engineering proceedings

MDPI

Proceeding Paper

Efficient Unmanned Aerial Vehicle Design: Automated Computational Fluid Dynamics Preprocessing from Geometry to Simulation †

Chris Pliakos [1,2], Giorgos Efrem [1,2], Thomas Dimopoulos [1,2] and Pericles Panagiotou [1,2,*]

[1] Laboratory of Fluid Mechanics and Turbomachinery, Department of Mechanical Engineering, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece; pliakosc@auth.gr (C.P.); egiorgosm@auth.gr (G.E.); thomasdn@auth.gr (T.D.)
[2] UAV Integrated Research Center (UAV-iRC), Center for Interdisciplinary Research and Innovation (CIRI), Aristotle University of Thessaloniki, 57001 Thessaloniki, Greece
* Correspondence: peripan@auth.gr
† Presented at the 14th EASN International Conference on "Innovation in Aviation & Space towards sustainability today & tomorrow", Thessaloniki, Greece, 8–11 October 2024.

Abstract: Current trends in the aerospace and UAV sectors emphasize integrating Artificial Intelligence (AI) technologies into the design process. AI technologies necessitate extensive data to capture the non-linearities in fluid phenomena. To address these needs, this work focuses on automating the data aggregation process for fixed-wing platforms, ranging from Micro–Mini to HALE-Strike UAVs, as classified by NATO. Specifically, this paper presents a framework for automating the tedious tasks required for geometry generation, mesh generation, and solution setup in a commercial Computational Fluid Dynamics (CFD) solver, for any arbitrary wing within the aforementioned design space. By combining various well-established open-source suites and commercial software via Python scripting, the preprocessing steps up to the solution require only a few minutes on a typical laptop workspace. Despite the rapid geometry acquisition, mesh generation, and solution setup through the pipeline, the guidelines and common practices for subsonic external flow simulations are still strictly followed. This results in solutions with a deviation of merely sub 5% from those of an experienced designer, even for the extremes of the flight envelope. The proposed framework significantly reduces design iteration times, enabling more efficient and innovative UAV development. Additionally, the framework's ability to accumulate high-quality data for machine learning enhances predictive modeling and optimization capabilities across UAV design practices.

Keywords: CFD automation; UAV design; BWB; machine learning; Fluent; Python

Academic Editors: Spiros Pantelakis, Andreas Strohmayer and Nikolaos Michailidis

Published: 14 March 2025

Citation: Pliakos, C.; Efrem, G.; Dimopoulos, T.; Panagiotou, P. Efficient Unmanned Aerial Vehicle Design: Automated Computational Fluid Dynamics Preprocessing from Geometry to Simulation. *Eng. Proc.* **2025**, *90*, 52. https://doi.org/10.3390/engproc2025090052

1. Introduction

Unmanned Aerial Vehicles (UAVs) have gained significant traction in various applications, ranging from surveillance and reconnaissance to cargo transport and environmental monitoring [1,2]. As UAVs grow in popularity and their operational requirements become increasingly complex, the aerospace sector is focusing on streamlining their design processes to achieve optimized performance in less time. Automation and machine learning (ML) are becoming essential components of these efforts, particularly in the context of Computational Fluid Dynamics (CFD), where the efficient preprocessing of geometry and mesh plays a crucial role in shortening the development cycle.

CFD analyses are fundamental for assessing aerodynamic properties and optimizing UAV designs, but traditional workflows involve labor-intensive mesh generation and simulation setups, often requiring expert intervention. The introduction of automated preprocessing frameworks addresses these challenges by enabling consistency, accuracy, and significantly faster turnaround times [3,4]. High-fidelity methods, such as Reynolds-Averaged Navier–Stokes (RANS) simulations, require detailed meshes that can be laborious to generate and validate, adding significant overhead to the design process.

Recent advancements in computational power, particularly through the use of Graphics Processing Units (GPUs), have enabled accelerated CFD analysis, making high-quality simulation an option for rapid optimization and data generation. Despite these advancements, maintaining accuracy while reducing manual input remains a significant challenge in developing robust preprocessing workflows. Relevant studies have demonstrated the need for combining automated geometry generation with reliable mesh processing to facilitate general applicability across a wide range of UAV configurations.

