor on a wind-tunnel support, a normal mode representation of the test module, strut, and balance is used. The aero-elastic analysis for rotorcraft in flight is applicable to a general two-rotor aircraft, including single main-rotor and tandem helicopter configurations, and side-by-side or tilting prop-rotor aircraft configurations. The rotor model includes rotor-rotor aerodynamic interference and ground effect. The aircraft model includes rotor-fuselage-tail aerodynamic interference, engine dynamics, and control dynamics. A constant-coefficient approximation is used for non-axial flow and a quasistatic approximation is used for the low frequency dynamics. The coupled system dynamics results are a set of linear differential equations which are used to determine the stability and aero-elastic response of the system. [81]

Unfortunately, this program is a “work in progress” and is not ready for general release. However, source codes may be seen in [81].

11) Wireframe generator (MAKEWGS)

Many computing procedures in engineering require the definition of a surface or solid object by means of an ordered lattice of points which define a grid of quadrilaterals. MAKEWGS can create wireframe models of simple wings and bodies and enables to make some of the classic shapes of aerodynamic theory with a minimum of effort. [87]

A script or input file describes the objects to be defined. The program reads the script and creates an output file with the grid points in NASA Langley Wireframe Geometry Standard format (LaWGS). [87]

Wings are defined by their root and tip chords and their grid densities. Several common airfoil sections are available. Bodies are defined by nose length, after-body length and overall length and maximum diameter. The common body shapes such as parabolic, conical, Sears-Haack, von Karman Ogive, and ellipsoidal are coded. [87]

12) 3-VIEW and SILHOUETTE

SILHOUETTE gives perspective views of an arbitrary configuration defined by wireframe meshes of grid points with hidden line removal. 3-VIEW produces plan, side, and rear views from the same input file (LaWGS) as SILHOUETTE uses. [9]

a) 3-VIEW

For many applications, it is simple and fast to make views of the configuration in the plan, side, and rear or front view. Even though hidden lines are shown, they are not usually as confusing as they are in isometric views. An example of an aircraft in the plan view can be seen in Fig. 31. [88]

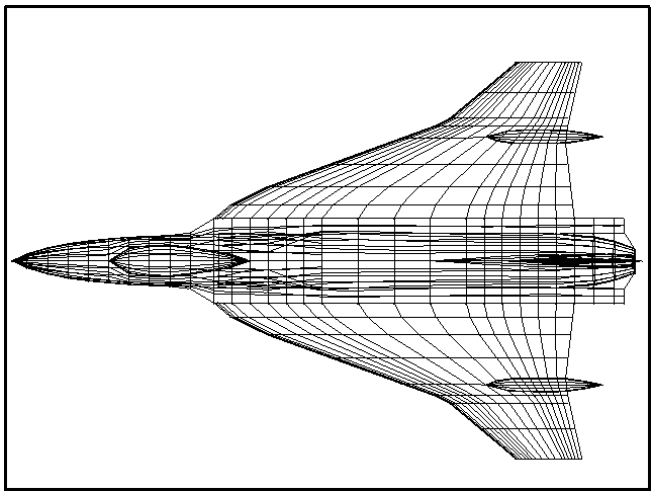

Fig. 31. The plan view of an aircraft [88]

This simple program takes a configuration in LaWGS format and produces several output files which may be used to visualize your vehicle. The files plan.gnu, side.gnu, and rear.gnu are formatted to be displayed with Gnuplot [104] or any other plotting package. [88]

b) SILHOUETTE - Hidden Line Program (HLP)

This program draws a perspective view of an object which has been defined as a wire frame and analyzes the image to remove the hidden lines. [89]

A polygonal representation of objects, even with hidden lines removed, is not always desirable. A more pleasing pictorial representation often can be achieved by removing some of the remaining visible lines, thus creating silhouettes (or outlines) of selected surfaces of the object. Additionally, it should be noted that this silhouette feature allows warped polygons, i.e. any polygon can be decomposed into constituent triangles. The consideration that these triangles are members of the same family will result in a polygon with no interior lines, and thus, the restriction of flat polygons will be removed. [89]

SILHOUETTE is a program for calligraphic drawings which can render any subset of polygons as a silhouette with respect to itself. The program is flexible enough to be applicable to every class of an object. SILHOUETTE offers all possible combinations of silhouette and non-silhouette specifications for an arbitrary solid. Thus, it is possible to enhance the clarity of any three-dimensional scene presented in two dimensions. [89]

Input to the program can be line segments or polygons. Polygons designated with the same number will be drawn as a silhouette of those polygons. The output is a plot file, encoded for Gnuplot [104], of the object in question. [89]

13) Geometry Conversion to LaWGS (2WGS)

PANAIR (described in the chapter V.E.8)) allows the user to define the geometry of a vehicle and reads a wire frame mesh as part of its input file. PANAIR was developed before the definition of the LaWGS and have their own input schemes. As a result, 2WGS package with utility program which converts the input files for PANAIR (and for other programs in PDAS; e.g. WINGBODY) into the LaWGS format was developed. A program named a5022wgs converts the input file of PANAIR into a file with extension of .WGS. The converted file may be then inserted to the SILHOUETTE, 3-VIEW, and other viewers. [90]

14) Geometry Conversion to VRML World (VRML)

This program makes a VRML model (a .WRL file) from wireframe geometry in LaWGS format. This is an experimental program in the initial phases of testing and produces a file in VRML 1.0 format. The idea is to display the data (input and output) from a general class of CFD programs. [91]

The program asks for the name of the input file. This must be a file in LaWGS format. After reading the input data, the program produces a file with extension of .WRL which may be used as input to a VRML browser. [91]

F. Computational Fluid Dynamics (CFD)

Computational Fluid Dynamics (CFD) provides a qualitative (and sometimes even quantitative) prediction of fluid flows by means of: [137]

- Mathematical modelling (partial differential equations)

- Numerical methods (discretization and solution techniques)
- Software tools (solvers, pre- and post-processing utilities)

In other words, CFD uses numerical methods to solve how liquids and gases interact with surfaces. [143] [51]

1) Stanford University Unstructured (SU^2)

The Stanford University Unstructured (SU^2) suite is an open-source collection of C++ based software tools. This computational analysis and design software collection is being developed to solve complex, multi-physics analysis and optimization tasks using arbitrary unstructured meshes, to perform Partial Differential Equation (PDE) analysis and solve PDE constrained optimization problems. The toolset is designed with computational fluid dynamics and aerodynamic shape optimization in mind, but is extensible to treat arbitrary sets of governing equations such as potential flow, electrodynamics, chemically reacting flows, and many others. SU^2 is under active development in the Aerospace Design Lab (ADL) of the Department of Aeronautics and Astronautics at Stanford University, and is released under an open-source license. [138] [139]

The SU^2 software suite specializes in high-fidelity PDE analysis and in the design of PDE-constrained systems on unstructured meshes. The suite itself is composed of several C++ analysis modules which handle specific jobs, including: [139]

- SU2_CFD - The main PDE solution module which started primarily as an Euler and RANS CFD solver, but has been modified to treat many other governing equations, including the adjoint equations for many of the supported governing equation systems.
- SU2_DDC - The Domain Decomposition Code, used to prepare SU^2 for computations involving multiple processors.
- SU2_MAC - The Mesh Adaptation Code which can be used to refine the unstructured computational meshes to improve the accuracy of the predictions.
- SU2_GPC - The Gradient Projection Code which allows for the calculation of sensitivities for use in optimization and uncertainty quantification.
- SU2_MDC - The Mesh Deformation Code which can be used to perturb an existing unstructured volume mesh to conform to new surface geometries dictated by either shape optimization processes or aero-structural simulations.
- SU2_PBC - The Periodic Boundary Code, a pre-processor to allow for the solution of PDEs on periodic domains.
- SU2_SMC - The Sliding Mesh Code, a pre-processor which enables the solution of PDEs on meshes which slide (translational or rotational capabilities included) past each other.

Additional modules may be added as further capabilities are needed and included in the software. This structure makes SU^2 an ideal tool for performing multi-physics simulations, including multi-species thermochemical non-equilibrium flow analysis, combustion modelling, two-phase flow simulations, magneto-hydrodynamics simulations, and other simulations. [139]

The SU^2 software suite was conceived as a common infrastructure for solving Partial Differential Equation (PDE) problems, using the Finite Volume Method (FVM) or Finite Element Method (FEM). The code structure and the high-level time and spatial integration structure is shared by all of the applications, and only specific numerical methods for the convective, viscous and source terms are re-implemented for different models where necessary. There is no fundamental limitation on the number of state variables or the number of governing equation systems which can be solved simultaneously in a coupled or segregated way (other than the physical memory available on a given computer architecture), and the more complicated algorithms and numerical methods, including parallelization, multigrid and linear solvers, have been implemented in such a way that they can be applied without special consideration during the implementation of a new physical model. [139]

Several forms of the Reynolds-averaged Navier-Stokes (RANS) equations have also been implemented in SU^2; for instance compressible, incompressible, Arbitrary Lagrangian-Eulerian, etc. Moreover, both the laminar Navier-Stokes and Euler equations are also available in the code as subsets of the RANS equations by disabling turbulence modelling and, respectively, by completely removing viscosity. [139]

Numerical discretization of the governing fluid dynamic equations using a conservative formulation often results in excess artificial viscosity at low Mach numbers. This degrades the performance of a compressible solver in regions of low Mach number flow. Preconditioning techniques such as Roe-Turkel have been developed for solving nearly incompressible flow problems using the same numerical methods developed for compressible flows. This can be particularly useful when only part of a flow field is essentially incompressible. For example, flow over a multi-element airfoil at high angles of attack has regions of both compressible and incompressible flow. [139]

SU^2 is built to enable vertical integration with optimizers and to reduce the amount of user overhead required for setup. There are five levels of components in the optimization control architecture, and most rely on Python scripts to modify the configuration input, execute lower-level components and post-process any resulting data. To simplify and shorten overhead time during problem setup, all levels start from a common configuration file, which is modified as needed when passed to lower levels. Listed in order from lowest to highest, these levels are: [139]

- Core tools
- Solution decomposition/re-composition
- Sensitivity analysis

- Design evaluation
- Design optimization

A set of tutorials which cover all the basic capabilities of SU^2 was created and is distributed with SU^2. There can be found, for example, the flow and adjoint simulation of external, inviscid flow around a 2D geometry (NACA 0012 airfoil) by using steady, 2D, Euler and continuous Adjoint Euler equations [140], and optimal shape design of a rotating airfoil [141]. [139]

Moreover, compressible RANS simulations, low-Mach number simulations, airfoil and fixed wing optimization, wing design using RANS equations, redesign of a rotor in hover (Fig. 32), Adaptive Mesh Refinement, goal-oriented mesh adaptation, engine propulsion effect adaptation and other applications of SU^2 are described in [139].

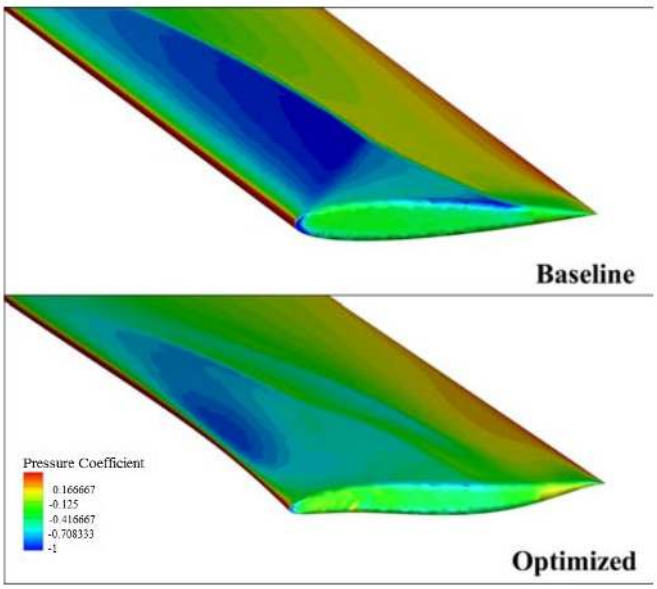

Fig. 32. A comparison of a baseline and optimized rotor geometries [139]

2) OpenFOAM

The OpenFOAM (Open Field Operation and Manipulation) CFD Toolbox is a free, open source CFD software package which has a large user base across most areas of engineering and science, from both commercial and academic organizations. For example, in [149], OpenFOAM was used for simulation of flow around flapping wings, and in [143] as a part of system for optimization of wing, body, and tail of aircraft. [142]

OpenFOAM has an extensive range of features to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to solid dynamics and electromagnetics. It includes tools for meshing, notably snappyHexMesh, a parallelized mesher for complex CAD geometries, and for pre- and post-processing. Almost everything (including meshing, and pre- and post-processing) runs in parallel as standard, enabling users to take full advantage of computer hardware at their disposal. [142] [143] [144]

The core technology of OpenFOAM is a flexible set of efficient C++ modules, used primarily to create executables, known as applications – overview of OpenFOAM structure can be seen in Fig. 33. The applications fall into two categories: solvers, which are each designed to solve a specific problem in continuum mechanics; and utilities, that are designed to perform tasks which involve data manipulation. OpenFOAM includes over 80 solver applications which simulate specific problems in engineering mechanics and over 170 utility applications which perform pre- and post-processing tasks, e.g. meshing, data visualization, etc. [144] [142]

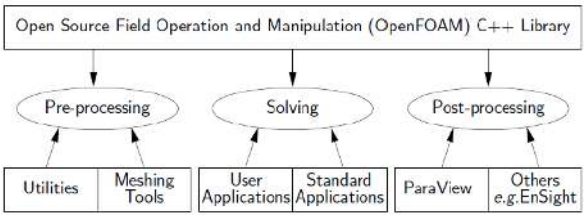

Fig. 33. The overview of the OpenFOAM structure [144]

An extensive set of OpenFOAM solvers has evolved. Despite OpenFOAM is used mainly for CFD, it has found use in other areas such as stress analysis, electromagnetics and finance because; it is fundamentally a tool for solving partial differential equations rather than a CFD package in the traditional sense. OpenFOAM has standard solvers for: [145]

- Basic CFD codes (Laplace, potential flow, and scalar transport solvers)
- Incompressible flow
- Compressible flow
- Multiphase flow
- Direct Numerical Simulation (DNS) and Large Eddy Simulation (LES)
- Combustion
- Particle-tracking flows
- Heat transfer and buoyancy-driven flows
- Molecular dynamics methods
- Direct simulation Monte Carlo methods
- Electromagnetics
- Stress analysis of solids
- Finance

Moreover, OpenFOAM offers ODE solvers for non-stiff systems including: fifth-order Cash-Karp Runge-Kutta with error estimation and adaptive time step control. [146]

OpenFOAM also offers ODE solvers for stiff systems including: the fourth-order semi-implicit Runge-Kutta scheme of Kaps, Rentrop and Rosenbrock with error estimation and adaptive time step control, and the semi-emplicit Bulirsch-Stoer method of Bader and Deuflhard. [146]

OpenFOAM contains a suite of numerical tools to solve a range of problems in engineering and science. It includes methods to solve problems where matter is represented as a continuum and where it is represented by discrete particles. To solve equations for a continuum, OpenFOAM uses a numerical approach with the following features: [147]

- Segregated, iterative solution - For the system of equations governing our problem of interest, separate matrix equations are created for each equation, and are solved within an iterative sequence (as opposed to created one, big matrix equation for the entire system of equations).
- Finite volume method - Matrix equations are constructed using the finite volume method applied to arbitrary shaped cells (any number of faces, any number of edges).
- Collocated variables - The solution variable for each matrix equation is defined at cell centers.
- Equation coupling - The coupling between equations, particularly pressure and velocity is performed using adapted versions of well-known algorithms such as e.g. PISO and SIMPLE.

OpenFOAM contains a range of dynamic mesh functionality within a set of libraries that is plugged into a range of dynamic mesh solvers. The type of dynamic mesh functionality includes the following: [148]

- Solid body motion of a mesh according to prescribed motion function, e.g. sloshing in a tank.
- Internal motion (e.g. distortion) of a mesh calculated from boundary motion, e.g. object floating at a free surface.
- Dynamic refinement/unrefinement of hex meshes, e.g. about a fluid interface, shock, etc.
- Prescribed motion of a mesh, e.g. according to a periodic boundary motion.

There are also a number of tools to specify the boundary motion in conjunction with mesh motion: [148]

- Prescribed six degree of freedom (6DOF) motion functions, e.g. translations and rotations.
- Tabulated 6DOF motion which interpolates discrete data.
- Specialized Ship design analysis (SDA) 3DOF motion function.
- 6DOF motion caused by flow, e.g. floating object, which can also permit constraints (fixed points) and restraints (springs and dampers).

