Validation of an Aeroelastic Numerical Tool for Rotor Performance in Hover

Alessandro Cocco
Politecnico di Milano, Dipartimento di Scienze e Tecnologie Aerospaziali
Milan, Italy

ABSTRACT

Hover prediction is a critical task in the helicopter's design phases. In this study, we propose a coupled framework utilizing DUST-MBDyn, which employs vortex particle methods. The method generally outperforms traditional CFD by avoiding the mesh generation and significantly reducing computational overhead. The framework is validated against the NASA hover testbed experiment by comparing the numerical predictions with experimental data. The results demonstrate that the framework not only overcomes the limitations of traditional computational fluid dynamics (CFD), but also offers a computationally efficient and reliable solution for integration into optimization workflows.

INTRODUCTION

Conventional rotary-wing aircraft, such as helicopters and tiltrotors, are designed to operate across a range of flight conditions, each demanding distinct performance objectives. These include efficient cruising, stable hovering, and high-performance maneuvering. Accurate numerical predictions of performance in these regimes are critical for both the development of new aircraft and the assessment of existing designs. Hovering performance, in particular, imposes a critical design constraint due to its high power demands. This challenge is amplified in Urban Air Mobility (UAM) vehicles, which often employ multiple rotors and unconventional configurations. For such designs, precise hover performance predictions are essential, as underestimating power requirements can severely compromise vehicle capabilities—especially for electrically powered systems reliant on energy-dense batteries (Ref. 1).

Despite significant advancements in computational fluid dynamics (CFD), current methodologies face persistent challenges in modeling fluid–structure interactions (FSI) for rotors in hover (Ref. 2). A key limitation lies in the labor-intensive process of generating high-quality meshes required to resolve intricate flow phenomena around rotor blades. Furthermore, the high computational cost of these simulations renders them impractical for iterative design processes and optimization loops, where rapid evaluations are paramount.

To address these limitations, we propose a coupled framework utilizing DUST-MBDyn (Ref. 3). Unlike conventional CFD approaches, the aerodynamic solver DUST (Ref. 4) leverages vortex particle methods, bypassing traditional mesh generation and significantly reducing computational overhead. This is complemented by MBDyn (Ref. 5), an open-source multibody dynamics solver that efficiently resolves complex structural interactions.

This study evaluates the validity of the coupled DUST-MBDyn tool by comparing its predictions with experimental data from the NASA hover testbed (Ref. 6). Through this validation, we demonstrate that the framework not only overcomes the limitations of traditional CFD but also offers a computationally efficient and reliable solution for integration into optimization workflows.

The work is organized as follows:

1. The software tools are presented, namely the coupling between MBDyn and DUST focusin in particular on the the latest improvements of the DUST code are briefly introduced.

2. The lifting line and wake models are verified against the analytic results of the elliptic wing.

3. The HAVB rotor is described.

4. The dynamic stability, the aerodynamic performances of the rotor, and blade deformation are compared against the experimental results.

METHODS

To obtain an aeroelastic solver with a good balance between the quality of the results and the computational cost, the coupling between DUST (Ref. 4) and MBDyn (Ref. 5) is exploited.

MBDyn[1] is a free general-purpose multibody solver. MBDyn formulate and solves the equations of motion of a system of entities possessing degrees of freedom (nodes) connected through algebraic constraints and subjected to internal and external loads. The nodes that describe the kinematics of the

[1] https://public.gitlab.polimi.it/DAER/mbdyn

structural problem can be connected by elastic/viscoelastic internal forces, namely lumped structural components (Ref. 7) and beams (Ref. 8), shells (Ref. 9), and Component Mode Synthesis (CMS) elements (Ref. 10)), with a variety of viscoelastic constitutive laws or by kinematic constraints. Simple aerodynamics can be modeled by built-in elements that exploit the 2D strip theory model by look-up tables of the aerodynamic coefficients and classical rotor inflow models based on momentum theory.

DUST[2] is a free and open-source code for aerodynamic analysis, licensed under MIT. It can handle different levels of fidelity, such as thick or thin surfaces, vortex lattices, or lifting line elements. It also has a vortex particle method for wake modeling that can capture the free-vorticity flow field, which is useful for simulating configurations with strong aerodynamic interactions. Recent enhancements implement a CPU-GPU coupled architecture, significantly improving computational performance. Implementation details of this hybrid framework are discussed in the following section.