This study aims to fill this gap by presenting a framework for automating the preprocessing and CFD simulation of UAV wing designs. By integrating various well-established open-source and commercial tools through Python scripting, the proposed method ensures rapid, consistent, and high-quality CFD analysis for a broad UAV design space, from Micro–Mini to high-altitude long-endurance (HALE) platforms. Additionally, this study aims to create a framework that will be utilized for generating data for machine learning (ML) applications.

2. Design Space

The goal of this framework is to be able to provide reliable automated CFD results for a wide variety of UAVs. More specifically, the generated wing geometries and their simulation conditions were influenced by a wide range of UAVs and their operating conditions. The design space consisted of UAVs from the Micro category (<2 kg MTOW), all the way up to MALE and HALE-Strike (>600 kg MTOW), as classified by NATO and presented in [5]. A quick summary of the UAV categories, alongside their respective operating condition ranges, are presented in Table 1.

Table 1. Operating conditions for UAV classifications according to NATO.

Class	Category	GTOW [kg]	Altitude [ft]	Cruise Speed Range [km/h]	Reynolds Number Range	Flow Regime
Class I	Micro	<2	<200	30–100	10^4–10^6	Laminar to transitional
	Mini	2–15	<3000	30–100		
	Small	15–150	<5000	50–150		
Class II	Tactical	150–600	<18,000	100–400	10^6–10^7	Predominantly turbulent
Class III	MALE	>600	<45,000	200–300	>10^7	Fully turbulent
	HALE	>600	<65,000	400–800		
	Strike	>600	<65,000	400–800		

To enable fully autonomous geometry generation without human intervention, the wing planform must be parameterized in such way as to account for the degrees of freedom necessary for a designer during the conceptual and preliminary stages of UAV design. Since the design space of the framework covers both BWB-type UAVs and conventional ones, it is necessary to use at least two spanwise wing sections, thus enabling the generation of the “kink” airfoil or otherwise “cranked” wings.

The selected design variables are depicted in Figure 1, and their corresponding limits are showcased in Table 2. These low-level parameters are selected as the minimum number of variables required to completely describe the wing planform, while also keeping the interpretability high for the design team.

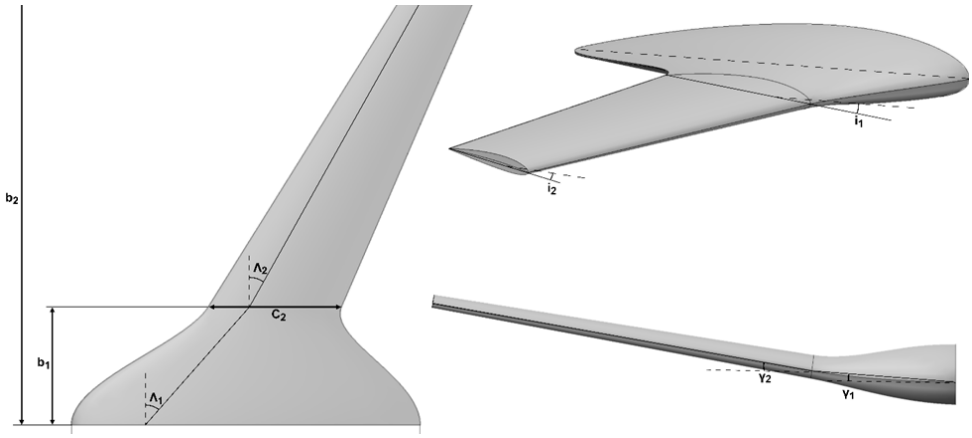


Figure 1. Design variables used to parameterize the wing planform.

Table 2. Design variables used to parameterize the wing planform and their respective ranges.

Planform Parameter	Description	Units	Limits
c_1	Centerline chord length	m	$c_1 \in [0.2,\ 10]$
c_2	Root chord length	m	$c_2 \in [0.3 \times c_1,\ c_1]$
c_3	Tip chord length	m	$c_3 \in [0.3 \times c_2,\ c_2]$
b_1	c_2 spanwise location	m	$b_1 \in [0.1 \times b_2,\ 0.9 \times b_2]$
b_2	c_3 spanwise location (semispan)	m	$b_2 \in [1,\ 10]$
Λ_1	Sweep angle of section 1 @ $c/4$	deg	$\Lambda_1 \in [0,\ 60]$
Λ_2	Sweep angle of section 2 @ $c/4$	deg	$\Lambda_2 \in [0,\ 60]$
i_1	Twist angle of c_2 @ $c/4$	deg	$i_1 \in [-5,\ 5]$
i_2	Twist angle of c_3 @ $c/4$	deg	$i_2 \in [-7,\ 7]$
γ_1	Dihedral angle of section 1	deg	$\gamma_1 \in [-10,\ 10]$
γ_2	Dihedral angle of section 2	deg	$\gamma_2 \in [-10,\ 10]$