3) Code_Saturne

Code_Saturne is open-source CFD software which solves the Navier-Stokes equations for 2D, 2D-axisymmetric and 3D flows, steady or unsteady, laminar or turbulent, incompressible or weakly dilatable, isothermal or not, with scalars transport if required. [150] [151]

Several turbulence models are available, from Reynolds-Averaged models (RANS models) to Large-Eddy Simulation models (LES models). In addition, a number of specific physical models are available as modules: gas, coal, biomass, pollutant, and heavy-fuel oil combustion, semi-transparent radiative transfer, particle-tracking with Lagrangian modelling, Joule effect, electric arcs, multi-physics modelling of arc welding, weakly compressible flows, atmospheric flows, rotor/stator interaction for hydraulic machines. [150] [151]

Code_Saturne has been under development since 1997 by EDF R&D (Electricité de France). The software is based on a co-located Finite Volume Method (FVM) that accepts three-dimensional meshes built with any type of cell (tetrahedral, hexahedral, prismatic, pyramidal, and polyhedral) and with any type of grid structure (unstructured, block structured, hybrid). [152] [151]

Code_Saturne is composed of two main elements and an optional GUI (as shown in Fig. 34): [151]

- The Kernel module which is the numerical solver
- The Preprocessor module which is in charge of mesh import

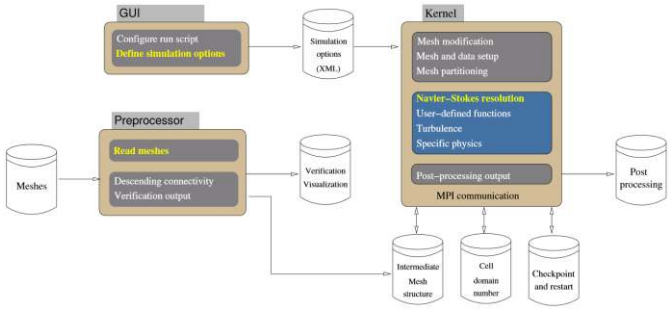

Fig. 34. The Code_Saturne elements [151]

Code_Saturne can use different numerical methods: [153]

- Discretization
- Velocity-pressure coupling
- Linear system resolution (Jacobi (default for velocity, temperature, turbulent variables, passive scalars), algebraic multigrid (default for pressure), conjugate gradient, and stabilized bi-conjugate gradient (BI-CGSTAB))
- Convective scheme (First order Upwind Scheme, Centered scheme, Second Order Linear Upwind (SOLU) Scheme, and Blended scheme between upwind and second order scheme)
- Gradient calculation

The supported compatible mesh generators' formats include: SALOME SMESH, I-DEAS Nx, Gmsh, Gambit (Fluent), Simail, Harpoon, ICEM-CFD, and Star-CCM+. [154]

Code_Saturne also relies on the PLE (Parallel Location and Exchange) library for the management of code coupling; this library can also be used independently. [151]

Code_saturne can be coupled with: [155]

- itself (in order to couple different models (RANS/LES), to model fluid-structure interaction with large displacement or the rotating machines)
- Code_Aster (in order to model the fluid-structure interaction)
- SYRTHES (in order to model the conjugate heat transfer)

4) High Fidelity Large Eddy Simulation (HiFiLES)

High-order numerical schemes may have the potential to advance CFD beyond the current plateau of second-order methods and RANS turbulence modelling, ushering in new levels of accuracy and computational efficiency in turbulent flow simulations. [156] [160]

Because of new aircraft roles (e.g. very small or large concepts, Reynolds numbers 10^4–10^7, very high or low altitude, Mach numbers between ca. 0.01–1.0, quiet vehicles, low fuel consumption vehicles, etc.), the need for high-fidelity simulation techniques to predict their performance is growing; furthermore, revolutionary aircraft design concepts may appear in the near future. As a result, high-order numerical methods may find their place in the aeronautical industry. [157] [50] [160]

Unsteady simulations, including those of flapping wings, wake capturing, noise prediction, and turbulent flows via Large Eddy Simulation (LES), are just a few examples of computations that could benefit from high-order numerical methods. In particular, high-order methods have a significant edge in applications that require accurate resolution of the smallest scales of the flow. Such situations include the generation and propagation of acoustic noise from an airframe, or at the limits of the flight envelope where unsteady, vortex-dominated flows have a significant effect on aircraft performance. On a given grid, utilizing a high-order representation enables smaller scales to be resolved with a greater degree of accuracy than standard second-order methods. Furthermore, high-order methods are inherently less dissipative, resulting in less unwanted interference with the correct development of the turbulent energy cascade. [157] [160]

HiFiLES is open-source software, written in C++. HiFiLES is high-order numerical methods for flow simulations capture complex phenomena like vortices and separation regions using fewer degrees of freedom than their low-order counterparts. The High Fidelity (HiFi) provided by the schemes, combined with turbulence models for small scales and wall interactions, gives rise to a powerful LES software package. [156]

HiFiLES is compressible flow solver for unstructured grids built from the ground up to take full advantage of parallel computing architectures. In general, the code is designed as an ideal base for further development on a variety of architectures; for example, it is especially well-suited for Graphical Processing Unit (GPU) architectures. The code uses the MPI protocol to run on multiple processors, and CUDA to harness GPU performance. [156] [157]

HiFiLES v. 0.1 contains the following capabilities: [156] [157]

- High-order compressible Navier-Stokes and Euler equations solver in 2D and 3D with support for triangular, quadratic, hexahedral, prismatic, and tetrahedral elements. Implementation for spatial orders of accuracy 2 through 4 has been verified.
- Numerical scheme: Energy-Stable Flux Reconstruction.
- Time advancement: explicit time-stepping with low-storage RK45 method (4^{th} order) or forward Euler (1^{st} order). Local time-stepping when running on CPUs.
- Boundary conditions: Wall: no-slip isothermal, no-slip adiabatic, and symmetry (slip wall). Inflow and outflow: characteristic, supersonic, subsonic. Periodic.
- High-order surface representation.
- Mesh format compatibility: neutral (.neu) and Gmsh (.msh).
- Large Eddy Simulation: Sub-grid Scale Models: Smagorinsky, WALE, similarity, and combinations of these. Wall models: log-law, three-layer Breuer-Rodi.
- Parallelization: MPI, and GPU (strong scalability 88% of ideal for up to 16 GPUs; weak scalability above 90% of ideal for up to 16 GPUs).

In [157], the SD7003 airfoil flows with *Re* = 10,000; 22,000; 60,000 (Fig. 35); and other simulation cases performed by using HiFiLES are described.

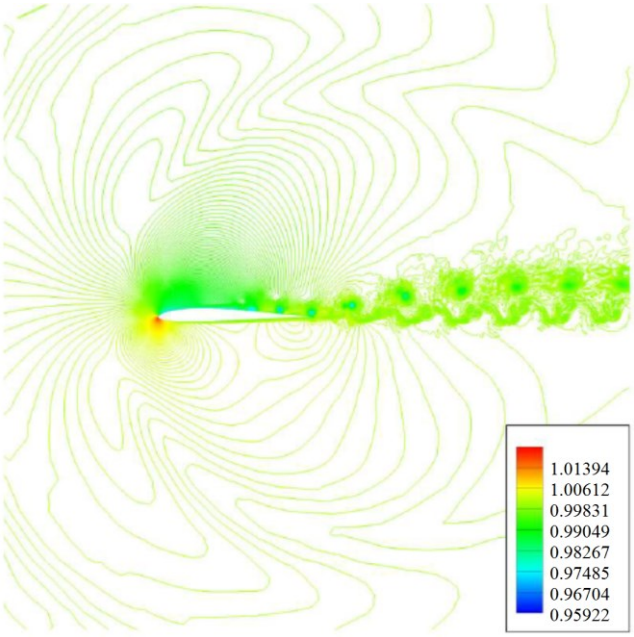

Fig. 35. A density contour for the flow with Re = 60,000 around the SD7003 airfoil [157]

5) PyFR

PyFR is an open-source Python based framework for solving advection-diffusion type problems on streaming architectures using the Flux Reconstruction approach of Huynh. The framework is designed to solve a range of governing systems on mixed unstructured grids containing various element types. PyFR is also designed to target a range of hardware platforms via use of an in-built domain specific language derived from the Mako template engine. The current release (v. 0.2.4) has the following capabilities: [158] [159] [160]

- Governing equations - Euler, Navier Stokes
- Dimensionality - 2D, 3D
- Element types - Triangles, Quadrilaterals, Hexahedra, Prisms, Tetrahedra, Pyramids
- Platforms - CPU clusters, Nvidia GPU clusters, AMD GPU clusters
- Spatial discretization - High-order flux reconstruction
- Temporal discretization - Explicit Runge-Kutta
- Precision - Single, Double
- Mesh files read - Gmsh (.msh)
- Solution files produced - Unstructured VTK (.vtu)

PyFR is being developed in the Vincent Lab, Department of Aeronautics, Imperial College London, UK. However, PyFR is not currently a fully-fledged production flow solver; in addition, no level of support is guaranteed. [158]

PyFR aims to expand the industrial CFD envelope from its current RANS plateau; enabling affordable and accurate simulation of currently intractable unsteady flow problems via scale resolving approaches such as LES. As a result, it is envisaged that PyFR may have significant impact in a range of application areas including design of next-generation unmanned aerial vehicles, aircraft noise reduction, design of jet engines, and other areas. [159]

G. Computer-Aided Design (CAD)

Computer Aided Design (CAD) is a set of methods and tools to assist product designers in creating a geometrical representation of the artifacts, in dimensioning, configuration management, archiving, exchanging part and assembly information between organizations, feeding subsequent design steps (analysis and manufacturing – Computer-Aided Engineering (CAE) and Computer-Aided Manufacturing (CAM)) by means of a computer system. These applications came out of the manufacturing world; thus, they have certain characteristics which make them well suited for manufacturing. [161] [177]

1) FreeCAD

FreeCAD is a fully multi-platform, open-source, general-purpose, parametric 3D CAD/CAE modeler. FreeCAD is aimed directly at mechanical engineering and product design; however it also fits in a wider range of uses around engineering, such as architecture. [162] [163] [131] [166]

FreeCAD's focus is to allow user to create high-precision 3D models, to keep tight control over those models (being able to go back into modelling history and change parameters), and eventually to build those models (via 3D printing, CNC machining or even construction worksite). [163]

FreeCAD has many 2D components in order to sketch 2D shapes or extract design details from the 3D model to create 2D production drawings; Nevertheless, direct 2D drawing (like AutoCAD LT) is not the focus, neither animation or organic shapes are (like in Maya, 3ds Max, Blender, or Cinema 4D). [162]

FreeCAD features tools are similar to Catia, SolidWorks or Solid Edge; as a result it also falls into the category of MCAD, PLM, CAx and CAE. Key features of FreeCAD are: [164] [162] [166]

- A complete Open CASCADE Technology-based geometry kernel
- A full parametric model
- A modular architecture which makes to provide additional functionality without modifying the core system
- Import/export to standard formats, such as STEP, IGES, OBJ, STL, DXF, SVG, STL, DAE, IFC or OFF, NASTRAN, VRML, in addition to FreeCAD's native FCSTD file format
- A Sketcher with constraint-solver, allowing to sketch geometry-constrained 2D shapes
- A Robot simulation module which allows to study robot movements
- A Drawing sheets module which permit to put 2D views of 3D models on a sheet
- A Rendering module which can export 3D objects for rendering with external renderers
- An Architecture module

One of the most powerful features of FreeCAD is the scripting environment. From the integrated python console (or from any other external Python script), almost any part of FreeCAD can be accessed; for example it may create or modify geometry, modify the representation of objects in a 3D scene, or access and modify the FreeCAD interface. Python scripting can also be used in macros, which provide an easy method to create custom commands. [163] [131] [166]

The main concept behind the FreeCAD interface is that it is separated into workbenches. A workbench is a collection of tools suited for a specific task, such as working with meshes, or drawing 2D objects, or constrained sketches. The current workbench can be switched with the workbench selector at any time. The tools included in each workbench, add tools from other workbenches, or even self-created tools (macros) can be customized. There is also a generic workbench which gathers the most commonly used tools from other workbenches, called the complete workbench. [163]

The selection of a workbench depends on the type of job which is needed to be done; for instance, PartDesign Workbench is focused on mechanical models or more generally any small-scale objects, Draft Workbench works in 2D, as well as Sketcher Workbench which, in addition, uses constraints; there is also Arch Workbench, special Ship Workbench, OpenSCAD Workbench and other Workbenches. [163]

However, FreeCAD is still in the early stages of development; thus, although it already offers a large list of features, many of them is still missing, specially comparing it to commercial solutions. Nevertheless, there is a fast-growing community of enthusiastic users, and it can already be found many examples of quality projects developed with FreeCAD. [163]

For example, the tutorial of the creation of a very simple, elemental airplane model in Part Workbench is described in [165]; more complicated geometry is shown in [131], where a particular aircraft configuration for the Long Endurance Electric UAV (LEEUAV), which can be seen in Fig. 36, was designed by using FreeCAD and developed scripts.

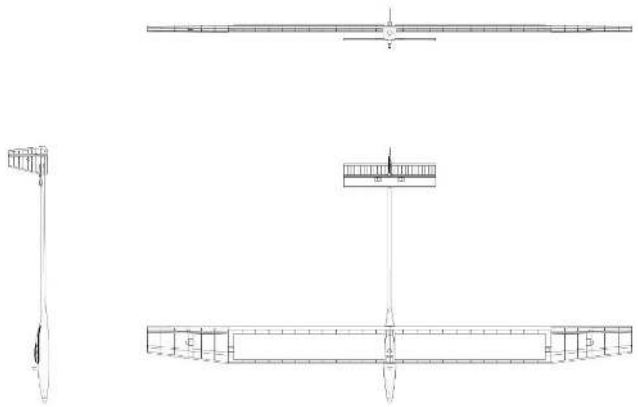

Fig. 36. LEEUAV designed with FreeCAD [131]

2) SALOME

SALOME is open-source software which provides a generic platform for Pre- and Post-Processing for numerical simulation. It is based on an open and flexible architecture made of reusable components. [222] [223]

SALOME can be used as standalone application for generation of CAD model, its preparation for numerical calculations and post-processing of the calculation results. Moreover, SALOME can be used as a platform for integration of the external third-party numerical codes to produce a new application for the full life-cycle management of CAD models. [222] [223]

SALOME can: [222] [223]

- Support interoperability between CAD modelling and computation software (CAD-CAE link)
- Make easier the integration of new components into heterogeneous systems for numerical computation
- Set the priority to multi-physics coupling between computation software
- Provide a generic user-friendly and efficient user interface, which helps to reduce the costs and delays of carrying out the studies
- Reduce training time to the specific time for learning the software solution based on this platform
- Provide access to all functionalities via the integrated Python console

The following operations can be done with SALOME: [222] [223]

- Create/modify, import/export (IGES, STEP, BREP), repair/clean CAD models
- Mesh CAD models, edit mesh, check mesh quality, import/export mesh (MED, UNV, DAT, STL)
- Handle physical properties and quantities attached to geometrical items
- Perform computation using one or more external solvers (coupling)
- Display computation results (scalar, vector)
- Manage studies (create, save, reload)

3) BRL-CAD

BRL-CAD is a powerful cross-platform open-source combinatorial Constructive Solid Geometry (CSG) solid modelling system which includes interactive 3D solid geometry editing, high-performance ray-tracing support for rendering and geometric analysis, network-distributed framebuffer support, image and signal-processing tools, path-tracing and photon mapping support for realistic image synthesis, a system performance analysis benchmark suite, an embedded scripting interface, and libraries for robust high-performance geometric representation and analysis. [167] [168] [169]

BRL-CAD is a collection of more than 400 tools, utilities, and applications comprising more than a million lines of source code. For more than 20 years, BRL-CAD has been the primary tri-service solid modelling CAD system used by the U.S. military to model weapons systems for vulnerability and lethality analyses. The solid modelling system is frequently used in a wide range of military, academic, and industrial applications including in the design and analysis of vehicles, mechanical parts, and architecture. The package has also been used in radiation dose planning, medical visualization, computer graphics education, CSG concepts and modelling education, and system performance benchmark testing among other purposes. [167] [168] [169]

BRL-CAD supports a great variety of geometric representations including an extensive set of traditional CSG primitive implicit solids such as boxes, ellipsoids, cones, and tori, as well as explicit solids made from closed collections of Uniform B-Spline Surfaces, Non-Uniform Rational B-Spline (NURBS) surfaces, n-Manifold Geometry (NMG), and purely faceted mesh geometry. All geometric objects may be

combined using boolean set-theoretic CSG operations including union, intersection, and difference. [167] [168]

Although BRL-CAD has been used for a wide variety of engineering and graphics applications, the package's primary purpose continues to be the support of ballistic and electromagnetic analyses. Accordingly, developers have found CSG modelling to be the best approach in terms of model accuracy, storage efficiency, precision, and speed of computational analysis. [168]

While polygonal and boundary representation (B-rep) modelling often focuses on just the surfaces of objects, CSG modelling focuses on the entire volume and content of objects. This gives BRL-CAD the capability to be "more than skin deep" and build objects with real-world materials, densities, and thicknesses so that physical phenomena such as ballistic penetration and thermal, radiative, neutron, and other types of transport can be studied. [168]

For example in [170], the methodology of the aircraft survivability analysis considering vulnerability of the aircraft against fragmenting warhead threat was studied, and for the shot-line analysis, the functions and required libraries of the BRL-CAD software are integrated in the code which is used as the shot-line subroutine of the main survivability analysis code.

The BRL-CAD libraries are designed primarily for the geometric modeler who also wants to edit software and, perhaps, design custom tools. Each library fits into one of three categories: creating and/or editing geometry, ray-tracing geometry, or image handling. [168]

The application side of BRL-CAD also offers a number of tools and utilities. They primarily concern is geometric conversion, geometric interrogation, image format conversion, and command-line-oriented image manipulation. An overview of libraries, tools, and utilities of BRL-CAD can be found in [168].

4) QCAD

QCAD is a multi-platform, open source application for computer aided drafting (CAD) in two dimensions (2D). Technical drawings such as plans for aircraft models (Fig. 37), buildings, interiors, mechanical parts, or schematics and diagrams can be created with QCAD. QCAD was designed with modularity, extensibility and portability in mind; moreover, QCAD has intuitive user interface. [171]

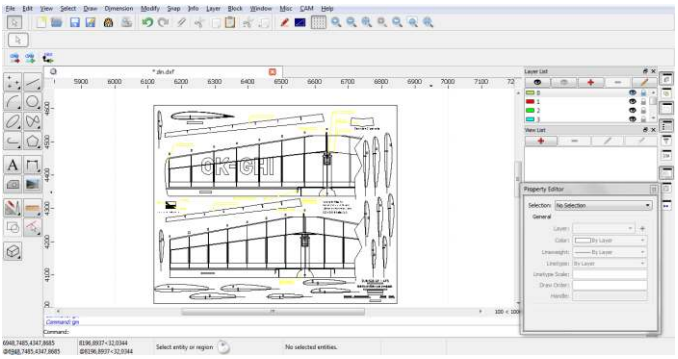

Fig. 37. The wings and airfoil of the ZLIN 526 AF/AFS aircraft model in QCAD

Main Features of QCAD are: [171] [172]

- Layers
- Blocks (grouping)
- 35 CAD fonts included
- Support for TrueType fonts
- Various Metrical and Imperial units
- DXF and DWG import and export
- Import bitmaps into drawing and export drawing as bitmap (BMP, JPEG, PNG, TIFF, ICO, PPM, XBM, XPM)
- Printing to scale
- Printing on multiple pages
- Over 40 construction tools
- Over 20 modification tools
- Construction and modification of points, lines, arcs, circles, ellipses, splines, polylines, texts, dimensions, hatches, fills, raster images
- Various entity selection tools
- Object snaps
- Measuring tools
- QCAD Library Browser with over 5000 CAD parts
- Complete ECMAScript scripting interface

Using the QCAD 3 scripting interface, new interactive tools and user interface components can be added to QCAD without having to set up a development environment or requiring a special developer license. In the same manner, completely new applications can be developed using only the script interface of QCAD. [173]

If an extension of QCAD by using scripts is not possible, a QCAD C++ plugin which wraps developer's library can be created; however, the necessary hooks to access library functionality through the script interface have to be added. [173]

H. OpenVSP

OpenVSP (Vehicle Sketch Pad) is a parametric aircraft geometry tool which allows the user to create a 3D model of an aircraft defined by common engineering parameters. This model can be processed into formats suitable for engineering analysis, for example into STL, MSH, HRM, 3DM, FEL, X3D and other formats. OpenVSP was successfully used, for instance, in [143] as a part of system for the optimization of wing, body, and tail of aircraft. Furthermore, in [180], four types of reduced-fidelity geometric representations were defined in response to a need for bridging the gap between conceptual design and analysis. [174] [177] [178] [179]

The predecessors to OpenVSP have been developed by NASA since the early 1990's. In January 2012, OpenVSP was

released as an open source project under the NOSA 1.3 license; the logo of OpenVSP can be seen in Fig. 38. [174] [175] [178]

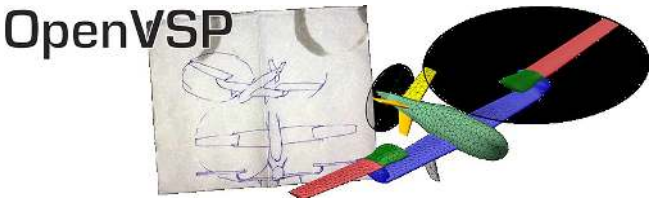

Fig. 38. The Logo of OpenVSP [175]

In [176], there is a VSP Hangar where over 170 aircraft VSP models of various types can be found. Fig. 39 shows Dragon Eye UAV which was downloaded from VSP Hangar and displayed in OpenVSP application.