Communications between the two solvers are managed by preCICE (Precise Code Interaction Coupling Environment) (Ref. 11), a coupling library for partitioned multiphysics simulations, originally developed for fluid-structure interaction and conjugate heat transfer simulations. The interface between structural and aerodynamic grids is obtained as a weighted average of the distance between the nodes of the two grids and is used for motion interpolation and consistent force and moment reduction. The details about the coupling between DUST and MBDyn, with the relative validations, are described in (Ref. 3), in which the flutter speed of the Goland's wing (Ref. 12) has been verified against other analytical tools.

DUST Wake Formulation

The original `DUST` implementation is based on the classical Vortex Particle Method (VPM), which assumes a constant vortex core size over time, i.e., $\dot{\sigma} = 0$. In contrast, `FlowVPM` (Ref. 13) introduces a reformulated VPM (rVPM) that ensures both mass conservation within vortex tubes and angular momentum conservation for spherical vortex particles. This is achieved by modifying the evolution equation of the particle strength vector to include a correction term in the vortex stretching component, and by introducing a differential equation that governs the temporal evolution of the core radius σ.

Alvarez and Ning propose a generalized formulation of the VPM governing equations using two parameters, f and g. These parameters allow transitioning between different formulations: the classical VPM (cVPM) is recovered with $f = g = 0$, while the reformulated version corresponds to $f = 0$, $g = \frac{1}{5}$. The system of ordinary differential equations for the rVPM is given by:

$$\begin{aligned}
&\frac{d}{dt}\boldsymbol{x}^p = \boldsymbol{u}_\sigma^h(\boldsymbol{x}^p) \\
&\frac{d}{dt}\boldsymbol{\alpha}^p = (\boldsymbol{\alpha}^p \cdot \nabla)\boldsymbol{u}_\sigma^h(\boldsymbol{x}^p) - \frac{g+f}{\frac{1}{3}+f}\left[(\boldsymbol{\alpha}^p \cdot \nabla)\boldsymbol{u}_\sigma^h(\boldsymbol{x}^p)\cdot\hat{\boldsymbol{\alpha}}^p\right]\hat{\boldsymbol{\alpha}}^p + \\
&\qquad + 105\nu \frac{\sigma^4}{(\|\boldsymbol{x}^p - \boldsymbol{x}^q\|^2 + \sigma^2)^{\frac{9}{2}}}\left(vol^p\boldsymbol{\alpha}^q - vol^q\boldsymbol{\alpha}^p\right) \\
&\frac{d}{dt}\sigma^p = -\frac{g+f}{1+3f}\frac{\sigma^p}{\|\boldsymbol{\alpha}^p\|}\left[(\boldsymbol{\alpha}^p \cdot \nabla)\boldsymbol{u}_\sigma^h(\boldsymbol{x}^p)\right]\cdot\hat{\boldsymbol{\alpha}}^p
\end{aligned}$$

The evolution of σ^p relies solely on the vortex stretching term, which is already computed in standard VPM procedures; thus, it does not introduce any significant computational overhead. The modified evolution equation for $\boldsymbol{\alpha}^p$ introduces an additional correction term acting only along the direction of $\boldsymbol{\alpha}^p$, leaving the turning (rotational) components unaltered compared to the classical formulation. Notably, in the rVPM case ($f = 0$, $g = \frac{1}{5}$), the component of $\frac{d}{dt}\boldsymbol{\alpha}^p$ aligned with $\boldsymbol{\alpha}^p$ is reduced by a factor of 3/5 relative to cVPM. This results in a more gradual change in particle magnitude over time, enhancing numerical stability.

Unlike `FlowVPM`, which models viscosity via the Core Spreading (CS) method, `DUST` employs the Particle Strength Exchange (PSE) scheme. In this approach, diffusion is modeled by directly exchanging vorticity among particles rather than by altering particle positions or core sizes. The diffusion operator $\nabla^2\boldsymbol{\omega}$ is approximated using an integral formulation:

$$\nabla_\sigma^2\boldsymbol{\omega}(\boldsymbol{x}) = 2\int(\boldsymbol{\omega}(\boldsymbol{y}) - \boldsymbol{\omega}(\boldsymbol{x}))\,\eta_\sigma(\boldsymbol{x}-\boldsymbol{y}),d\boldsymbol{y}, \tag{2}$$

where η_σ is a Gaussian smoothing kernel centered at $\boldsymbol{x}$ with width σ. To account for unresolved subgrid-scale (SGS) turbulence effects in particle-based simulations, a Smagorinsky-type eddy viscosity model can be incorporated into the VPM framework. The model introduces an effective turbulent viscosity ν_t defined as:

$$\nu_t = (C_s\Delta)^2\|\mathbb{S}\| \tag{3}$$

where C_s is the Smagorinsky constant (in this work set equal to 0.11), Δ is the filter width (chosen to be equal to the smoothing radius σ of the inducing particle), and $\mathbb{S}$ is the strain-rate tensor, computed as:

$$\mathbb{S} = \frac{1}{2}\left(\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^T\right). \tag{4}$$