3. Software Packages/Tools

Streamlining the software suites with the target of seamless geometry modeling, mesh generation, and, finally, a complete CFD analysis, required a high degree of controllability of the available tools. Thus, Python 3.11 was equipped, since every chosen suite provided a Python API with almost complete authority on the underlying features. In this section, the three main software packages will be showcased, describing how they are utilized, as well as which procedure is followed to align their output as an input for the next module in the pipeline.

3.1. Vehicle Sketch Pad—OpenVSP

OpenVSP (Open Vehicle Sketch Pad) is an open-source parametric aircraft geometry modeling tool developed by NASA, which is widely used in aerospace engineering for conceptual design and analysis [6]. It enables users to create detailed, parametric representations of aircraft geometries that can be readily exported to various formats for further aerodynamic and structural analysis. OpenVSP also provides a robust API, allowing its seamless integration into custom workflows and enabling the automation of geometry generation, modification, and data extraction processes, which is particularly advantageous

for research and optimization tasks [7]. In this study, OpenVSP 3.40.1 was used due to its compatibility with the selected Python version.

Beyond geometry generation, OpenVSP supports advanced modeling tasks, including trailing edge (TE) treatments and surface blending, which are essential for aerodynamic fidelity and meshing automation. For the TE designs, straight cuts were chosen over rounded edges to prioritize robustness in feature recognition by the meshing pre-processors. OpenVSP allows for the precise parameterization of TE cut thicknesses, which were set to 1% of the chord length in accordance with the CFD guidelines [8]. This process was automated using OpenVSP's API, ensuring its consistent application across airfoil sections while preserving aerodynamic characteristics. Blending between wing sections is another critical operation performed seamlessly through OpenVSP's API. By applying universal blending rules, sharp transitions are avoided without adding dimensionality to the design space, as depicted in Figure 2. Adjustments ensure tangency at key points and symmetry at the centerline, using a consistent "blending strength" for the smooth integration of sections. This automation enables the efficient modeling of complex geometries, such as Blended Wing Body (BWB) configurations, while maintaining aerodynamic consistency across varying design spaces.

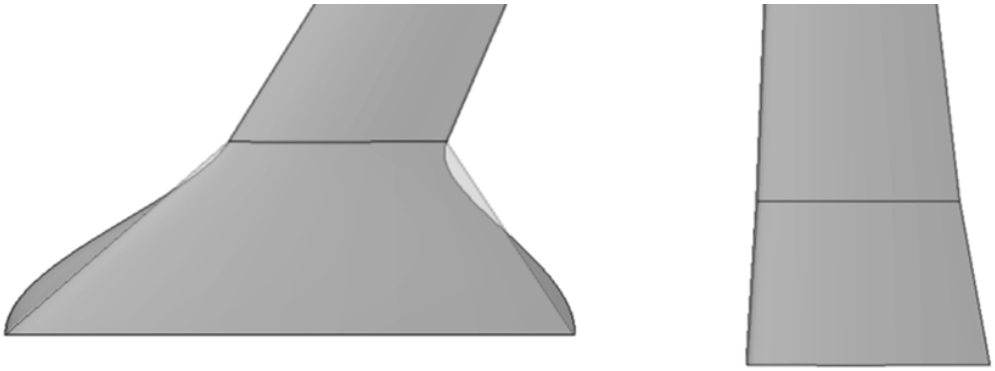


Figure 2. The before and after blending paradigm for the BWB (**right**) and conventional wing (**left**).

The final output from OpenVSP is a STEP file, which serves as a high-fidelity representation of the modeled geometry. This format is well suited for compatibility with downstream meshing software, facilitating the creation of computational grids for aerodynamic and structural analyses.