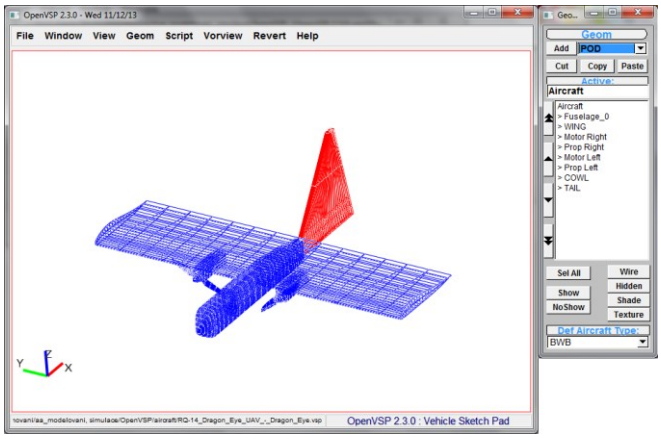

Fig. 39. Dragon Eye UAV displayed in OpenVSP

Traditional CAD tools generate a static solution to a design problem. However, many of the strengths of CAD which make it beneficial in the preliminary and detailed design phases actually become either of little benefit or even hindrances in the conceptual design phase. Instead of manually creating a CAD model by dragging and dropping components, the parametric design is specified using variables and functions. Parametric design defines the relationships between components in a design. As a result, changing a variable which defines part of a model will adapt all the connected components so as to maintain a coherent design. Although there is a longer lead time to implement the initial model, once the model is developed, the user can easily create endless variations of the original. In other words, parametric design systems make the computer a generative design tool and are already used extensively as a rapid prototyping technique in architecture and aeronautics. [143] [177] [178] [179]

OpenVSP builds a text file which is filled out through a series of forms presented graphically to the user, and the geometry is displayed in real-time in a three-dimensional display window. [177] [178] [179]

In order to provide real-time performance, the true wetted surface of the aircraft is not generated or updated with every design change. Instead, aircraft components are modeled and displayed independently but simultaneously; the wing and fuselage are both represented, but the wing-fuselage intersection is not calculated and the components are not trimmed at the intersection. [179]

At any point in the design process, the designer may elect to calculate the component intersections and generate the outer mold lines of the vehicle. This true wetted surface can be used by OpenVSP to perform a number of analysis tasks. The surface representation generated by OpenVSP can be used as the starting point for volume mesh generation to be used in CFD analysis. Moreover, it can be used as input to a rapid prototyping machine or 3D printer. The surface model output by OpenVSP can also be used to create high quality illustrations and renderings of the design concept. [179]

Mesh grid density is defined and controlled by using point, line and box sources. These sources specify the desired edge length near the source. The radius of the source dictates the volume affected by that source. The target length is decreased by the square of the fractional distance away from the source. The source size may be automatically increased to prevent the edge length from changing more than 20% at source borders. The position of the sources is specified in the parameter space of the components. This allows the user to change the geometry without having to redefine sources as can be seen in Fig. 40 and Fig. 41. [179]

Line and box sources require two points to specify location and size. To improve the speed of the mesh density specification, the meshing process is split into two parts: intersect and mesh. If the geometry does not change, only the mesh process is required to see changes in density. When the mesh density is satisfactory, the mesh can be exported, for example, in Nascart (DAT) or STL format. [179]

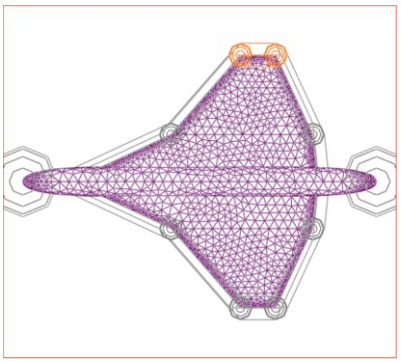

Fig. 40. The original geometry [179]

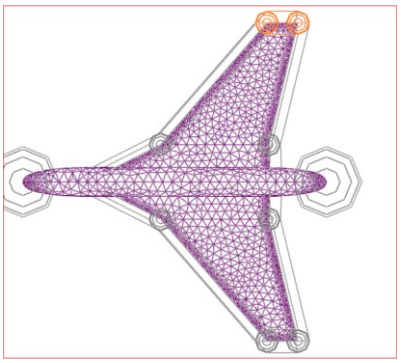

Fig. 41. The altered geometry [179]

The primary mesh control interface presented to the user is depicted in Fig. 42. This interface allows the user to create and modify the grid density control sources. [179]

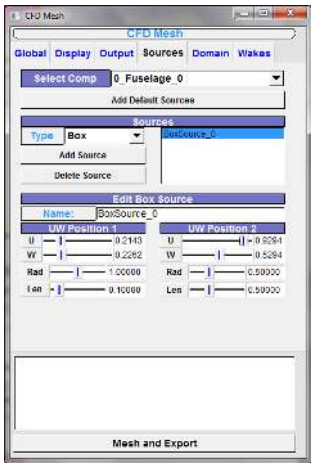

Fig. 42. CFD Mesh control dialog

OpenVSP has the capability to automatically create a default set of mesh control sources for each geometric component in the model. These default sources provide a reasonable foundation for tweaking and customization. In addition to the controls for the individual sources, the user can adjust some global parameters scaling mesh size and limiting the largest triangle in the mesh. These global parameters allow the user to quickly adjust a mesh as a whole, thereby creating a series of related meshes. [179]

An example of how parameterization can be used effectively is in the area of defining high-lift devices. OpenVSP allows for arbitrary airfoil coordinates in relation to the traditional zero to one chord reference. This means that the cruise configuration wing can be defined, and if the high-lift configuration airfoils are defined properly, then multiple copies of the cruise wing only have to differ by the airfoil file chosen in order to correctly size and position all of the elements. Any other high-lift flap settings need only to have different airfoil files read in to define them as well. The accuracy of the positioning is only dependent upon the accuracy of the airfoil file. If for any reason the cruise wing geometry is altered, then, everything that is necessary to be done is updating the copies to the same values. Unfortunately, this update must be performed manually. However, a future feature may be to automate this process by allowing the designer to specify dependencies down to the individual parameter level. Currently, parent-child dependencies only extend to position and rotation. [177]

OpenVSP also has some capability to do arbitrary shapes. In general, this should be avoided because it negates the main advantage which OpenVSP has over CAD, namely parametric input and the ease with which the geometry can be modified. [177]

I. JavaFoil

JavaFoil uses several traditional methods for airfoil analysis. The following two methods build the backbone of the program: [229] [230]

- The potential flow analysis is performed with a higher order panel method. It calculates the local, inviscid flow velocity along the surface of an airfoil for any desired angle of attack by using a set of airfoil coordinates.
- The boundary layer analysis module steps along the upper and the lower surfaces of the airfoil, starting at the stagnation point. It solves a set of differential equations to find the various boundary layer parameters; it is a so called integral method. The equations and criteria for transition and separation are based on the procedures described by Eppler. The boundary layer module works best in the Reynolds number regime between 500,000 and 20,000,000.

A standard compressibility correction according to Karman and Tsien has been implemented to take mild Mach number effects into account. As long as the flow stays subsonic (this usually means that Mach numbers are between zero and 0.5), the results should be fairly accurate. Airfoils in supersonic flow cannot be analyzed with the methods in JavaFoil. [229]

First, it calculates the distribution of the velocity on the airfoil surface which can be integrated to get the lift and the moment coefficient. Then, it calculates the behavior of the flow close to the airfoil surface (the boundary layer). The boundary layer data can be used to calculate the friction drag of the airfoil. Both steps are repeated for the given range of angle of attacks, which yields a complete polar of the airfoil for one fixed Reynolds number. [229] [230]

JavaFoil does not model laminar separation bubbles and flow separation, the results will be incorrect if either of these occur. Flow separation, as it occurs at stall, is modeled to some extent by empirical corrections, so that maximum lift can be predicted for “conventional” airfoils. If an airfoil beyond stall is analyzed, the results will be quite inaccurate. Two dimensional analysis methods should not be used at all in this regime, as the flow field beyond stall is fully three dimensional with spanwise flow and strong vortices. [229] [230]

If laminar separation is detected, the method switches to turbulent flow and continues. When turbulent separation is found, the boundary layer integration is stopped and an empirical drag penalty depending on the length of the separated region is added to the result. [230]

The drag is applied by examining the boundary layer parameters at the trailing edge using the Squire-Young formula. [230]

The following empirical transition criteria have been implemented and can be selected by the user: [230]

- Eppler 1
- Eppler 2
- Michel 1
- Michel 2
- Granville
- Drela’s e^n approximation 1

- Drela's e^n approximation 2
- Arnal e^n approximation by Würz

The JavaFoil program contains a row of tabs on top and a card area below. Each tab shows its associated card which contains input and output elements for a certain topic. The cards are divided in topics like Geometry, Modify, Velocity, Flow Field (see Fig. 43), Boundary Layer, Polars and Options. [231]

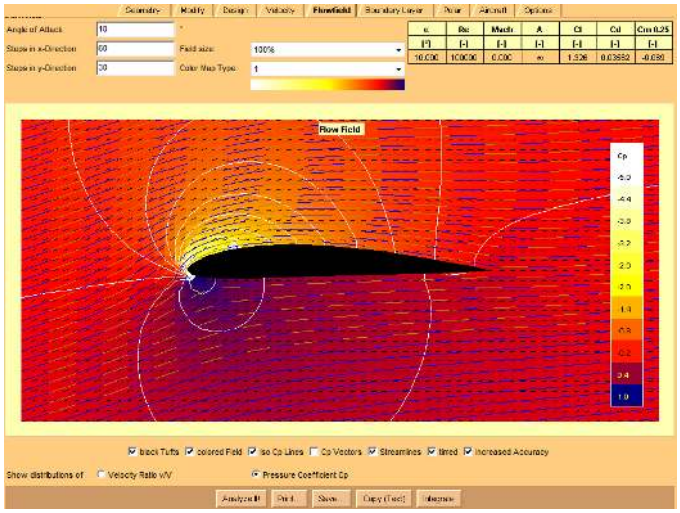

Fig. 43. The flow field card in JavaFoil with an analysis of an airfoil

JavaFoil may create the following standard airfoils: [231]

- 4-digit series (like NACA 2415)
- 5-digit series (like NACA 23015)
- 16-series (like NACA 16-412)
- 6-series (like NACA 64-412)
- TsAGI "B" series airfoils
- NPL "EC", "EH" series airfoils
- Symmetrical Circular Arc airfoils
- Symmetrical Double Wedge airfoils
- Cambered Plate airfoils
- Van de Vooren conformal mapping airfoil
- Newman airfoil
- Helmbold-Keine airfoil

J. JavaProp

JavaProp is based on the blade element theory presented in [234]. The blade is divided into small sections, which are handled independently from each other. Each segment has a chord and a blade angle, and associated airfoil characteristics. The theory makes no provision for three dimensional effects, like sweep angle or cross flow; however it is able to find the additional axial and circumferential velocity added to the incoming flow by each blade segment. This additional velocity results in an acceleration of the flow and thus thrust. Usually this simplified model works very well, when the power and thrust loading of the propeller (power per disk area) is relatively small, as it is the case for most aircraft propellers. [232] [233] [235]

Based on the theory of the optimum propeller (as developed by Betz, Prandtl, Glauert), only a small number of design parameters must be specified. These are: [233]

- The number of blades B
- The axial velocity v of the flow (flight speed or boat speed)
- The diameter D of the propeller
- The selected distributions of airfoil lift and drag coefficients C_L and C_D along the radius
- The desired thrust T or the available shaft power P
- The density ρ of the medium (air: ~1.22 kg/m^3, water: ~1000 kg/m^3)

The design procedure creates the blade geometry in terms of the chord distribution along the radius as well as the distribution of the blade angle. The influence of blade number and tip loss are taken into account by the "Prandtl Tip-Loss Factor". [233] [235]

JavaProp offers two ways to use the analysis procedure: [235]

- Analyze the propeller for its full operating range, from static to the beginning of the wind-milling range.
- Perform an analysis for one advance ratio only which gives a user more details for the aerodynamic conditions along the radius.

The local chord length c depends mainly on the prescribed lift coefficient C_L - if designer would like to have wider blades, he have to choose a smaller design lift coefficient (or angle of attack) and vice versa. It should be noted that the design procedure does not work accurately for high thrust loadings as they occur under static conditions. If nonsense values for the blade chord are received, the power loading of the propeller is probably too high. The power coefficient P_C should be less than 1.5; otherwise the theory is not fully applicable and may lead to errors. [233]

However, the blade element method is limited when flow separation occurs e.g. at static conditions. Moreover, JavaProp comes with a set of airfoil polars which can represent only a limited model of the whole range of possible airfoil sections. [236] [235]

Finally, the flow field around a propeller is complex and fully three dimensional with boundary layers, Mach number effects and local flow separation. This problem may also be difficult to model accurately with the most sophisticated tools such as Navier-Stokes solvers which typically require long time to calculation. On the other hand, JavaProp can give a first answer in fractions of a second for price of the accuracy. However, results may be overestimated. [236]

The JavaFoil program contains a row of tabs on top and a card area below. Each tab shows its associated card which contains input and output elements for a certain topic. The cards are divided in topics like Design, Airfoils, Geometry (see Fig. 44), Analysis and Options. [237]

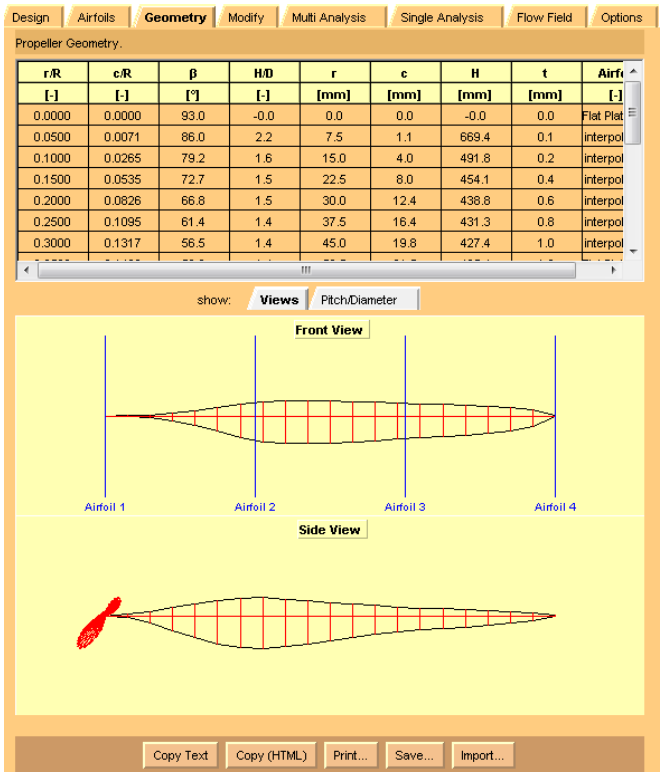

Fig. 44. The geometry card in JavaProp with the geometry of a propeller

A user can work only with a single propeller in JavaProp. It means that he has a single propeller, which can be designed, analyzed, modified, and analyzed again. All manipulations such as the airfoil choice, specification of diameter, or RPM setting, applying modifications to the blade shape, and importing new blade geometry will alter this virtual propeller. The unit system is metric for all entries and results if not noted otherwise. [237]

K. *Other Software*

In this section, other interesting or acceptable software which can be used for modelling, simulation, and development is briefly described or only mentioned.