The total viscosity used in the diffusion model (e.g., in the PSE scheme) is then modified to include both contributions:

$$\nu_{\text{eff}} = \nu + \nu_t. \tag{5}$$

The computational bottleneck of Vortex Particle Methods lies in the evaluation of the velocity field $\boldsymbol{u}_\sigma^h(\boldsymbol{x}^p)$ and its gradients, which, in a naive pairwise implementation, scale as $\mathscr{O}(N^2)$ with the number of particles. To mitigate this cost and enable large-scale simulations, `DUST` utilizes the Fast Multipole Method (FMM) to accelerate both the velocity and stretching computations.

[2] https://public.gitlab.polimi.it/DAER/dust

Fast Multipole Method

The Fast Multipole Method (FMM) is employed to accelerate the computation of long-range interactions between vortex particles. The domain is initially discretized into a user-defined number of cubic cells, forming the coarsest level of an octree. Each cell is recursively subdivided into eight smaller cubes, generating a multilevel hierarchy of increasingly finer resolution. This hierarchical structure enables efficient grouping of particles and separation of near- and far-field interactions.

Particles are first sorted into the finest level of the octree. Each cell is then classified as a leaf, branch, or inactive cell based on particle content. Leaf cells handle direct near-field interactions, branch cells accumulate contributions from their children, and inactive cells are excluded from computations.

The FMM proceeds through the following steps:

- P2M (Particle to Multipole): Leaf cells compute a multipole expansion from their contained particles.
- M2M (Multipole to Multipole): Expansions are passed upward from children to parents to form aggregated representations.
- M2L (Multipole to Local): Well-separated cells interact through kernel-based local expansions.
- L2L (Local to Local): Local expansions are propagated downward from parents to children.
- Application: At the leaf level, the velocity and velocity gradient at each particle are evaluated using the local expansions.
- P2P (Particle to Particle): At the end, the direct particle to particle interaction is computed within the particle belonging to the same cell and the neighboring one.

To enhance computational performance, the most intensive operations—M2L (interactions with distant cells) and P2P (direct particle-particle interactions within leaf cells)—are implemented on the GPU. Figure 1 shows the computational time required for each step, comparing the CPU vs. GPU time for 1 million particles. Figure 2 shows the speedup achieved for the M2L and P2P operations as a function of the number of particles. The results are obtained using a workstation equipped with two Intel Xeon Gold 6230R with 26 cores (2.1Ghz) and an NVIDIA Quadro RTX A4000.

The greatest benefit of using GPU capabilities is achieved for P2P operation, as it requires $\mathcal{O}(N^2)$ operations, where N is the number of particles contained in a cell and its neighbors.

ELLIPTIC WING VERIFICATION

As a validation benchmark, for the wake model and the DUST lifting line, the classical elliptic wing test case is considered. The wing has a span $b = 1.6\,\mathrm{m}$, a root chord $c_0 = 0.34\,\mathrm{m}$, and