3.2. ANSA

ANSA, developed by BETA CAE Systems, is an advanced preprocessing tool widely used in the engineering field for preparing models for Finite Element Analysis (FEA) and CFD. It offers robust tools for geometry cleanup, meshing, and simulation setup, ensuring high-quality and efficient model preparation. ANSA also features a powerful Python API, enabling the automation of repetitive tasks and providing complete control over mesh parameters, the customization of workflows, and effortless integration with other software tools, making it particularly valuable for handling complex geometries and large-scale simulations. In this study, v24.1.0 (Root, Switzerland) was utilized.

Surface meshing is a critical step in the CFD preprocessing workflow, directly influencing simulation accuracy and stability [9]. Using ANSA's CFD meshing algorithm, an unstructured curvilinear surface mesh is generated, consisting of first-order quadrilateral elements, with triangular elements applied in areas where quadrilaterals cannot form. Mesh sizing and curvature refinement parameters are globally defined to ensure smooth transitions and maintain fidelity in curved regions [9]. The leading edge (LE) is treated with an anisotropic quad surface mesh, featuring four constant-height rows near the stagnation

zone to accurately capture steep gradients. Similarly, the trailing edge (TE) is addressed using a structured, anisotropic grid with four rows of elements, where the initial height is set to one-quarter of the TE cut thickness, ensuring smooth transitions across the surfaces [Figure 3]. These tailored treatments enhance the surface mesh quality, ensuring the numerical stability and precise resolution of aerodynamic features, particularly in areas with high curvature or sharp geometric transitions [9,10].

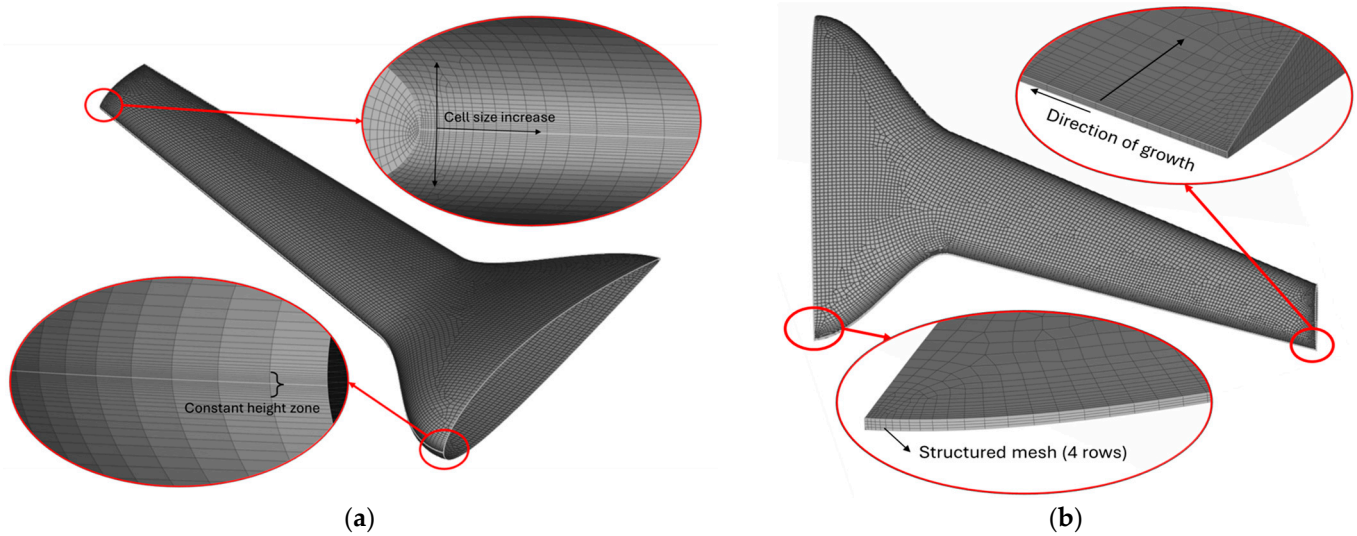


Figure 3. Special mesh treatment: (**a**): leading edge and (**b**): trailing edge.