1) Larosterna

The Larosterna software includes a surface modelling tool and mesh generator (SUMO), and a visualization program (SCOPE). [225]

The surface modeler SUMO is a graphical tool aimed at rapid creation of aircraft geometries and automatic surface mesh generation. The plan and side views of a fuselage in SUMO are shown in Fig. 45 and the meshed aircraft can be seen in Fig. 46. [225] [226]

SUMO is not a CAD system, but rather an easy-to-use parametric sketchpad, highly specialized towards aircraft configurations. SUMO can import IGES, STEP, STL, CGNS, SU2, CEASIOM and other files. After the generation of a mesh, the surface can be exported to CGNS, MSH, STL, SU2, and ZML files. [225] [226]

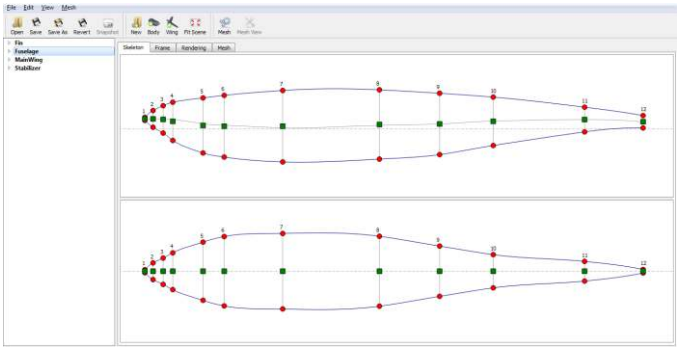

Fig. 45. The plan and side views of a fuselage in SUMO

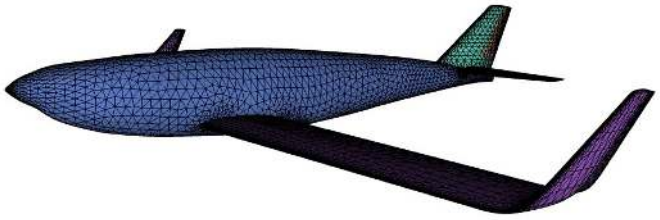

Fig. 46. A meshed aircraft generated with SUMO

SCOPE is a visualization program to display surface data, flutter mode-shapes and flight trajectories. Furthermore, it can read NASTRAN as well as modal analysis results and animate Eigen-mode shapes. SCOPE can also import/export CGNS, SU2, MSH, STL and other files. [225] [227]

2) VAMPzero, CPACS, and TiGL

VAMPzero is an open-source software tool for the conceptual design of aircraft. Based on well-known handbook methods (e.g. Raymer and Roskam), the design of new configurations includes outer geometry as well as structures, engines, systems, mission analysis and costs. It supports working in multi-disciplinary and multi-fidelity environments. VAMPzero can interpret data from CPACS (Common Parametric Aircraft Configuration Schema) and can be used to generate CPACS files. [181] [178] [182] [183]

VAMPzero is based on an object oriented structure and, as a result, is highly flexible. Furthermore, the structure distinguishes feature aspects (file handling, convergence control, and process control) and design aspects (parameter definition, calculation methods) in a way that makes extensions easy to implement. The design aspects are grouped into components, disciplines and parameters, whereas the parameters contain the actual design knowledge. [181] [178] [182] [183]

As VAMPzero is a supportive analysis module for CPACS, it has to handle two tasks: initialization and integration. Obviously, a design process needs to go through requirements definition and conceptual design before preliminary methods can be applied. VAMPzero handles the conceptual design stage

but also initializes the CPACS data set; thus, higher level analysis modules can be triggered. It creates geometries following a knowledge-based engineering approach and writes necessary process data like, for example, tool-specific settings. [183]

The Common Parametric Aircraft Configuration Schema (CPACS) is a data definition for the civil and military aircraft, rotorcraft, jet engines, and entire air transportation systems. CPACS is based on XML technology and enables engineers to exchange information between their tools. As a result, CPACS is a driver for multidisciplinary and multi-fidelity design in distributed environments. CPACS describes the characteristics of aircraft, rotorcraft, engines, climate impact, fleets and mission in a structured, hierarchical manner. Not only product but also process information is stored in CPACS. The process information helps in setting up workflows for analysis modules. Since CPACS follows a central model approach, the number of interfaces is reduced to a minimum. [184] [178] [183]

The TiGL Geometry Library can be used for easy processing of geometric data stored inside CPACS data sets. TiGL offers query functions for the geometry structure. These functions can be used, for example, to detect how many segments are attached to a certain segment, which indices these segments have, or how many wings and fuselages the current airplane configuration contains. This functionality is necessary because not only the modelling of simple wings or fuselages but also the description of quite complicated structures with branches or flaps is targeted. TiGL uses OpenCASCADE to represent the airplane geometry by B-spline surfaces in order to compute surface points and also to export the geometry in the IGES/STEP/STL/VTK format. The library provides external interfaces for C, C++, Python, Java, MATLAB and FORTRAN. The TiGLViewer is an application used to visualize the geometries. [185]

3) TetrUSS

TetrUSS is a time-tested computational aerodynamic capability servicing the configuration aerodynamic needs of NASA's airframe and exploration programs. Present capabilities include rapid grid generation, inviscid and viscous flow analysis and design, special functional boundary conditions, and ease of use. TetrUSS also includes a modular capability for computing aero-elastic effects, iterative design, and interactive boundary layer. Future goals are focused on improving process automation, better integrating functional capabilities, and increasing its impact on new NASA projects and programs. [186] [188]

However, TetrUSS is only available to U.S. entities, citizens, and permanent residents because the software has been developed by the United States government and is subject to US export regulations and NASA policy. As a result, TetrUSS cannot be tested by the author of this paper. [187]

4) Extensions and Tools of Computational Environments

Tornado for Octave (also for Matlab) is a Vortex Lattice Method for linear aerodynamic wing design applications in conceptual aircraft design or in aeronautical education. By modelling all lifting surfaces as thin plates, Tornado can solve for most aerodynamic derivatives for a wide range of aircraft geometries. [22]

Aerospace blockset for Scilab/XCos is an external module providing aerospace palette. It is based on CelestLab aerospace library. Although Scilab/XCos and aerospace blockset are very interesting compensation for Matlab/Simulink, Aerospace blockset is now designed rather for satellites than aircraft. However, there is an example named "Quadrocopter attitude estimation with TRIAD" which demonstrates that UAV simulation is also possible. [23]

OpenFDM is an open source flight dynamics library for Modelica and has basic functions for the modelling of aircraft, aerodynamics, control, navigation, and propulsion. [192]

a) Flapping Flight Simulation Package

Flapping Flight Simulation package can be used to simulate the physics of flapping-wing flight which includes simulating the flight of living organisms, such as birds and insects, and also man-made flapping-wing air vehicles. [196]

This package contains three applications for simulation flapping flight: [193] [196]

- FLAPSIM - An inverse dynamics application which simulates the dynamics of flapping wings, predicting aerodynamic forces and torques, and mechanical power.
- FLAPOPTIMISE - A numerical optimization tool which predicts the most energy efficient wing kinematics.
- WAKESIM - A point vortex simulator which simulates the geometry of wake shed by flapping wings.

Wing dynamics for a hoverfly cruising at 3m/s is shown in Fig. 47 [194]. Detailed information about Flapping Flight Simulation package can be found in [195] and [196].

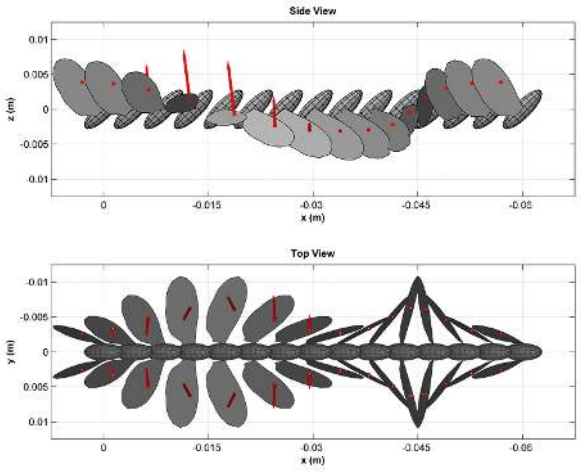

Fig. 47. Wing dynamics for a hoverfly cruising at 3m/s [194]

Applications in the package require the Matlab Compiler Runtime (MCR) to be installed. MCR is a standalone set of shared libraries which enables the execution of compiled

MATLAB applications. Installing MCR does not install a complete version of Matlab; as a result, no Matlab license is required. [193]

b) CEASIOM

CEASIOM is a free conceptual aircraft design tool and involves stability and control driven sizing and optimization in the design cycle earlier than is standard practice nowadays. CEASIOM is the result of the EU funded project SimSAC (SIMulating Aircraft Stability and Control) which had as objective to build an integrated simulation environment for computing stability and control information with quantifiable uncertainty. [189] [178]

CEASIOM integrates into one application the main design disciplines, aerodynamics, structures, and flight dynamics, impacting on the aircraft's performance. In other words, CEASIOM is a tri-disciplinary analysis which should participate in the design of the aero-servo-elastic aircraft. [189] [178]

Significant features developed and integrated in CEASIOM as modules are: (see also Fig. 48) [190] [178] [183]

- Geometry module / Aircraft Builder (ACBuilder) - A customized geometry construction system coupled to surface and volume grid generators; port to CAD via IGES.
- Aerodynamic Module (AMB-CFD) - A replacement of current handbook aerodynamic methods (Digital Datcom) with new adaptable-fidelity modules: Steady and unsteady TORNADO vortex-lattice code for low-speed aerodynamics and aero-elasticity, Panel method, Inviscid Euler solver EDGE for high-speed aerodynamics, and RANS flow simulator for high-fidelity analysis of extreme flight conditions.
- Stability and Control module (SDSA) - A static and dynamic stability and control analyzer, and flying-quality assessor. Test flights with 6DOF flight simulation, and performance prediction, also includes human pilot model, Stability Augmentation System (SAS) and a LQR-based (Linear-Quadratic Regulator) flight control system package are among the major functionalities of this module. The standalone version of SDSA is described in the chapter V.K.8).
- Aero-elastic module (NeoCASS) - Quasi-analytical structural analysis methods which support aero-elastic problem formulation and solution.
- Flight Control System Design Toolkit (FCSDT) - A designer toolkit for flight control-law formulation, simulation and technical decision support, permitting flight control system design philosophy and architecture to be coupled early in the conceptual design phase.

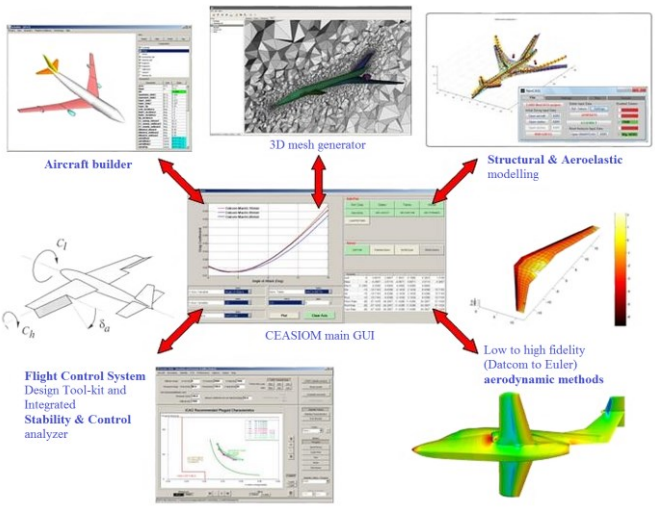

Fig. 48. The core modules in CEASIOM [189]

CEASIOM can be linked to VAMPzero via CPACS (briefly described in the chapter V.K.2)). The main reason why modules must be interfaced to a code like VAMPzero is that CEASIOM does not perform the initial sizing of a baseline configuration. [178] [183]

Despite CEASIOM is freeware, to use it, Matlab is need to be installed (Release 2008a or later), including Simulink for use of FCSDT. As a result, CEASIOM can be used without fee when an institution has already owned a Matlab license. [191] [189]

5) Calculators

FlapDesign (Fig. 49) is a simple, free software program which runs within a web browser with installed Java plug-in. It can find the correct dimensions of an ornithopter wing-flapping mechanism. [197]

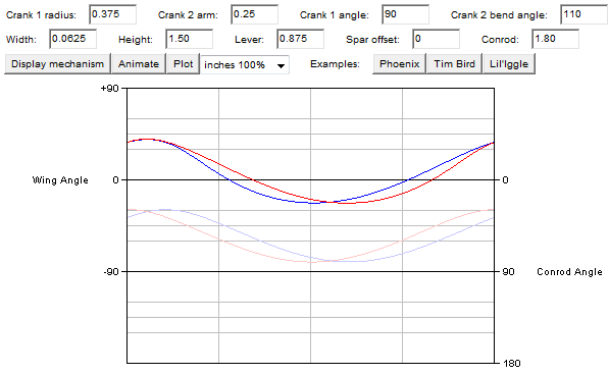

Fig. 49. A graph of the wing angle as the crank rotates in FlapDesign [197]

Orni calculation tools are based on the calculation method specified for Ornithopters and can determine the power and the twisting of the wing at stationary flight situations. Furthermore, the calculation method permits at least an approximate quantitative specification of dynamics and aerodynamics of profiled flapping wings. Primarily, the numerical comparison of various factors influencing of flapping wings. [198]

Profiled flapping wings and quasi-stationary flow conditions are presumed; thus, calculations lead to useful results only for a fast forward flight with a relatively low flapping frequency (large birds, flying with lift). [198]

The mathematical models are executable with the relevant software applications. The Orni calculation tools contains: [198]

- Orni 1 (XLS or XLSX file) is a simple calculation tool to describe the wing twisting.
- Orni 2 (MathCAD file) is a calculation method for ornithopter models. For the construction of a flapping wing model, the calculation method describes progress of various force and moment at the wing. Furthermore, the required power and, especially, the wing twisting along the span are being calculated.
- Orni 3 (MathCAD file) is mathematical model for ornithopters which is based on the calculation tool Orni 2. Gliding and power flights of a flapping wing model can be analyzed singly and in series.
- Comparison (XLSX file) - the flight performance of a mute swan is compared with the performances of a propeller and a flapping wing model.

However, Orni 2 and Orni 3 cannot be imported into MathCAD Express (free for use) because these files were developed with commercial MathCAD Professional 13 and 14, and furthermore, they are in XMCDZ format.

WebOcalc (Fig. 50) was created to make the easy selection of motors, propellers, gearbox ratios, and batteries for electric-powered model airplanes. Moreover, WebOCalc includes several app wizards to recommendation of battery chemistry, cell capacity and pack voltage, current draw, propeller size, motor Kv, etc. This is a fast way how to get a model in the air. However, the experienced user still has complete flexibility to insert any values of his choice. [200]

Traditional motor calculator programs ignore airframe characteristics, and can result in poor-flying systems. On the other hand, WebOCalc matches airframe and power system characteristics every time. This may ensure great performance for model aircraft. [200]

Fig. 50. WebOcalc 1.7.6 with estimated data

PowerCalc is motor/propeller simulation software which can help a user to effectively choose a motor and propeller for his electric powered model airplane. It may be used along with WebOCalc to quickly focus on the power system. [201]

The traditional mathematical model for motor simulation uses three motor constants, K_v, R_m, and I_0. This three-constant model works well, but has significant inaccuracies. In contrast, PowerCalc is based on a four-constant mathematical model which should predict motor performance better than the traditional three-constant motor model does. [201]

Web site [202] contains the series of calculators, and Aerodynamics, Beginners' Guide. The links to calculators (or calculators themselves) are often placed inside the text of the Guide. There are at least these calculators:

- Aircraft Center of Gravity Calculator [203]
- Canard Center of Gravity Calculator [204]
- Wing loading Calculator [205]
- Stall Speed Calculator [206]
- Level Flight Speed Calculator [207]
- Motor Efficiency Calculator [208]
- Propeller's Static Thrust Estimation [209]
- Electric Motor & Prop Combination Estimation [210]
- Power/Weight Performance Estimation [211]

eCalc is a set of four calculators as can be seen in Fig. 51. eCalc provides web-based services to calculate, evaluate, and design electric motor driven systems for remote controlled (RC) models. However, despite eCalc can be useful, only its restricted demo version with reduced functionality and 25% random database is free for use. [199]

Fig. 51. The set of four eCalc calculators [199]

6) Apame

Apame is a 3D Panel Method program used for calculating aerodynamic forces and moments acting on an aircraft in flight as can be seen in Fig. 52. [220]

Apame can replace CFD programs (like Fluent, OpenFOAM, etc.) for subsonic attached flows where calculation time is important and friction drag can be ignored (optimization problems, conceptual designs, aerodynamic load generation, etc.). The calculation time is much shorter compared to classic CFD; seconds vs. hours. [220] [221]

Project can perform two operations; evaluation and optimization: [220] [221]

- Evaluation: for this purpose, ApameGUI is used to import already available meshes (e.g. meshes in

NASTRAN and FLUENT format), pre-process it, send it to the ApameSolver and evaluate results.

- Optimization: in this case, user-defined scripts are used to parametrically generate mesh and send it to the ApameSolver in a single optimization step. For this purpose, ApameScripts are given inside Apame package as base examples.

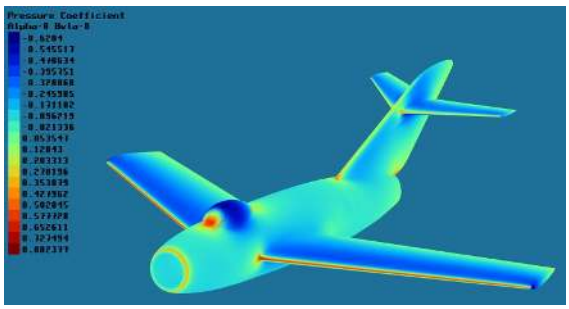

Fig. 52. An analysis of an aircraft by using APAME [220]

7) PANUKL 2012

PANUKL is a package to compute the aerodynamic characteristics of an aircraft using low order panel method, where the Dirichlet problem is solved and the quadrangle panels are used. The flat vortex wake, parallel to the free stream velocity or parallel to chord is assumed. Moreover, it contains functions for pre- and post-processing. [238] [239] [240]

Current version contains new editor and mesh generator; however, the old version of mesh generator is still embedded. New geometry definition methods are available only in the new mesh generator. Current version contains also option to export the geometry and the results (pressure distribution) to FEM analysis (Calculix). [238]

Computational method strongly depends on the way of aircraft body modelling; generally, there are two methods in which the body of an aircraft is modeled using: [239]

- Thin surfaces
- A three dimensional model

PANUKL 2012 application is composed of three main subprogram groups: [239]

- The data preparation programs
- The programs to process data and make computations
- The managing program for showing the obtained results and make appropriate changes and modifications.

PANUKL can be used to create a grid (made from quadrangle panels as can be seen in Fig. 53) which describes an aircraft body. To create the grid file (.INP), the following input files have to be prepared: [239]

- Main aircraft geometry description file (.MS2) - contains aircraft reference data, information about wing, tail, fuselage overall geometry
- Wing airfoil geometry file (.PRF, .DAT, .KOO)
- Fuselage geometry file (.F)

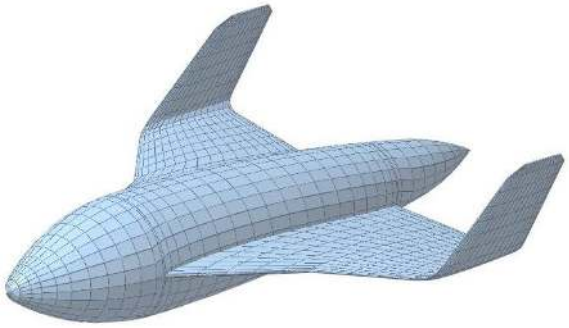

Fig. 53. A model displayed in the Grid Viewer of PANUKL [239]

8) Simulation and Dynamic Stability Analysis (SDSA)

SDSA module was developed as the CEASIOM (described in the chapter V.K.4)b)) module; however it can be run as a standalone application as well. SDSA was developed for stability and control analysis and is able to compute stability characteristics using linear and nonlinear simulation model as well. [240] [241]

SDSA uses the same 6-DOF mathematical nonlinear model of the aircraft motion for all functions. For the eigenvalue analysis, the model is linearized numerically by computing the Jacobian matrix of state derivatives around the equilibrium (trim) point. Eigenvalues and eigenvectors analysis allow automatic recognition of the typical modes of motion and their parameters. The flight simulation module can be used to perform test flights and record flight parameters in real-time. The recorded data can be used for identification of the typical modes of motions and their parameters (period, damping coefficient, phase shift). The stability analysis results can be assessed on basis of CS/FAR, ICAO, and MIL requirements. [240] [241]

Necessary data (aerodynamics, mass, inertia) can be imported to SDSA as an XML file or as a set of plain text files. The second option is useful e.g. for experimental data. The data set contains aerodynamic coefficients or/and stability derivatives tables, mass and inertia data, propulsion data, control derivatives and reference dimensions. The control and propulsion data can be completed and edited using special options of SDSA. SDSA accepts aerodynamic data as tables of stability derivatives as function of angle of attack and Mach number. SDSA also accepts a multidimensional array of force and moment coefficients versus six state parameters (angle of attack, Mach number, sideslip angle and rotational velocity components). A similar array is defined for control derivatives and stability derivatives versus selected accelerations (i.e. alpha dot derivatives). All aerodynamic data (derivatives) can be reviewed and are checked by comparison with typical values. [240] [241]

SDSA may run in the batch mode and can send necessary output data for an optimization procedure without any prompting; as a result, the optimization process can run completely in an automatic way; however, SDSA needs an

external application for the optimization process because it is not included. [240]

9) XFLR5

XFLR5 is an analysis tool for airfoils, wings and planes operating at low Reynolds Numbers and includes: [212] [213] [214]

- XFoil's Direct and Inverse analysis capabilities
- Wing design and analysis capabilities based on the Lifting Line Theory, on the Vortex Lattice Method, and on a 3D Panel Method

The code has been intended and written exclusively for the design of model sailplanes, for which it gives reasonable and consistent results. One analysis of a sailplane can be seen in Fig. 54. [214] [213]

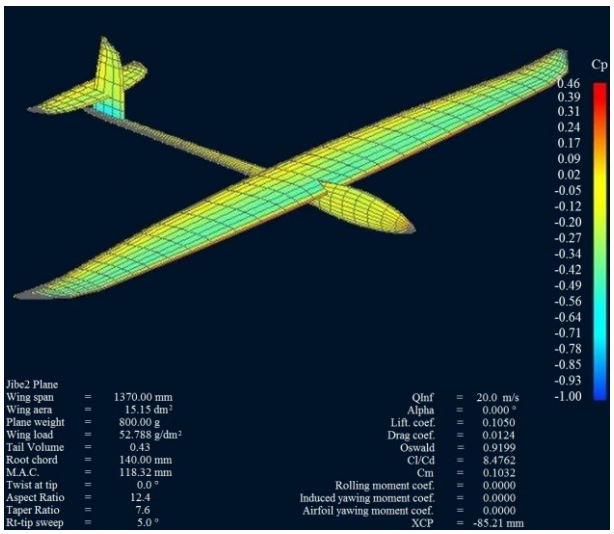

Fig. 54. An analysis of a sailplane by using XFLR5 [213]

10) JBLADE

JBLADE is an open-source propeller design and analysis code based on QBLADE and XFLR5. The airfoil performance figures needed for the blades simulation come from QBLADE's coupling with XFOIL. This integration allows the fast design of custom airfoils and computation of their polars. [215] [216]

JBLADE uses the classical Blade Element Momentum (BEM) theory modified to account for the 3D flow equilibrium. The code can estimate the performance curves of a given propeller design for off-design analysis. The software has a graphical interface making easier to build and analyze the propeller simulations. [215] [216]

JBLADE Capabilities are, for example, Extrapolation of XFOIL generated or imported polar data to 360° Angle of Attack with improved airfoil leading edge radius correlation, Blade design and optimization including 3D visualization, parametric simulations including evaluation of performance over an airspeed, rotational speed, and pitch range, analysis and prediction of performance curves for a given blade geometry, manual selection of BEM correction algorithms, manual selection of all simulation parameters, data browsing and visualization as post processing, export functionality for all created simulation data, blade geometry export functionality, and storing of projects, blades, propellers and simulations in a runtime database. [215] [216]

11) NACA Airfoil Generation – FoilGen and LADSON

FoilGen is an interactive FORTRAN program which allows the user to construct airfoils using the NACA 4 digit or modified 4 digit airfoil thickness distributions and the NACA 4 digit, 5 digit or 6- and 6A series camber lines. A variety of output options are available on the screen. It can also create a file for use as input to airfoil analysis programs. [217]

LADSON is a FORTRAN program which allows the user to approximately obtain the NACA 6 digit or 6A digit airfoils. The thickness distribution of these airfoils is not described by a single equation. [217]

12) Gmsh

Gmsh is a 3D finite element grid/mesh generator with a build-in CAD engine and post-processor. Its design goal is to provide a fast, light and user-friendly meshing tool with parametric input and advanced visualization capabilities. Gmsh is built around four modules: geometry, mesh, solver and post-processing. The specification of any input to these modules is done either interactively using the graphical user interface or in ASCII text files using Gmsh's own scripting language. [218]

Gmsh may import MSH, STL, STEP, IGES, BREP, and other files. Export to MSH, STL, MED, UNV, SU2, WRL and others are possible. It can also be used in collaboration with, for example, Code_Saturne, HiFiLES, PyFR, and enGrid (described in the chapter V.K.13)). [224] [154] [156] [158]

13) enGrid

enGrid is an open-source mesh generation software which is used predominantly for CFD applications. enGrid uses an in-house development for surface meshing and prismatic boundary layers. Internally, enGrid uses the VTK data structures as well as the VTU file format. Currently, enGrid has interfaces to Gmsh, STL, and few other file formats. [224]

enGrid provides native export to OpenFOAM and SU^2. This includes export capabilities for complete OpenFOAM cases (including boundary conditions), as well as support for polyhedral cells. [224]

14) Mission Planner

Mission Planner has more features than its name indicates: it can interface with a PC flight simulator (FlightGear, JSBSim, Xplane, and AeroSim RC) to create a full hardware-in-the-loop (HIL) UAV simulator. Furthermore, Point-and-click waypoint entry using Google Maps/Bing/Open street maps/Custom WMS may be created. [219]

VI. Results and Discussion

As can be seen in previous chapters, many applications can be used for design, analysis, modelling, and simulation of a UAV. However, all of them may not be used in one UAV project because they are, for example, focused on different types of UAVs (fixed-wing, rotary-wing, flapping-wing, lighter-than-air, etc.), or have better alternatives.

Fig. 55 shows all software which can create, or supports the items required for the efficient modelling and simulation of a UAV as described in the chapter IV (i.e. a UAV mathematical model, a control system, FDM, etc.)

The extended version of the categories in Fig. 55 may be listed as follows (the applications may not be limited only to this classification):

- Conceptual/Preliminary Design
  - OpenVSP (V.H)
  - Larosterna (V.K.1))
  - VAMPzero (V.K.2))
  - CEASIOM (V.K.4)b))
- CAD
  - FreeCAD (V.G.1))
  - SALOME (V.G.2))
  - BRL-CAD (V.G.3))
  - QCAD (V.G.4))
- Aerodynamics and Mechanics
  - Airfoil
    - JavaFoil (V.I)
    - XFOIL (V.D.2))
    - The Eppler Airfoil Code (PROFILE - V.E.1))
    - Tornado (V.K.4))
    - Minimum Drag Camber Surface by Vortex Lattice (VLMD - V.E.2))
    - Induced Drag from Span Load Distribution (INDUCED - V.E.3))
    - Flutter Analysis by Strip Theory (FLUTTER - V.E.4))
    - Mean Aerodynamic Chord of a Wing (GETMAC - V.E.5))
    - NACA Airfoil Coordinates (NACA456 - V.E.6))
    - AirfoilTools.com (V.E.6)a))
    - NACA Airfoil Generation (FoilGen and LADSON - V.K.11))
  - Complete Aerodynamics
    - CFD
      - Stanford University Unstructured (SU^2 - V.F.1))
      - OpenFOAM (V.F.2))
      - Code_Saturne (V.F.3))
      - High Fidelity Large Eddy Simulation (HiFiLES - V.F.4))
      - PyFR (V.F.5))
    - Athena Vortex Lattice (AVL - V.D.1))
    - Apame (V.K.6))
    - PANUKL 2012 (V.K.7)) & SDSA (V.K.8))
    - Digital Datcom (V.E.9)), Datcom R2 and OpenDatcom (V.E.9)a)), or Datcom+ (V.E.9)b))
    - Predicting Subsonic or Supersonic Linear Potential Flows about Arbitrary Configurations Using a Higher Order Panel Method (PANAIR - V.E.8))
    - XFLR5 (V.K.7))
    - CEASIOM (V.K.4)b))
    - Aeroelastic Analysis for Rotorcraft in Flight or in a Wind Tunnel (ROTOR, work-in-progress - V.E.10))
  - Propulsion
    - JavaProp (V.J)
    - QPROP/QMIL (V.D.3))
    - CROTOR/XROTOR (V.D.4)a) / V.D.4))
    - Ducted Fan Design Code (DFDC / DFDC v070-ES - V.D.5) / V.D.5)a))
    - JBLADE (V.K.10))
  - Mass Analyzer
    - Mass Properties of a Rigid Structure (MASSPROP - V.E.7))
  - Calculators (V.K.5))
    - Aerodynamics
      - Aircraft Center of Gravity
      - Canard Center of Gravity
      - Wing loading
      - Stall Speed
    - Propulsion
      - WebOcalc
      - PowerCalc
      - Motor Efficiency
      - Propeller's Static Thrust
      - Electric Motor & Prop Combination
      - Level Flight Speed
      - Power/Weight Performance
      - eCalc
    - Flapping-Wing Calculators

        - FlapDesign
        - Orni
- Flight Dynamics Model (FDM)
    - JSBSim (V.A)
    - YASim (V.B)
    - UIUC (V.B)
    - Aerospace blockset for Scilab/XCos (V.K.4))
    - SDSA (V.K.8))
    - CEASIOM (V.K.4)b))
- Simulation Software (Application/Framework)
    - FlightGear (V.B)
    - OpenEaagles (V.C)
    - Simulation and Dynamic Stability Analysis (SDSA - V.K.8))
    - Flapping Flight Simulation Package (V.K.4)a))
    - Dynamic Soaring simulation and optimization program (DSOPT - V.D.6))
    - CEASIOM (V.K.4)b))
    - Mission Planner (HIL Simulation - V.K.14))
- Propulsion and Control System
    - JSBSim (V.A)
    - YASim (V.B)
    - FlightGear (V.B)
    - OpenEaagles (V.C)
    - Digital Datcom (V.E.9))
    - SDSA (V.K.8))
    - CEASIOM (V.K.4)b))
- View
    - 3-VIEW and SILHOUETTE (V.E.12))
    - The TiGL Geometry Library and TiGLViewer (V.K.2))
    - OpenVSP (V.H)
    - CAD applications (V.G)
    - Many other 3D/2D Viewers.
- Geometry Generators or Converters
    - Wireframe generator (MAKEWGS - V.E.11))
    - PANAIR input pre-processor (PANIN - V.E.8)a))
    - Aeromatic (V.A.1))
    - Gmsh (V.K.12))
    - enGrid (V.K.13))
    - OpenVSP (V.H)
    - Larosterna (V.K.1))
    - FreeCAD (V.G.1))
    - CEASIOM (V.K.4)b))
    - Geometry Conversion to LaWGS (2WGS - V.E.13))
    - Geometry Conversion to VRML World (VRML - V.E.14))
    - The Common Parametric Aircraft Configuration Schema (CPACS - V.K.2))
    - Other generators or converters; for example, in the CAD applications.

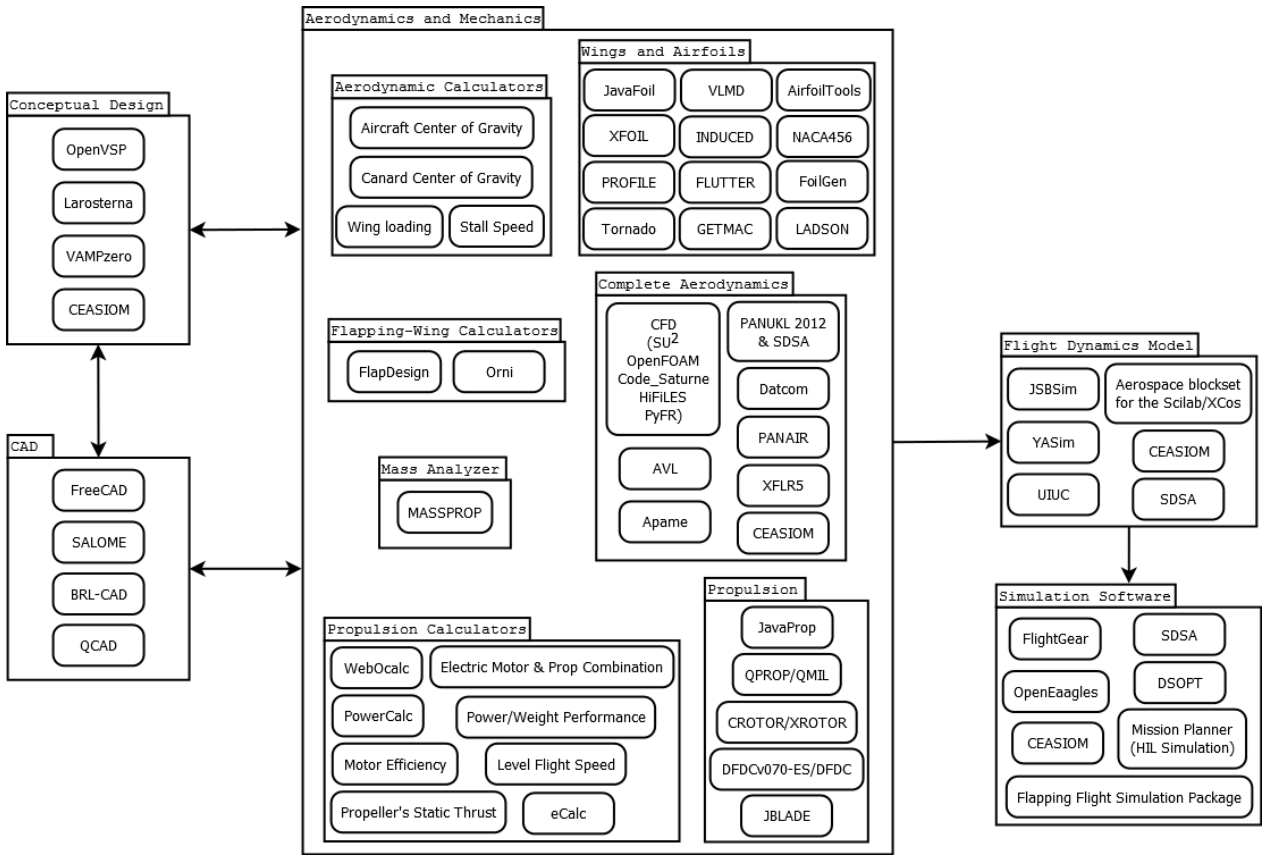

Fig. 55. The block diagram of the free software for the design, analysis, modelling, and simulation of UAVs

If we consider the structure of the UAV system as illustrated in the chapter III, software for airframe, wings, and airfoil creation and for aerodynamics analysis represent a majority in Fig. 55. This situation is not surprise because the aerodynamics of a UAV is probably always the most important section due to its maneuverability, stability, and energy efficiency; thus, the modelling of any new UAV should always start with this part.

Propulsion and control systems are the second most often investigated area of UAVs and are predominantly solved in FDMs, simulators, and calculators. However, detailed aerodynamics and efficiency of propellers, rotors, or ducted fans should be analyzed by using separate applications such as JavaProp, QPROP/QMIL, CROTOR/XROTOR, DFDC, and JBLADE.

Energy storage systems are mostly included in FDMs, simulators, and calculators. However, the models of energy storage should be innovated, e.g. the electric storage in JSBSim should simulate its charge and discharge process, and the new energy source models such as solar cells or chemical fuels may be added.

Transmission and sensors have not been sufficiently solved in the free software. Some principles of transmission and sensors can be found inside OpenEaagles and FlightGear source codes; however, more development is necessary in this area. The best candidates for these innovations are, of course, OpenEaagles and FlightGear; nevertheless also JSBSim and CEASIOM may be appropriate.

A. Evaluation, Compatibility and Conversion

The software in Fig. 55 can be separated into the aerodynamic and mechanical part, and the modelling and simulation part. The diagram also shows the ways how information can be exchanged between the software. In fact, there could also be illustrated the potential feedback from simulation to aerodynamic or mechanical part. However, when the modelling and simulation part starts, the aerodynamic and mechanical part should be already solved. It is obvious, that when all the aerodynamic and mechanical properties of a UAV are known, the aerodynamic and mechanical part does not have to be performed.

All applications have an indirect connection between themselves. It means that their results may affect inputs to other programs; however, these dependencies have to be recorded and applied manually by user. On the other hand, some applications also contain a direct connection which means that an exchange file format exists. It applies especially for 3D geometric and mesh formats.

If we start with the conceptual design, OpenVSP seems to be the best option because the parametric modelling provides an easy way for the building of an initial UAV model. Moreover, OpenVSP may import and export various types of formats, such as STL, MSH, 3DM, HRM, FEL, X3D,

NASTRAN, and other formats which simplify the use in other software.

Larosterna cleverly combines the 2D and 3D visual parametric designs which make it the great second option for conceptual design. This approach gives designers the possibility to perform easy and precise changes. Larosterna can import/export various types of the exchange files; as a result, the connection from/to OpenVSP, CEASIOM, FreeCAD, SU^2, OpenFOAM, Code_Saturne, and other applications may be found.

Another approach is to use CEASIOM which can be linked to VAMPzero via CPACS. The reason for this connection has been described in the chapter V.K.4)b). CEASIOM can be used in the wide range of the design areas, e.g. conceptual design, aerodynamic analysis, flight control system simulation, etc.; thus, the problems with conversion are not important unless a more accurate or other method for analysis is needed. However, the dependence on Matlab and maybe also the closed source code are the main disadvantages of CEASIOM. It is obvious that when the institution owns a Matlab license, the first disadvantage becomes irrelevant; and consequently user can use this comprehensive software for conceptual and preliminary design without major problems.

CAD applications can also be used for conceptual design; for example, FreeCAD was presented as appropriate software for this in [131]. Moreover, FreeCAD may convert many file formats, such as STL, VRML, DXF, NASTRAN, DAE, SVG and other formats; consequently, a 3D model can be imported to many other programs, including OpenVSP which may be advantageous to easy mesh generation.

SALOME has almost same advantages as FreeCAD; it can import or export IGES, STEP, BREP, MED, UNV, DAT, STL files. Consequently, exported files can be imported to CFD applications such as Code_Saturne or SU^2. In fact, the process of the conversion to SU^2 is not direct; SALOME has to export the file to enGrid, Larosterna, or Gmsh (Gmsh is not recommended for the SU2 file in 3D), which may export the file to SU^2. Nevertheless, a script was written to perform the direct conversion from SALOME to SU^2 in [228]. Since the script may be incomplete, the conversion via mesh convertors seems to be better. The same way for the conversion to SU^2 can be used also in other applications, such as OpenVSP and FreeCAD. The mesh of the SU2 format may be displayed probably only in Larosterna.