Accurately resolving the boundary layer near aerodynamic surfaces is essential for capturing critical flow phenomena such as skin friction, separation points, and laminar-to-turbulent transition. A non-dimensional wall distance (y+) of approximately 1 is targeted to ensure the first inflation layer cell lies within the viscous sublayer, enabling precise turbulence modeling in Reynolds-Averaged Navier–Stokes (RANS) simulations. This approach ensures the accurate prediction of aerodynamic performance metrics like Lift and Drag [8]. The first layer height is calculated based on freestream flow properties, surface shear velocity, and fluid viscosity, ensuring the correct resolution of steep velocity gradients near walls.

The volume mesh is composed of a structured hexahedral grid (inflation layer and additional layers) near the wall to capture near-wall effects, while the far-field regions are meshed with an unstructured hexahedral grid. Best practices are followed for control volume placement, with upstream boundaries at 10 chord lengths, downstream boundaries at 25 chord lengths, and lateral boundaries at 10 chord lengths, minimizing boundary effects while maintaining computational efficiency [8,9,11]. Also, trapezoidal-shaped bodies of influence are constructed around the wing, providing the ability to utilize the same grid for multiple angles of attack. This hybrid grid approach ensures high accuracy and robust simulation results.

The aforementioned meshing practices are fully parametrized with the geometric design variables and simulation conditions, aiming to achieve the "fully autonomous" goal of this framework. Finally, the output of the meshing module is an *.msh.h5* mesh file compatible with the ANSYS software suite [v2023-R2].

3.3. PyFluent

ANSYS Fluent is a powerful CFD software widely used for simulating fluid dynamics [12]. PyFluent, the Python API for Fluent, enables users to automate tasks such as setup, execution, and the postprocessing of simulations. This API allows for the scripting

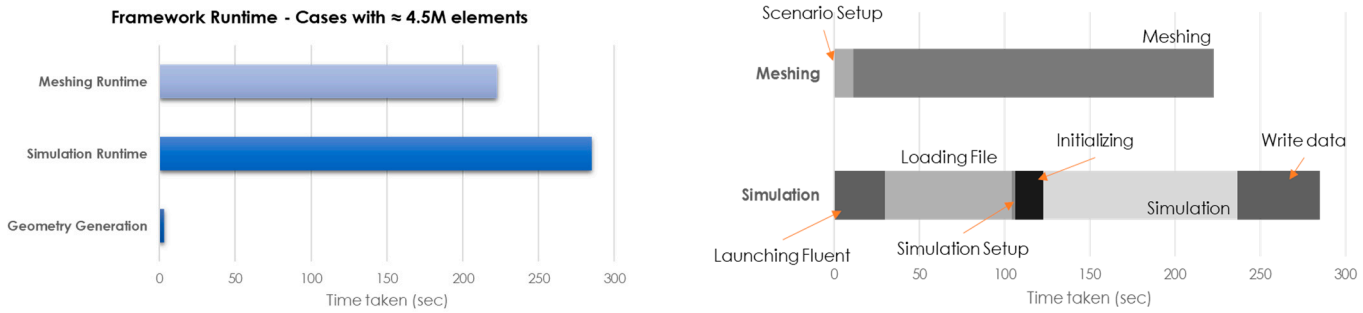
of workflows, including geometry import, solver configuration, and results extraction. Through the integration of PyFluent, UAV wings can be evaluated rapidly and consistently, minimizing manual effort and enabling efficient design validation within the automated preprocessing framework. In this study, ANSYS Fluent 2023R2 and PyFluent version 0.21 were used, ensuring compatibility and leveraging the latest features for streamlined simulation workflows. Additionally, the GPU native solution offered by Fluent was utilized to accelerate the analyses, significantly reducing computation time and enhancing the overall efficiency of the simulation process [13].

The solution setup involved selecting an appropriate turbulence model, assigning boundary conditions, and monitoring the convergence criteria to ensure simulation accuracy. The turbulence model selected for this study was the k-omega SST, which is well suited for capturing both boundary layer separation and flow transition phenomena. The turbulence viscosity ratio was assigned according to [14], and scaled with the Reynolds number to ensure the appropriate modeling of turbulence effects across varying flow regimes. Residuals were monitored throughout the simulation to track convergence, with specific attention paid to Lift and Drag forces over a moving window of 100 iterations. Convergence was declared when both Lift and Drag changed by less than 0.1%, or a periodic behavior was detected, even if this led to residuals higher than 0.1%.

The simulation module provided all the required outputs to the user in multiple files of the *.txt* format, comprising the forces exerted on the wing, the residuals from the partial differential equations, and information about convergence. Simultaneously, an *.xml* file was constructed containing concisely the key elements of the simulation (converged forces, iterations, y^+ values) assisting in rapid decision making and/or the coupling of an optimizer.

4. Analysis Efficiency and Validation

This section presents the performance assessment of the automated CFD framework in terms of simulation efficiency and accuracy. The geometry generation time is minimal compared to meshing and simulation [Figure 4a]. The mesh is generated only once per wing and can be reused for multiple angles of attack (AoAs), increasing the overall efficiency. Additionally, the usage of GPUs has significantly lowered the computational cost of the simulations, reducing time, and making high-fidelity aerodynamic analysis more feasible.



(**a**) (**b**)

Figure 4. Computational time taken: (**a**): complete framework and (**b**): breakdown of components.

The accuracy of the CFD framework was evaluated using the ONERA M6 wing at its operational conditions, a well-known benchmark geometry chosen for its simplicity and established aerodynamic characteristics [15]. The specialist, a CFD mechanical engineer with over five years of experience, manually prepared the simulation (both mesh and setup) for comparison. The results show that the drag polar predicted by the automated framework is within 3% of the expert's values, demonstrating high reliability [Figure 5].

This agreement highlights the framework's capability to match expert-level precision while reducing manual effort.

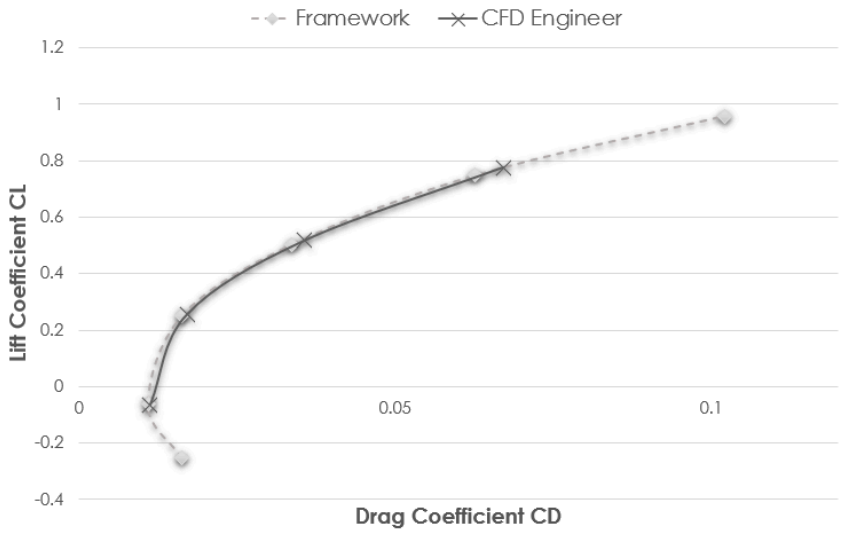


Figure 5. Drag polar comparison of ONERA M6 between the framework and a CFD engineer.

5. Conclusions

The presented approach demonstrates the significant potential for automating the preprocessing and execution of CFD analyses for UAV wings, providing considerable time savings compared to traditional manual processes. By leveraging Python scripting and GPU acceleration, this automated workflow minimizes human intervention while ensuring consistent quality in the generated meshes and simulations. The results obtained have proven to be both reliable and robust, aligning closely with those produced by experienced CFD engineers. Furthermore, adherence to common aerodynamic guidelines ensures that the methodology is applicable to a wide range of wing geometries, covering diverse configurations within the UAV design space. This generalizability makes the proposed framework a practical and scalable solution for future aerodynamic studies, streamlining the process of Computational Fluid Dynamics analysis in aerospace design.

Author Contributions: Conceptualization, C.P. and P.P.; methodology, C.P.; software, G.E. and C.P.; validation, C.P., G.E., T.D. and P.P.; investigation, G.E.; resources, C.P.; data curation, C.P. and G.E.; writing—original draft preparation, C.P. and G.E.; writing—review and editing, T.D. and P.P.; visualization, G.E. and C.P.; supervision, P.P.; funding acquisition, P.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research work was supported by the Hellenic Foundation for Research and Innovation (HFRI) under the Basic Research Financing (Horizontal support for all Sciences), National Recovery and Resilience Plan (Project Number: 016429, Project Acronym: INDIANA).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are available on request due to restrictions related to the details of the baseline platform. The data presented in this study are available on request from the corresponding author.