The MSH format (and other formats) may be used to import files to CFD applications (OpenFOAM, Code_Saturne, HiFiLES, and PyFR), where an aircraft can be analyzed by using quality methods and algorithms. Some other applications for aerodynamic analysis might also import different types of formats, for example Apame may import NASTRAN format. Another interesting choice of the interconnection of the applications can be seen in the CAELinux distribution which is based on open-source CAD/CAM, CFD, FEA, and CAE software such as Freecad, LibreCAD, PyCAM, Cura, Salome, Code_Saturne, OpenFOAM, Code_Aster, and ParaView. [242] [243]

The collection of PDAS (V.E) has mostly possibility to interchange UAV's information via LaWGS file format. If not, the internal formats of some applications can be converted via 2WGS to the LaWGS format. Moreover, LaWGS may be converted to VRML 1.0 via VRML World program; thus it can be imported to FlightGear or FreeCAD.

An example of the indirect connection can be seen in calculators. The results from calculators may be used as an input to OpenVSP, Larosterna, FreeCAD, JSBSim models, and especially to propulsion applications. The calculators mostly solve the problems in the area of engines, propellers, power, batteries, and aerodynamics.

The section for the wing and airfoil analysis can be used nearly separately from other parts of design. Nevertheless, results may be applied to conceptual and preliminary design; e.g. OpenVSP can read an airfoil file and thus change the airfoil of a designed UAV.

The software for the analysis of propellers, rotors, and fans can be used almost separately from other parts of design. However, when we want to use, for example, ducted fan, we must take into account its placement, and thus, suitably change the physical design of the UAV. In addition, data from these applications may be manually inserted to, for instance, JSBSim.

Since JSBSim is based predominantly on aerodynamic and control derivatives, the conversion to the JSBSim model should be simple. The elemental structure of the JSBSim model may be created with Aeromatic. Then, the particular generated values should be changed to correct values. It is always important to check the Aeromatic results because they are questionable and may not always meet the specific requirements for the UAV (as described in the chapter V.A.1)).

However, it is self-evident that a direct conversion to the JSBSim model would be better. The direct conversion is implemented in Datcom+ and Datcom2JSBSim applications; however, it is necessary to define a new airframe specification of the UAV in Datcom format. Moreover, the output JSBSim model depends on Datcom results only.

Because JSBSim is the default FDM in FlightGear and OpenEaagles, there is no problem with the conversion. However, the creation of other important files, such as the 3D model or electrical file, may be necessary. Nevertheless, FlightGear can import 3D formats, such as VRML1 and DXF, thus a simple exchange from, for example, FreeCAD or applications of PDAS should be possible.

If we compare FlightGear and OpenEaagles: OpenEaagles may be better used as a battle manager or for the development of a multi-agent cooperation and complex cybernetic system of the UAV; in contrast, FlightGear is more appropriate for the visualization of a UAV during flight and the examination of UAV flight control system. Moreover, OpenEaagles is a framework with the possibility to simulate tanks, boats, lifeforms, etc. On the other hand, FlightGear is the complete application focused mainly (but not only) on aircraft and may use more FDMs to simulation.

All the flight simulators contain FDM; however, only some of them may be easily separated (at least partially) from the rest of the program. This is the reason why these two categories have been divided in the diagram (Fig. 55).

Because the development seems to be cyclical process, the results from simulation systems may be the reason for modification of the aerodynamic design of the UAV which is being developed. However, after the successful design of the UAV, the production drawings can be created with CAD tools, such as FreeCAD, to help with the physical realization of the UAV. In addition, the 3D model may also be usable for the physical realization; e.g. because of the 3D printing of a prototype or due to a more intelligible display.

B. Possible Future Improvements

The aerodynamic and mechanical part is the most variable section in the free software. There are many of these applications which are being developed, and as a result, new programs in this field may signify the wastage of resources. On the other hand, the support and extension of these already developing applications should be the proper way.

The main lack seems to be the exchange formats of the free software system. Although the direct conversion paths between applications exist, some of them may not be optimal. As a result, the development of new import/export formats might be the key to creation of the complex system with the variable possibilities of use. For example, a direct conversion between internal parametric formats of OpenVSP and Larosterna together with a direct conversion of aerodynamic characteristic to JSBSim model may be usable. Moreover, import/export formats can reduce the time of the development of a UAV in future.

In CAD or conceptual-design programs, the automatic generation of production drawings from 3D model with all necessary views, spot heights and UAV proportions might simplify the process of the following manufacture. FreeCAD seems to be the best option for this purpose because it contains the module to create the 2D production drawings which can be easily extended.

In modelling and simulation part, there are more things which need additional programming for efficiently simulation of UAVs. For instance, in JSBSim and FlightGear, the situation about electric engines, electric sources, and electronic circuits should be improved; for example, these applications might simulate the engine temperature, accumulators, solar cells, fuel cells, methanol and hydrogen energy sources, differences between AC and DC engines, etc.

A LQR- or Artificial Intelligence-based control system should be implemented into flight control system of JSBSim, FlightGear, or OpenEaagles. In addition, an optimal control setting could be calculated with solving the LQG (Linear-Quadratic-Gaussian) problem or with using an evolution algorithm. However, LQR-based flight control system is included in SDSA application; thus, this type of control can also be simulated.

Furthermore, tests of new algorithms intended for intelligent UAVs which should detect living beings (target tracking), possible collisions, and perform reconnaissance might be implemented into FlightGear and OpenEaagles. There is also the basis to program the physical principles of various radio communications which can be applied to the investigation of radio communication failure and to the testing of the UAV behavior when the failure occurs. The simulation of an interface for the command adjustment from control ground station to the UAV may be another interesting improvement.

OpenEaagles and FlightGear are probably the best candidates for the simulation of the sensors and cameras. In this case, appropriate sources (e.g. chemical substances) for sensor activation should also be programmed. This functionality might be added by using an external library. However, the integration of OpenEaagles framework into FlightGear seems to be an interesting option; these two applications may be used, for example, for the investigation of multi-agent cooperation.

Other software, such as Orni tools might be a suitable basis for the flight dynamics model of ornithopters. The theory described in Flapping Flight Simulation Package may also be usable. However, another computational environment or programming language, than that which was used for Orni, (for instance, C++, Python, Scilab/Xcos, Octave, etc.) should be used for Orni Flight Dynamics Model because the original computational environment is not appropriate for FDM development for free use.

The rewriting of ROTOR program to a more convenient programming language (such as C++ or Python) would enable to create the extensive options for the modelling of a rotorcraft. Moreover, a rotorcraft-based FDM may be developed on basis of ROTOR.

VII. CONCLUSION

This paper has described the most interesting free software for the design, analysis, modelling, and simulation of a UAV. Although the selection of the free software has been focused on small (mini) subsonic UAVs, the software can be used for other categories of aircraft in some cases; e.g. for MAVs, large gliders, transonic airplanes, etc. The fundamentals of airplane flight mechanics and aerodynamics, the general structure of a UAV, and the basics of the modelling and simulation of a UAV have also been presented.

The design, analysis, modelling and simulation are probably the first steps in development of a UAV. This approach is advantageous because a computer model allows better repeatability in testing. Consequently, it reduces the probability that the UAV and especially the autopilot will be designed and implemented incorrectly which could result in the UAV crash in the real world; and every crash can increase the distrust of UAVs and of their commercial using, especially in cities.

It has been explained that FDM and Flight Simulators are used in the development process of a UAV for the testing of its design and control systems. The combination of JSBSim Flight Dynamic Model and FlightGear Flight Simulator provide an excellent base for building the simulation environment.

However, because the aerodynamic coefficients and other parameters are not generally provided by FDM, a large number of the programs for the UAV aerodynamic and mechanical analysis have been described in detail in this paper. Despite the many connections between applications have been presented in the chapter VI.A, there may be found other possible connections.

The CFD programs (such as SU^2, OpenFOAM, Code_Saturne, HiFiLES, and PyFR) have the prominent position in the aerodynamic analysis nowadays. Moreover, HiFiLES and PyFR represent the high-order methods which should be very flexible and accurate; however, further development of these applications may be necessary. Finite Element Method (FEM) software, e.g. Calculix, might also be used in an analysis process. Nevertheless, these methods are focused more on the structural analysis than the aerodynamic analysis.

We can agree with the statement in [143] that the parametric design systems, such as OpenVSP or Larosterna, are excellent tools for a rapid prototyping technique in aeronautics.

The free software for the design, analysis, modelling, and simulation of the fixed-wing configuration is extensive. There is not such a wide selection for the rotary-wing configuration; however, the selection should be satisfactory. The creation of a hybrid UAV should also be possible, at worst in a limited degree. On the other hand, the number of the free software for the flapping-wing and especially for the lighter-than-air configuration may not be sufficient.

The future of any UAV system is open to a wide range of research topics: collision avoidance, autonomous formation flight, navigation without the use of GPS, the automatic detection of a potential criminal, and other research topics.

ACKNOWLEDGMENT

This work was supported by the Internal Grant Agency of Tomas Bata University in Zlín under the projects No. IGA/FAI/2015/001 and IGA/FAI/2014/006.

REFERENCES

[1] X.Q. Chen, Y.Q. Chen, and J.G. Chase. *Mobile Robots - State of the Art in Land, Sea, Air, and Collaborative Missions*. Croatia: In-Teh, May 2009, pp.177-201.

[2] N. M. Jodeh. *Development of Autonomous Unmanned Aerial Vehicle Research Platform: Modeling, Simulating, and Flight Testing*. Ohio, USA: Wright-Patterson Air Force Base, March 2006, 185 p.

[3] T. Abdunabi. *Modelling and Autonomous Flight Simulation of a Small Unmanned Aerial Vehicle*. Sheffield, UK: The University of Sheffield, August 2006, 61 p.

[4] T. M. Foster. *Dynamic Stability and Handling Qualities of Small Unmanned-Aerial-Vehicles*. Brigham, USA: Brigham Young University, April 2005, 125 p.

[5] R. Austin. *Unmanned Aircraft Systems: UAVs Design, Development and Deployment*. Wiltshire, UK: Wiley, 2010, 332 p.

[6] D. G. Hull. *Fundamentals of Airplane Flight Mechanics*. Springer, 2007, 298 p.

[7] T.V. Chelaru, V. Pana, and A. Chelaru. "Dynamics and flight control of the UAV formations," *WSEAS Transactions on Systems and Control*, vol. 4 no. 4, pp. 198-210, April 2009.

[8] R. Carmichael. (2013, January 31). *Public Domain Aeronautical Software*. [Online]. Available: http://www.pdas.com

[9] R. Carmichael. (2013, March 28). *Public Domain Aeronautical Software: Contents*. [Online]. Available: http://www.pdas.com/contents15.html

[10] R. Carmichael. (2013, April 21). *Description of Digital Datcom*. [Online]. Available: http://www.pdas.com/datcomDescription.html

[11] R. Carmichael. (2013, February 7). *Addressable Configurations in Digital Datcom*. [Online]. Available: http://www.pdas.com/datcomTable1.html

[12] JSBSim contributors. *JSBSim Open Source Flight Dynamics Model*. [Online]. Available: http://jsbsim.sourceforge.net/

[13] JSBSim contributors. (2005, December 31). *Aeromatic: version 0.9*. [Online]. Available: http://jsbsim.sourceforge.net/aeromatic2.html

[14] FlightGear contributors. *FlightGear Flight Simulator: Introduction*. [Online]. Available: http://www.flightgear.org/about/

[15] FlightGear contributors. *FlightGear Flight Simulator: Features*. [Online]. Available: http://www.flightgear.org/about/features/

[16] FlightGear contributors. (2014, May 12). *FlightGear Wiki - UIUC*. [Online]. Available: http://wiki.flightgear.org/UIUC

[17] D. Hodson. (2014, January 10). *OpenEaagles Simulation Framework*. [Online]. Available: http://www.openeaagles.org

[18] D. Hodson. (2012, December 03). *OpenEaagles Simulation Framework: Overview*. [Online]. Available: http://www.openeaagles.org/wiki/doku.php?id=overview:overview

[19] D. Hodson, D. Gehl, and R. Baldwin. "Building Distributed Simulations Utilizing the EAAGLES Framework," *Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC)*, vol. 5, no. 2, May 2006.

[20] D. Hodson. "OPENEAAGLES, An Open Source Simulation Framework," *A Publication of the AIAA Modeling and Simulation Technical Committee*, vol. 1, no. 1, January 2008.

[21] M. Drela. (2013, December 23). *XFOIL: Subsonic Airfoil Development System*. [Online]. Available: http://web.mit.edu/drela/Public/web/xfoil/

[22] Redhammer Consulting Ltd. (2010). *TORNADO*. [Online]. Available: http://www.redhammer.se/tornado/index.html

[23] P. Zagórski and C. David. (2013). *Aerospace Blockset for Xcos*. [Online]. Available: http://forge.scilab.org/index.php/p/aerospace-blockset/

[24] Luca Petricca, Per Ohlckers, and Christopher Grinde. "Micro- and Nano-Air Vehicles: State of the Art," *International Journal of Aerospace Engineering*, vol. 2011, Article ID 214549, 17 pages, 2011. doi:10.1155/2011/214549

[25] M. G. Perhinschi, M. R. Napolitano, and S. Tamayo. "Integrated Simulation Environment for Unmanned Autonomous Systems—Towards a Conceptual Framework," *Modelling and Simulation in Engineering*, vol. 2010, Article ID 736201, 12 pages, 2010. doi:10.1155/2010/736201

[26] Madhava Syamlal, Thomas J. O'Brien, Sofiane Benyahia, Aytekin Gel, and Sreekanth Pannala. "Open-Source Software in Computational Research: A Case Study," *Modelling and Simulation in Engineering*, vol. 2008, Article ID 937542, 10 pages, 2008. doi:10.1155/2008/937542

[27] W. L. Oberkampf and T. G. Trucano. "Verification and validation in computational fluid dynamics," *Sandia Report SAND2002-0529*, Sandia National Laboratories, Albuquerque, NM, USA, March 2002.

[28] T. J. Mueller and J. D. DeLaurier. Aerodynamics of small vehicles. *Annu. Rev. Fluid Mech*, vol. 35, no. 1, pp. 89-111. 2003. DOI: 10.1146/annurev.fluid.35.101101.161102.

[29] P. B. S. Lissaman. Low-reynolds-number airfoils. *Annu. Rev. Fluid Mech*, vol. 15, no. 1, pp. 223-239. 1983. DOI: 10.1146/annurev.fl.15.010183.001255.

[30] P. Shankar, W. Chung, J. Husman, and V. Wells. "A novel software framework for teaching aircraft dynamics and control," *Computer Applications in Engineering Education*, 9 pages. 2013. DOI: 10.1002/cae.21579.

[31] H. Boussalis, K. Valavanis, D. Guillaume, F. Pena, E. U. Diaz, and J. Alvarenga. "Control of a simulated wing structure with multiple

segmented control surfaces," *21st Mediterranean Conference on Control and Automation (Med)*, pp. 501-506. 2013.

[32] C. Lafountain, K. Cohen, and S. Abdallah. Use of XFOIL in design of camber-controlled morphing UAVs. *Computer Applications in Engineering Education,* vol. 20, no. 4, pp. 673-680. 2012. DOI: 10.1002/cae.20437.

[33] A. Swarup and Sudhir. "Comparison of Quadrotor Performance Using Backstepping and Sliding Mode Control," *Proceedings of the 2014 International Conference on Circuits, Systems and Control*, Interlaken, Switzerland, February 22-24, 2014, pp. 79-82.

[34] Chao Yun, Xiaomin Li. "Design of UAV Flight Simulation Software Based on Simulation Training Method," *WSEAS TRANSACTIONS on INFORMATION SCIENCE and APPLICATIONS*, volume 10, issue 2, February 2013, pp. 37-46.

[35] G. Gol, N. F. Bayraktar, and E. Kiyak. "PID Controlling of the Quadrotor and Sensor Performance Tests," *INTERNATIONAL JOURNAL OF CIRCUITS, SYSTEMS AND SIGNAL PROCESSING*, Volume 8, 2014, pp. 266-275.

[36] Y. Naidoo, R. Stopforth, and G. Bright. "Development of an UAV for search & rescue applications: mechatronic integration for a quadrotor helicopter," *IEEE Africon 2011*, Livingstone, Zambia, September 13-15, 2011.

[37] O. Gonzalez-Espasandin, T. J. Leo, and E. Navarro-Arevalo. "Fuel cells: A real option for unmanned aerial vehicles propulsion," *The Scientific World Journal*. Volume 2014, Article ID 497642, 12 pages. DOI: 10.1155/2014/497642.

[38] B. Uragun. "Energy efficiency for unmanned aerial vehicles," *2011 10th International Conference on Machine Learning and Applications*, Honolulu, Hawaii, USA, December 18-21, 2011. DOI: 10.1109/ICMLA.2011.159

[39] M. B. Srikanth, Z. T. Dydek, A. M. Annaswamy, and E. Lavretsky. "A robust environment for simulation and testing of adaptive control for mini-UAVs," *2009 American Control Conference*, St. Louis, MO, USA, June 10-12, 2009.

[40] E. Kahale, P. C. Garcia, and Y. Bestaoui. "Autonomous path tracking of a kinematic airship in presence of unknown gust," *J. Intell. Robot. Syst*, vol. 69, no. 1-4, pp. 431-446. 2013. DOI: 10.1007/s10846-012-9709-2

[41] U.S. Department of Transportation. *Rotorcraft Flying Handbook*. [Online]. U.S. Department of Transportation, Federal Aviation Administration, Flight Standards Service, FAA-H-8083-21, Washington, D.C., USA, 2000. Available: http://www.faa.gov/regulations_policies/handbooks_manuals/aircraft/media/faa-h-8083-21.pdf

[42] U.S. Department of Transportation. *Helicopter Flying Handbook*. [Online]. U.S. Department of Transportation, Federal Aviation Administration, Flight Standards Service, FAA-H-8083-21A, USA, 2012. Available: http://www.faa.gov/regulations_policies/handbooks_manuals/aviation/helicopter_flying_handbook/media/helicopter_flying_handbook.pdf

[43] Dai Jing and Wang Haifeng. "System Health Management for Unmanned Aerial Vehicle: Conception, State-of-Art, Framework and Challenge," *The 11th IEEE International Conference on Electronic Measurement & Instruments*, Harbin, China, August 16-18, 2013.

[44] Tomáš Vogeltanz and Roman Jašek. "Free Software for the Modelling and Simulation of a mini-UAV," *Mathematics and Computers in Science and Industry*, Varna, Bulgaria, September 13-15, 2014, pp. 210-215.

[45] S. Kurnaz, O. Cetin, and O. Kaynak. "Adaptive neuro-fuzzy inference system based autonomous flight control of unmanned air vehicles," *Expert Systems with Applications*, vol. 37, issue 2, 2010, pp. 1229-1234.

[46] P. Fabiani, V. Fuertes, A. Piquereau, R. Mampey, F. Teichteil-Königsbuch. "Autonomous flight and navigation of VTOL UAVs: from autonomy demonstrations to out-of-sight flights," *Aerospace Science and Technology*, vol. 11, issue 2-3, pp. 183-193, 2007.

[47] J. Qi, J. Liu, B. Zhao, S. Mei, J. Han, and H. Shang, "Visual simulation system design of soft-wing UAV based on FlightGear," *IEEE International Conference on Mechatronics and Automation, IEEE ICMA 2014*, Tianjin, China, August 2-5, 2014, pp. 1188-1192.

[48] O. Cetin, S. Kurnaz, and O. Kaynak. "Fuzzy logic based approach to design of autonomous landing system for unmanned aerial vehicles," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 61, issue 1-4, 2011, pp. 239-250.

[49] V. Kumar, H. Yong, D. Min, and E. Choi, "Auto landing control for small scale unmanned helicopter with flight gear and HILS," *Proceeding - 5th International Conference on Computer Sciences and Convergence Information Technology, ICCIT 2010*, Seoul, Korea, November 30 - December 2, 2010, pp. 676-681.

[50] J. Ye, H. Guo, S. Tang, and Q. Wang. "The research on visual flight simulation for unmanned helicopter," *Communications in Computer and Information Science*, vol. 325 CCIS, 2012, pp. 332-341.

[51] F. Mazhar, A. M. Khan, I. A. Chaudhry, and M. Ahsan. "On using neural networks in UAV structural design for CFD data fitting and classification," *Aerospace Science and Technology*, vol. 30, issue 1, 2013, pp. 210-225.

[52] David W. Babka. *Flight Testing in a Simulation Based Environment*. San Luis Obispo, California, USA: California Polytechnic University, 2011, 18 p. [online] Available: http://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=1045&context=aerosp

[53] Douglas J. Pederson. *Conceptual design tool to analyze electrochemically-powered micro air vehicles*. Ohio, USA: Wright-Patterson Air Force Base, March 2011, 193 p.

[54] Q. R. Wald. "The aerodynamics of propellers," *Progress in Aerospace Sciences*, vol. 42, issue 2, 2006, pp. 85-128.

[55] Jon S. Berndt. "JSBSim: An open source flight dynamics model in C++," *Collection of Technical Papers - AIAA Modeling and Simulation Technologies Conference*, vol. 1, 2004, pp. 261-287.

[56] J. S. Berndt, A. De Marco, "Progress on and usage of the open source flight dynamics model software library, JSBSim," AIAA Modeling and Simulation Technologies Conference, Chicago, Illinois, USA, August 10-12, 2009.

[57] Jon S. Berndt and the JSBSim Development Team. (2011, June 9). *JSBSim: An open source, platform-independent, flight dynamics model in C++*. [Online]. Available: http://jsbsim.sourceforge.net/JSBSimReferenceManual.pdf

[58] Ser Keong Lim and Chua Ching Hao. (2012). *Modeling Unmanned Vehicle System*. [Online]. Available: https://engineering.purdue.edu/HSL/uploads/papers/UVG_S10.pdf

[59] Anders Gidenstam. (2013, June 25). *Lighter-than-air support for airship and balloon simulation in JSBSim and FlightGear*. [Online]. Available: http://www.gidenstam.org/FlightGear/Airships/

[60] Michael Basler, Martin Spott, Stuart Buchanan, Jon Berndt et al. (2014, August 11). *The FlightGear Manual*. [Online]. Available: http://mapserver.flightgear.org/getstart.pdf

[61] Michael S. Selig, Rob Deters, and Glen Dimock. (2002, March 10). *Aircraft Dynamics Models for Use with FlightGear: Modeling and Simulation*. [Online]. Available: http://m-selig.ae.illinois.edu/apasim/Aircraft-uiuc.html

[62] J. Zhang, Q. Geng, and Q. Fei, "UAV flight control system modeling and simulation based on flightgear," *IET Conference Publications*, Xiamen, China, March 3-5, 2012.

[63] 3DRobotics. (2014, May 27). *HIL Quad Simulator*. [Online]. Available: http://copter.ardupilot.com/wiki/hil-quad/

[64] R. Carmichael. (2014, February 13). *Properties Of The U.S. Standard Atmosphere 1976*. [Online]. Available: http://www.pdas.com/atmos.html

[65] R. Carmichael. (2013, March 5). *Real Gas Properties*. [Online]. Available: http://www.pdas.com/gasp.html

[66] R. Carmichael. (2013, February 8). *Thermodynamic and Transport Properties of Fluids*. [Online]. Available: http://www.pdas.com/fluid.html

[67] R. Carmichael. (2013, February 12). *vuCalc - A Compressible Flow Calculator*. [Online]. Available: http://www.pdas.com/vucalc.html

[68] R. Carmichael. (2013, February 12). *A segmented mission analysis program for low and high speed aircraft (NSEG)*. [Online]. Available: http://www.pdas.com/nseg.html

[69] R. Carmichael. (2013, February 27). *Conical relaxation program for supersonic wing design and analysis (COREL)*. [Online]. Available: http://www.pdas.com/corel.html

[70] R. Carmichael. (2013, March 9). *W12SC3: Supersonic Wing Design And Analysis*. [Online]. Available: http://www.pdas.com/w12sc3.html

[71] R. Carmichael. (2013, February 13). *Two-Dimensional Grids About Airfoils And Other Shapes By The Use Of Poisson's Equation (GRAPE)*. [Online]. Available: http://www.pdas.com/grape.html

[72] R. Carmichael. (2013, February 13). *PROFILE - The Eppler airfoil code*. [Online]. Available: http://www.pdas.com/eppler.html

[73] R. Carmichael. (2013, March 5). *Modified Strip Analysis Method For Predicting Wing Flutter At Subsonic To Hypersonic Speeds*. [Online]. Available: http://www.pdas.com/flutter.html

[74] R. Carmichael. (2013, March 5). *Induced Drag from Span Load Distribution*. [Online]. Available: http://www.pdas.com/induced.html

[75] R. Carmichael. (2013, February 13). *Mean Aerodynamic Chord of a Wing*. [Online]. Available: http://www.pdas.com/getmac.html

[76] R. Carmichael. (2013, February 9). *Mass Properties Of A Rigid Structure*. [Online]. Available: http://www.pdas.com/massprop.html

[77] R. Carmichael. (2013, March 7). *NACA Airfoil Coordinates*. [Online]. Available: http://www.pdas.com/naca456.html

[78] R. Carmichael. (2013, February 10). *PANAIR: Predicting Subsonic Or Supersonic Linear Potential Flows About Arbitrary Configurations Using A Higher Order Panel Method*. [Online]. Available: http://www.pdas.com/panair.html

[79] R. Carmichael. (2013, February 14). *Input Pre-Processor for PanAir*. [Online]. Available: http://www.pdas.com/panin.html

[80] R. Carmichael. (2013, March 9). *NASA-AMES WingBody Panel Code*. [Online]. Available: http://www.pdas.com/wingbody.html

[81] R. Carmichael. (2013, March 8). *Aeroelastic Analysis For Rotorcraft In Flight Or In A Wind Tunnel (ROTOR)*. [Online]. Available: http://www.pdas.com/rotor.html

[82] R. Carmichael. (2013, March 9). *Turbulent Skin Friction by the Reference Temperature Method of Sommer and Short*. [Online]. Available: http://www.pdas.com/turbsf.html

[83] R. Carmichael. (2013, March 9). *V/STOL Aircraft Sizing And Performance (VASCOMP II)*. [Online]. Available: http://www.pdas.com/vascomp.html

[84] R. Carmichael. (2013, March 9). *Minimum Drag Camber Surface by Vortex Lattice*. [Online]. Available: http://www.pdas.com/vlmd.html

[85] Holy Cows, Inc.. *Datcom by Holy Cows, Inc.*. [Online]. (accessed 2014, November 24). Available: http://www.holycows.net/datcom/

[86] DATCOM-GUI contributors. (2011, August 2). *datcom-gui: Development of a GUI for the DATCOM program*. [Online]. Available: http://code.google.com/p/datcom-gui/

[87] R. Carmichael. (2013, February 14). *MakeWgs*. [Online]. Available: http://www.pdas.com/makewgs.html

[88] R. Carmichael. (2013, February 26). *Three View Program*. [Online]. Available: http://www.pdas.com/3view.html

[89] R. Carmichael. (2013, February 8). *Hidden Line Program*. [Online]. Available: http://www.pdas.com/hlp.html

[90] R. Carmichael. (2013, January 21). *Conversion to LaWGS*. [Online]. Available: http://www.pdas.com/2wgs.html

[91] R. Carmichael. (2013, February 11). *VRML World*. [Online]. Available: http://www.pdas.com/wgs2wrl.html

[92] Richard Eppler and Dan M. Somers. *A Computer Program for the Design and Analysis of Low-Speed Airfoils*. NASA Technical Memorandum 80210, August 1980.

[93] J. L. Lundry. "Calculation of lift and induced drag from sparse span loading data," *Journal of Aircraft*, vol. 14, no. 3, pp. 309-311, 1977.

[94] R. Carmichael. (2010, September 27). *Example 4 - Wing Similar to B-2 - page 1*. [Online]. Available: http://www.pdas.com/macex13.html

[95] R. Carmichael. (2010, September 27). *Example 4 - Wing Similar to B-2 - page 2*. [Online]. Available: http://www.pdas.com/macex14.html

[96] R. Carmichael. (2010, September 27). *Example 4 - Wing Similar to B-2 - page 3*. [Online]. Available: http://www.pdas.com/macex15.html

[97] Reid A. Hull, John L. Gilbert, and Phillip J. Klich. *Computer Program for Determining Mass Properties of a Rigid Structure*. NASA Technical Memorandum 78681, March 1978.

[98] *Reynolds number calculator*. [Online]. (accessed 2014, December 12). Available: http://airfoiltools.com/calculator/reynoldsnumber

[99] *Airfoil Tools*. [Online]. (accessed 2014, December 12). Available: http://airfoiltools.com/

[100] R. Carmichael. (2010, November 3). *Computation of NACA Airfoil Coordinates*. [Online]. Available: http://www.pdas.com/naca456pdas.html

[101] Ralph L. Carmichael. "Algorithm for Calculating Coordinates of Cambered Naca Airfoils At Specified Chord Locations," *1st AIAA, Aircraft, Technology Integration, and Operations Forum*, November 2001.

[102] Gary R. Saaris. *A502I User's Manual-PAN AIR Technology Program for Solving Problems of Potential Flow about Arbitrary Configurations*. Cage Code 81205, Document no. D6-54703, Boing, February 1992.

[103] T. Derbyshire and K.W. Sidwell. *PAN AIR Summary Document, (Version 1.0)*. NASA Contractor Report 3250, 1982.

[104] Gnuplot contributors. *gnuplot homepage*. [Online]. (accessed 2014, December 12) Available: http://www.gnuplot.info/

[105] Ilan Kroo and Juan Alonso. *Skin Friction and Roughness Drag*. [Online]. (accessed 2014, December 16) Available: http://adg.stanford.edu/aa241/drag/skinfriction.html

[106] R. Carmichael. (2013, April 21). *Digital Datcom*. [Online]. Available: http://www.pdas.com/datcom.html

[107] R. Carmichael. (2013, February 7). *Program Modules of Digital Datcom*. [Online]. Available: http://www.pdas.com/datcomc.html

[108] J.E. Williams and S.R. Vukelich. *The USAF Stability And Control Digital Datcom: Volume I, Users Manual*. USAF Technical Report AFFDL-TR-79-3032, April 1979.

[109] Mustafa Turan. *Tools for the conceptual design and engineering analysis of micro air vehicles*. Ohio, USA: Wright-Patterson Air Force Base, March 2009, 156 p.

[110] Xian-Zhong Gao, Zhong-XiHou, ZhengGuo, Rong-FeiFan, Xiao-QianChen. "Analysis and design of guidance-strategy for dynamic soaring with UAVs," *Control Engineering Practice*, vol. 32, pp. 218-226. 2014.

[111] Mark Drela. (2008, October 1). *DSOPT: Dynamic Soaring simulation and optimization program*. [Online]. Available: http://web.mit.edu/drela/Public/web/dsopt/summary.txt

[112] Harold Youngren and Mark Drela. (2005, December 4). *DFDC 0.70 User Primer*. [Online]. Available inside: http://web.mit.edu/drela/Public/web/dfdc/DFDC_v0.70.zip

[113] Harold Youngren, Mark Drela, and Scott Sanders. (2005, December 10). *DFDC Summary*. [Online]. Available: http://web.mit.edu/drela/Public/web/dfdc/

[114] Philip Carter. (2014, October 25). *DFDC: Ducted Fan Design Code - A Diamond in the Rough*. [Online]. Available: http://www.esotec.org/sw/DFDC.html

[115] Mark Drela and Harold Youngren. (2011, February 10). *XROTOR Download Page*. [Online]. Available: http://web.mit.edu/drela/Public/web/xrotor/

[116] Mark Drela and Harold Youngren. (2003, November 13). *XROTOR User Guide*. [Online]. Available: http://web.mit.edu/drela/Public/web/xrotor/xrotor_doc.txt

[117] C. Thipyopas, S. Kaewsutthi, and A. Tohwae-A-Yee. High performance propeller system for a multi-mission micro aerial vehicle. *International Journal of Micro Air Vehicles,* vol. 5, no. 3, pp. 179-191. 2013.

[118] Philip Carter. (2014, April 4). *CROTOR: XROTOR on Steroids*. [Online]. Available: http://www.esotec.org/sw/crotor.html

[119] Philip Carter. (2011, October 5). *SUBROUTINE CROTOR User Guide*. [Online]. Available: http://www.esotec.org/sw/dl/CRotor_doc.txt

[120] Philip Carter. (2014, April 5). *ESPROP*. [Online]. Available: http://www.esotec.org/sw/esprop.html

[121] Philip Carter. (2014, January 19). *SUBROUTINE ESLOFTX User Guide*. [Online]. Available: http://www.esotec.org/sw/dl/Esloftx_doc.txt

[122] Mark Drela. (2007, December 23). *QPROP: Propeller/Windmill Analysis and Design*. [Online]. Available: http://web.mit.edu/drela/Public/web/qprop/

[123] Mark Drela. (2006, June). *QPROP Formulation*. [Online]. Available: http://web.mit.edu/drela/Public/web/qprop/qprop_theory.pdf

[124] A. Betz. *Airscrews with minimum energy loss*. Report, Kaiser Wilhelm Institute for Flow Research, 1919.

[125] S. Goldstein. "On the vortex theory of screw propellers," *Proceedings of the Royal Society*, 123, 1929.

[126] T. Theodorsen. *Theory of Propellers*. McGraw-Hill, New York, 1948.

[127] E.E. Larrabee and S.E. French. "Minimum induced loss windmills and propellers," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 15, issue 1-3, pp. 317–327, 1983. DOI: 10.1016/0167-6105(83)90201-5.

[128] Mark Drela. (2007, July 6). *QPROP User Guide*. [Online]. Available: http://web.mit.edu/drela/Public/web/qprop/qprop_doc.txt

[129] Mark Drela. (2005, October 4). *QMIL User Guide*. [Online]. Available: http://web.mit.edu/drela/Public/web/qprop/qmil_doc.txt

[130] Mark Drela. (2001, November 30). *XFOIL 6.9 User Primer*. [Online]. Available: http://web.mit.edu/drela/Public/web/xfoil/xfoil_doc.txt

[131] Nuno António Silva. *Parametric Design, Aerodynamic Analysis and Parametric Optimization of a Solar UAV*. Lisboa, Portugal: Instituto Superior Técnico, May 2014, 10 p.

[132] Mark Drela and Harold Youngren. (2014, February 23). *AVL Overview*. [Online]. Available: http://web.mit.edu/drela/Public/web/avl/

[133] Mark Drela and Harold Youngren. (2010, August 18). *AVL 3.30 User Primer*. [Online]. Available: http://web.mit.edu/drela/Public/web/avl/avl_doc.txt

[134] Philip Carter. (2014). *SOFTWARE*. [Online]. Available: http://www.esotec.org/sw/swhome.html

[135] Mark Drela. (2014, August 30). *Index of /drela/Public/web*. [Online]. Available: http://web.mit.edu/drela/Public/web/

[136] Mark Drela. (2010, March 20). *TASOPT 2.00: Transport Aircraft System OPTimization - Technical Description*. [Online]. Available: http://web.mit.edu/drela/Public/web/tasopt/TASOPT_doc.pdf

[137] Dmitri Kuzmin. *Introduction to Computational Fluid Dynamics*. [Online]. (accessed 2014, December 19). Available: http://www.mathematik.uni-dortmund.de/~kuzmin/cfdintro/lecture1.pdf

[138] T. D. Economon. (2014, October 28). *SU2: The Open-Source CFD Code*. [Online]. Available: https://github.com/su2code/SU2/wiki

[139] F. Palacios, M. R. Colonno, A. C. Aranake, A. Campos, S. R. Copeland, T. D. Economon, Amrita K. Lonkar, Trent W. Lukaczyk, Thomas W. R. Taylor, and Juan J. Alonso, "Stanford University Unstructured (SU^2): An open-source integrated computational environment for multi-physics simulation and design," *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, Grapevine (Dallas/Ft. Worth Region), Texas, USA, January 7-10, 2013.

[140] T. D. Economon. (2014, November 6). *Quick Start*. [Online]. Available: https://github.com/su2code/SU2/wiki/Quick-Start

[141] T. D. Economon. (2014, November 14). *Optimal Shape Design of a Rotating Airfoil*. [Online]. Available: https://github.com/su2code/SU2/wiki/Optimal-Shape-Design-of-a-Rotating-Airfoil

[142] OpenFOAM Foundation. (2014). *Features of OpenFOAM*. [Online]. Available: http://www.openfoam.org/features/

[143] J. Byrne, P. Cardiff, A. Brabazon, and M. O'Neill. "Evolving parametric aircraft models for design exploration and optimisation," *Neurocomputing*, vol. 142, 39-47, 2014.

[144] Gertjan Glabeke. *The influence of wind turbine induced turbulence on ultralight aircraft, a CFD analysis*. Oostende, Belgium: Katholieke Hogeschool VIVES, 2011, 99 p.