Acknowledgments: The results presented in this work have been produced using the Aristotle University of Thessaloniki (AUTh) High Performance Computing Infrastructure and Resources.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Karageorgiou, A.; Kantouris, P.; Metallidou, N.; Charizani, A.; Panagiotou, P. Design, Manufacturing and Flight-Testing of a Fixed-Wing, Small-Scale UAV for the Transportation of Blood Bags to Remote Locations. *J. Phys. Conf. Ser.* **2023**, *2526*, 012087. [CrossRef]
2. Dimitriou, S.; Kapsalis, S.; Dimopoulos, T.; Mitridis, D.; Pouchias, K.; Panagiotou, P.; Yakinthos, K. Preliminary Design of a BWB UAV for Highway Traffic Monitoring. *IOP Conf. Ser. Mater Sci. Eng.* **2022**, *1226*, 012010. [CrossRef]
3. Zhang, M.; Gong, J.; Axner, L.; Barth, M. Automation of High-Fidelity CFD Analysis for Aircraft Design and Optimization Aided by HPC. In Proceedings of the 2020 28th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), Västerås, Sweden, 11–13 March 2020.
4. Papkov, V.; Shadymov, N.; Pashchenko, D. CFD-Modeling of Fluid Flow in Ansys Fluent Using Python-Based Code for Automation of Repeating Calculations. *Int. J. Mod. Phys. C* **2023**, *34*, 2350114. [CrossRef]
5. Mitridis, D.; Kapsalis, S.; Terzis, D.; Panagiotou, P. An Evaluation of Fixed-Wing Unmanned Aerial Vehicle Trends and Correlations with Respect to NATO Classification, Region, EIS Date and Operational Specifications. *Aerospace* **2023**, *10*, 382. [CrossRef]
6. Hahn, A.S. Vehicle Sketch Pad: A Parametric Geometry Modeler for Conceptual Aircraft Design. In Proceedings of the 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, Orlando, FL, USA, 4–7 January 2010.
7. McDonald, R.A. Advanced Modeling in Open VSP. In Proceedings of the 16th AIAA Aviation Technology, Integration, and Operations Conference, Washington, DC, USA, 13–17 June 2016; American Institute of Aeronautics and Astronautics Inc., AIAA: Reston, VA, USA, 2016.
8. Goetten, F.; Finger, D.F.; Marino, M.; Bil, C.; Havermann, M.; Braun, C. A Review of Guidelines and Best Practices for Subsonic Aerodynamic Simulations Using RANS CFD. In Proceedings of the Asia Pacific International Symposium On Aerospace Technology, Engineers Australia, Gold Coast, Australia, 4 December 2019.
9. Hirsch, C. *Numerical Computation of Internal and External Flows Fundamentals of Computational Fluid Dynamics*; Elsevier: Amsterdam, The Netherlands, 2007.
10. Wendt, J.F. (Ed.) *Computational Fluid Dynamics*; Scholars Portal, 2019; Springer: Berlin/Heidelberg, Germany, 2019; ISBN 9783540850557.
11. Menter, F.; Hüppe, A.; Matyushenko, A.; Kolmogorov, D. An Overview of Hybrid Rans–Les Models Developed for Industrial Cfd. *Appl. Sci.* **2021**, *11*, 2459. [CrossRef]
12. *Ansys Fluent User's Guide*; Ansys Inc.: Canonsburg, PA, USA, 2023.
13. ANSYS. *Unleashing the Full Power of GPUs for Ansys Fluent, Part 2*; Ansys Inc.: Canonsburg, PA, USA, 2022.
14. Rumsey, C.L.; Spalart, P.R. Turbulence Model Behavior in Low Reynolds Number Regions of Aerodynamic Flowfields. *AIAA J.* **2009**, *47*, 982–993. [CrossRef]
15. Mayeur, J.; Dumont, A.; Destarac, D.; Gleize, V. RANS Simulations on TMR Test Cases and M6 Wing with the Onera ElsA Flow Solver. In Proceedings of the 53rd AIAA Aerospace Sciences Meeting, Kissimmee, FL, USA, 5–9 January 2015; American Institute of Aeronautics and Astronautics Inc., AIAA: Reston, VA, USA, 2015.