[145] OpenFOAM Foundation. (2014). *Standard Solvers*. [Online]. Available: http://www.openfoam.org/features/standard-solvers.php

[146] OpenFOAM Foundation. (2014). *ODE System Solvers*. [Online]. Available: http://www.openfoam.org/features/ODE-solvers.php

[147] OpenFOAM Foundation. (2014). *Numerical Method*. [Online]. Available: http://www.openfoam.org/features/numerical-method.php

[148] OpenFOAM Foundation. (2014). *Dynamic Meshes*. [Online]. Available: http://www.openfoam.org/features/mesh-motion.php

[149] Hester Bijl. *Flow around flapping wings with Open FOAM at Aerodynamics, TU Delft*. [Online]. (accessed 2014, December 22) Available: http://www.tudelft.nl/fileadmin/UD/MenC/Support/Internet/TU_Website/TU_Delft_portal/Onderzoek/Kenniscentra/Delft_Research_Centres/Computational_Science/Events/Seminars/previous/doc/Hester_1.pdf

[150] EDF R&D. (2014). *Description of Code_Saturne*. [Online]. Available: http://code-saturne.org/cms/features

[151] EDF R&D. (2014, May). *Code Saturne documentation: Code Saturne version 3.3.0 practical user's guide*. [Online]. Available: http://code-saturne.org/cms/sites/default/files/docs/3.3/user.pdf

[152] Y. Fournier, J. Bonelle, C. Moulinec, Z. Shang, A. G. Sunderland, and J. C. Uribe. "Optimizing Code_Saturne computations on Petascale systems," *Computers and Fluids*, vol. 45, issue 1, pp. 103-108, 2011.

[153] EDF R&D. (2014). *Numerical Method*. [Online]. Available: http://code-saturne.org/cms/features/numerics

[154] EDF R&D. (2014). *Mesh flexibility*. [Online]. Available: http://code-saturne.org/cms/features/mesh

[155] EDF R&D. (2014). *Code_Saturne coupling*. [Online]. Available: http://code-saturne.org/cms/features/modules/coupling

[156] HiFiLES Developers. (2014). *HiFiLES: High Fidelity Large Eddy Simulation*. [Online]. Available: https://hifiles.stanford.edu/

[157] M. López-Morales, J. Bull, J. Crabill, T. D. Economon, D. E. Manosalvas, J. Romero, A. Sheshadri, J. E. Watkins, D. Williams, F. Palacios, and A. Jameson, "Verification and validation of HiFiLES: A high-order LES unstructured solver on multi-GPU platforms," *32nd AIAA Applied Aerodynamics Conference*, Atlanta, Georgia, USA, June 16-20, 2014.

[158] Vincent Lab. (2015). *PyFR: Home*. [Online]. Available: http://www.pyfr.org/

[159] Peter Vincent. (2014). *PyFR: A GPU-Accelerated Next-Generation Computational Fluid Dynamics Python Framework*. [Online]. Available: http://www.techenablement.com/pyfr-a-gpu-accelerated-next-generation-computational-fluid-dynamics-python-framework/

[160] F. D. Witherden, A. M. Farrington, and P. E. Vincent. "PyFR: An open source framework for solving advection-diffusion type problems on streaming architectures using the flux reconstruction approach," *Computer Physics Communications*, vol. 185, issue 11, pp. 3028-3040, 2014.

[161] Olivier de Weck. (2005, January 6). *Computer Aided Design (CAD)*. [Online]. Available: http://ocw.mit.edu/courses/aeronautics-and-astronautics/16-810-engineering-design-and-rapid-prototyping-january-iap-2005/lecture-notes/l4.pdf

[162] FreeCAD Contributors. (2014, September 27). *About FreeCAD*. [Online]. Available: http://www.freecadweb.org/wiki/index.php?title=About_FreeCAD

[163] FreeCAD Contributors. (2014, September 1). *Getting started*. [Online]. Available: http://www.freecadweb.org/wiki/index.php?title=Getting_started

[164] FreeCAD Contributors. (2014, December 6). *Feature list*. [Online]. Available: http://www.freecadweb.org/wiki/index.php?title=Feature_list

[165] FreeCAD Contributors. (2013, December 8). *Aeroplane*. [Online]. Available: http://www.freecadweb.org/wiki/index.php?title=Aeroplane

[166] J. W. Kim, K.-K. Kang, and J. H. Lee, "Template-based traditional building component modelling," *International Conference on Advanced Communication Technology, ICACT*, Pyeongchang, Korea (South), February 16-19, 2014, pp. 653-656.

[167] BRL-CAD Contributors. *About BRL-CAD*. [Online]. (accessed 2014, December 22). Available: http://brlcad.org/d/about

[168] BRL-CAD Contributors. *Overview*. [Online]. (accessed 2014, December 22). Available: http://brlcad.org/wiki/Overview

[169] J. Keyser, T. Culver, M. Foskey, S. Krishnan, and D. Manocha. "ESOLID - A system for exact boundary evaluation," *CAD Computer Aided Design*, vol. 36, issue 2, pp. 175-193, 2004.

[170] H. E. Konokman, A. Kayran, and M. Kaya, "Analysis of aircraft survivability against fragmenting warhead threat," *55th AIAA/ASMe/ASCE/AHS/SC Structures, Structural Dynamics, and Materials Conference*, National Harbor, Maryland, USA, January 13-17, 2014.

[171] QCAD Contributors. (2014, October 24). *QCAD - 2D CAD for Windows, Linux and Mac*. [Online]. Available: http://www.qcad.org/en/

[172] QCAD Contributors. (2014, November 27). *QCAD Features*. [Online]. Available: http://www.qcad.org/en/qcad-documentation/qcad-features

[173] QCAD Contributors. (2014, November 26). *The QCAD 3 Scripting Interface*. [Online]. Available: http://www.qcad.org/en/qcad-documentation/qcad-scripting

[174] OpenVSP Contributors. (2012, January 10). *OpenVSP*. [Online]. Available: https://github.com/nasa/OpenVSP

[175] OpenVSP Contributors. (2014, December 11). *OpenVSP*. [Online]. Available: http://openvsp.org/

[176] OpenVSP Contributors. (2014, December 2). *VSP Hangar*. [Online]. Available: http://hangar.openvsp.org/

[177] A. S. Hahn, "Vehicle sketch pad: A parametric geometry modeler for conceptual aircraft design," *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, Orlando, Florida, USA, January 4-7, 2010.

[178] D. Böhnke, B. Nagel, M. Zhang, and A. Rizzi, "Towards a collaborative and integrated set of open tools for aircraft design," *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, Grapevine (Dallas/Ft. Worth Region), Texas, USA, January 7-10, 2013.

[179] J. R. Gloudemans and R. McDonald, "Improved geometry modeling for high fidelity parametric design," *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, Orlando, Florida, USA, January 4-7, 2010.

[180] J. B. Belben and R. A. McDonaldy, "Enabling rapid conceptual design using geometry-based multi-fidelity models in VSP," *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, Grapevine (Dallas/Ft. Worth Region), Texas, USA, January 7-10, 2013.

[181] VAMPzero Contributors. *VAMPzero - Conceptual Design for the Needs of MDO*. [Online]. (accessed 2014, December 22). Available: http://code.google.com/p/vampzero/

[182] D. Böhnke, B. Nagel, and V. Gollnick, "Explicit modeling of technology improvement over time in conceptual aircraft design," *29th Congress of the International Council of the Aeronautical Sciences, ICAS 2014*, St. Petersburg, Russia, September 7-12, 2014.

[183] A. Rizzi, M. Zhang, B. Nagel, D. Boehnke, and P. Saquet, "Towards a unified framework using CPACS for geometry management in aircraft design," *50th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, Nashville, Tennessee, USA, January 9-12, 2012.

[184] CPACS Contributors. *CPACS – A Common Language for Aircraft Design*. [Online]. (accessed 2014, December 22). Available: http://code.google.com/p/cpacs/

[185] TIGL Contributors. *TIGL: A library for generating 3D geometries from parametrized CPACS/XML data sets*. [Online]. (accessed 2014, December 22). Available: http://code.google.com/p/tigl/

[186] N. T. Frink. (2006, August 25). *TetrUSS: CFD software for complex real-world aerodynamics problems*. [Online]. Available: http://tetruss.larc.nasa.gov/index.html

[187] N. T. Frink. (2009, March 27). *Requesting TetrUSS Software*. [Online]. Available: http://tetruss.larc.nasa.gov/download.html

[188] N. T. Frink, S. Z. Pirzadeh, P. C. Parikh, M. J. Pandya, and M. K. Bhat. "NASA tetrahedral unstructed software system (TetrUSS)," *Aeronautical Journal*, vol. 104, issue 1040, pp. 491-499, 2000.

[189] CEASIOM Contributors. (2014, September). *CEASIOM: Computerised Environment for Aircraft Synthesis and Integrated Optimisation Methods*. [Online]. Available: http://www.ceasiom.com/index.php

[190] CEASIOM Contributors. (2014) *CEASIOM Modules*. [Online]. Available: http://www.ceasiom.com/ceasiom-modules.html

[191] CEASIOM Contributors. (2014, April 11) *CEASIOM NEWSLETTER*. [Online]. Available: http://www.ceasiom.com/newsletter.html

[192] James Goppert. (2012, September 2). *OpenFDM: An open source flight dynamics library for Modelica*. [Online]. Available: https://github.com/arktools/openfdm

[193] Ben Parslew. *Downloads*. [Online]. (accessed 2014, December 27). Available: http://www.flappingwings.co.uk/main/downloads

[194] Ben Parslew. *Gallery*. [Online]. (accessed 2014, December 27). Available: http://www.flappingwings.co.uk/main/gallery

[195] Ben Parslew. *Simulating Avian Wingbeats and Wakes*. Manchester, UK: The University of Manchester, the Faculty of Engineering and Physical Sciences, 2012, 158 p.

[196] Ben Parslew. (2012, July 30). *Flapping Flight Simulation v1.1: User Manual*. [Online]. Available: http://www.flappingwings.co.uk/main/wp-content/uploads/2012/06/FlappingFlightSimulationManualV1.1.pdf

[197] Nathan Chronister. *FlapDesign - How to Use*. [Online]. (accessed 2014, December 29). Available: http://www.ornithopter.org/flapdesign.info.shtml

[198] Horst Räbiger. *Calculation tools for ornithopter models*. [Online]. (accessed 2014, December 29). Available: http://www.ornithopter.de/english/calculation.htm

[199] Markus Müller. (2014, December 28). *eCalc: the most reliable RC Calculator on the Web*. [Online]. Available: http://www.ecalc.ch/

[200] John Carri. *WebOCalc FAQ*. [Online]. (accessed 2014, December 4). Available: http://flbeagle.rchomepage.com/software/webocalc.html

[201] John Carri. *PowerCalc*. [Online]. (accessed 2014, December 4). Available: http://flbeagle.rchomepage.com/software/powercalc.html

[202] Adam One. *Welcome to Model Aircraft: Aerodynamics, Beginners' Guide and lots of info about R/C Model Aircraft*. [Online]. (accessed 2014, December 30). Available: http://adamone.rchomepage.com/index.html

[203] Adam One. *Aircraft Center of Gravity Calculator*. [Online]. (accessed 2014, December 30). Available: http://adamone.rchomepage.com/cg_calc.htm

[204] Adam One. *Canard Center of Gravity Calculator*. [Online]. (accessed 2014, December 30). Available: http://adamone.rchomepage.com/cg_canard.htm

[205] Adam One. *Trainer Design*. [Online]. (accessed 2014, December 30). Available: http://adamone.rchomepage.com/design.htm#calculate

[206] Adam One. *Calculate Stall Speed*. [Online]. (accessed 2014, December 30). Available: http://adamone.rchomepage.com/calc_stallspeed.htm

[207] Adam One. *Calculate Level Flight Speed*. [Online]. (accessed 2014, December 30). Available: http://adamone.rchomepage.com/calc_speed.htm

[208] Adam One. *Calculate Motor Efficiency*. [Online]. (accessed 2014, December 30). Available: http://adamone.rchomepage.com/calc_efficiency.htm

[209] Adam One. *Estimate Propeller's Static Thrus*. [Online]. (accessed 2014, December 30). Available: http://adamone.rchomepage.com/calc_thrust.htm

[210] Adam One. *Estimate Electric Motor & Prop Combo*. [Online]. (accessed 2014, December 30). Available: http://adamone.rchomepage.com/calc_motor.htm

[211] Adam One. *Beginners' Guide*. [Online]. (accessed 2014, December 30). Available: http://adamone.rchomepage.com/guide5.htm

[212] XFLR5 Contributors. *XFLR5*. [Online]. (accessed 2014, December 31). Available: http://www.xflr5.com/xflr5.htm

[213] XFLR5 Contributors. *XFLR5*. [Online]. (accessed 2014, December 31). Available: http://sourceforge.net/projects/xflr5/

[214] XFLR5 Contributors. (2013, February 28). *XFLR5: Analysis of foils and wings operating at low Reynolds numbers*. [Online]. Available: http://heanet.dl.sourceforge.net/project/xflr5/Guidelines.pdf

[215] J. Morgado. *JBLADE: a Propeller Design and Analysis Code*. [Online]. (accessed 2014, December 31). Available: https://sites.google.com/site/joaomorgado23/Home

[216] J. Morgado. (2013, September 20). *JBLADE v17 Tutorial*. [Online]. Available:

https://drive.google.com/viewerng/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxqb2FvbW9yZ2FkbzIzfGd4OjFjYzE3ZjhlYmY3YzBlNzE

[217] W. H. Mason. (2012, May 3). *Software for Aerodynamics and Aircraft Design (W.H. Mason, Virginia Tech)*. [Online]. Available: http://www.dept.aoe.vt.edu/~mason/Mason_f/MRsoft.html#foilgen

[218] Christophe Geuzaine and Jean-François Remacle. (2014, July 9). *Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities*. [Online]. Available: http://geuz.org/gmsh/

[219] 3DRobotics. *Mission Planner*. [Online]. (accessed 2014, December 31). Available: http://planner.ardupilot.com/

[220] Daniel Filkovic. *Apame - Aircraft 3D Panel Method*. [Online]. (accessed 2015, January 14). Available: http://www.3dpanelmethod.com/

[221] Daniel Filkovic. *Apame - Aircraft 3D Panel Method: Features*. [Online]. (accessed 2015, January 14). Available: http://www.3dpanelmethod.com/features.html

[222] SALOME Contributors. *SALOME*. [Online]. (accessed 2015, January 5). Available: http://www.salome-platform.org/

[223] SALOME Contributors. *About SALOME*. [Online]. (accessed 2015, January 5). Available: http://www.salome-platform.org/user-section/about

[224] enGrid Contributors. *enGrid - open-source mesh generation*. [Online]. (accessed 2015, January 5). Available: http://engits.eu/en/engrid

[225] David Eller. *Larosterna: about*. [Online]. (accessed 2015, January 6). Available: http://www.larosterna.com/index.html

[226] David Eller. *Larosterna: aircraft modeling & mesh generation*. [Online]. (accessed 2015, January 6). Available: http://www.larosterna.com/sumo.html

[227] David Eller. *Larosterna: visualization for aeroelasticity*. [Online]. (accessed 2015, January 6). Available: http://www.larosterna.com/scope.html

[228] Imanol Garcia and William Tougeron. *Read Group of Faces TUI from GUI*. [Online]. (accessed 2015, January 6). Available: http://www.salome-platform.org/forum/forum_10/83373997

[229] Martin Hepperle. (2007, January 27). *JavaFoil — Analysis of Airfoils*. [Online]. Available: http://www.mh-aerotools.de/airfoils/javafoil.htm

[230] Martin Hepperle. (2006, August 29). *The Boundary Layer Method*. [Online]. Available: http://www.mh-aerotools.de/airfoils/jf_analysis_boundarylayer.htm

[231] Martin Hepperle. (2008, February 16). *Users Manual*. [Online]. Available: http://www.mh-aerotools.de/airfoils/jf_users_manual.htm

[232] Martin Hepperle. (2003, September 8). *JavaProp - Design and Analysis of Propellers*. [Online]. Available: http://www.mh-aerotools.de/airfoils/javaprop.htm

[233] Martin Hepperle. (2003, September 8). *Design of a Propeller*. [Online]. Available: http://www.mh-aerotools.de/airfoils/jp_propeller_design.htm

[234] Charles N. Adkins and Robert H. Liebeck. "Design of optimum propellers," *Journal of Propulsion and Power*, vol. 10, no. 5, pp. 676-682, 1994.

[235] Martin Hepperle. (2006, December 24). *Analysis of a Propeller*. [Online]. Available: http://www.mh-aerotools.de/airfoils/jp_propeller_analysis.htm

[236] Martin Hepperle. (2008, February 16). *A Validation Exercise*. [Online]. Available: http://www.mh-aerotools.de/airfoils/jp_validation.htm

[237] Martin Hepperle. (2008, February 16). *Users Manual*. [Online]. Available: http://www.mh-aerotools.de/airfoils/jp_users_manual.htm

[238] Tomasz Goetzendorf-Grabowski. (2014, June 1). *PANUKL 2012*. [Online]. Available: http://www.meil.pw.edu.pl/add/ADD/Teaching/Software/PANUKL

[239] Tomasz Goetzendorf-Grabowski. (2013, February 11). *Users Manual for PANUKL: Version ENGv1*. [Online]. Available: http://itlims.meil.pw.edu.pl/zsis/pomoce/PANUKL/2012/PanuklMan_eng.pdf

[240] T. Goetzendorf-Grabowski, J. Mieloszyk, and D. Mieszalski, "MADO - software package for high order multidisciplinary aircraft design and optimization," *28th Congress of the International Council of the Aeronautical Sciences, ICAS 2012*, Brisbane, Australia, September 23-28, 2012, pp. 481-490.

[241] Tomasz Goetzendorf-Grabowski. (2012, February 7). *SDSA – Theoretical Basics*. [Online]. Available inside package: http://itlims.meil.pw.edu.pl/zsis/pomoce/SDSA/2015/SDSA_Setup.zip

[242] Joël Cugnoni. (2005, October 14). *Welcome to CAELinux*. [Online]. Available: http://www.caelinux.com/CMS/index.php?option=com_content&view=article&id=12:welcome-to-caelinux&catid=1:news&Itemid=29

[243] Joël Cugnoni. (2014, March 9). *New release: CAELinux 2013*. [Online]. Available: http://www.caelinux.com/CMS/index.php?option=com_content&view=article&id=56:new-release-caelinux-2013&catid=1:news&Itemid=29

[244] Jarret M. Lafleur. "Derivation and Application of a Method for First-Order Estimation of Planetary Aerial Vehicle Power Requirements". [Online]. Available: http://solarsystem.nasa.gov/docs/7_16LAFLEUR_paper.pdf