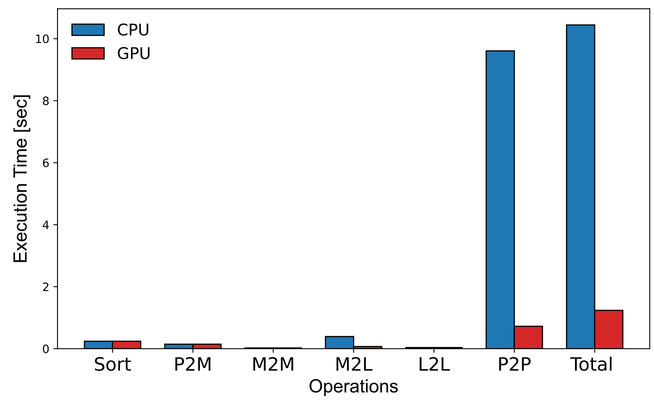

Figure 1: Computational time for each operation for 1M particles

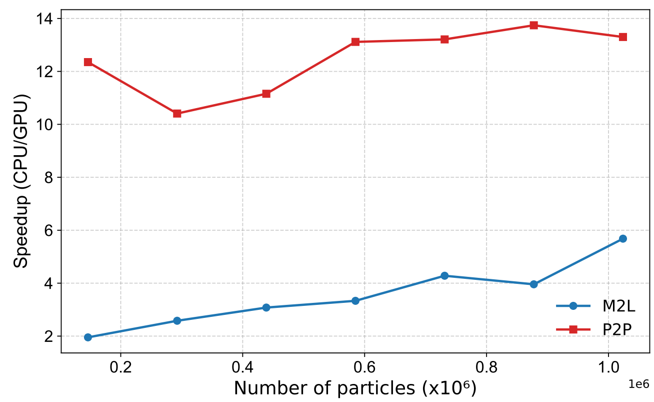

Figure 2: Speedup GPU/CPU for the M2L and P2P operation

a planform area $S = \frac{\pi}{4} c_0 b = 0.427\,\mathrm{m}^2$. The airfoil section is a symmetric NACA0012 with $C_{L/\alpha} = 2\pi$ and neglecting the drag coefficient.

In this configuration, the chord distribution along the span follows the elliptic law:

$$c(y) = c_0 \sqrt{1 - \left(\frac{2y}{b}\right)^2} \qquad (6)$$

The wing is discretized using 80 elements, discretized by using a geometric series with a growth ratio of 0.05. The analytical lift coefficient is given by:

$$C_L = \frac{2\pi\alpha}{\sqrt{1 - Ma^2} + \frac{2}{AR}}, \qquad (7)$$

and the induced drag is:

$$C_D = C_{D_0} + \frac{C_L^2 S}{\pi b^2}. \qquad (8)$$

Figures 3 and 4 show the lift and drag coefficients against the angle of attack. Overall a nearly perfect match is obtained between simulations and analytical results for all angles of attack. A slight underestimation of the drag can be noticed at the extreme of the curve: this can be explained by the difficulties of modeling a rounded tip in the panel method.

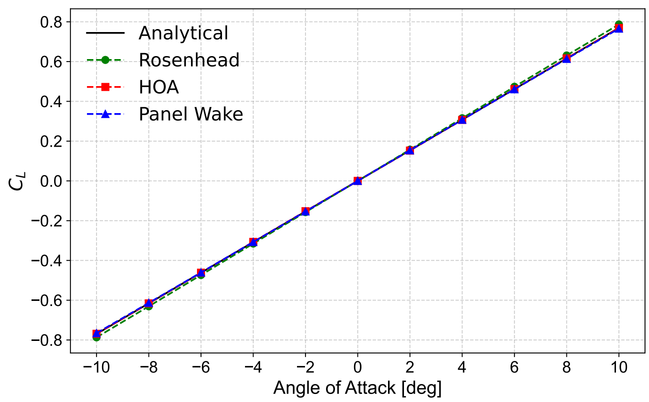

Figure 3: Lift coefficient vs angle of attack for the elliptic wing

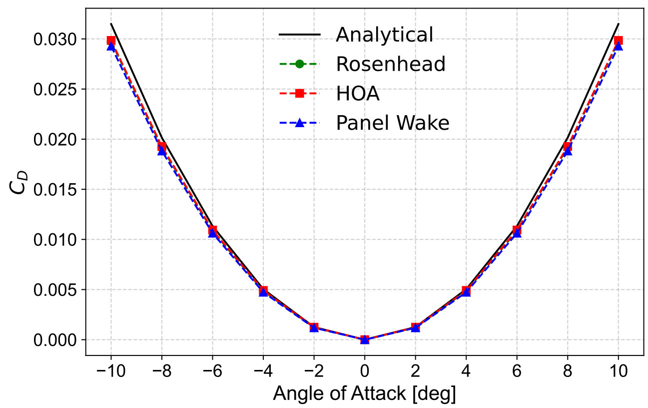

Figure 4: Drag coefficient vs angle of attack for the elliptic wing

ROTOR MODEL DESCRIPTION

The MBDyn multibody model of the Hover Aeroacoustic Validation Baseline (HAVB) articulated rotor comprises a helicopter control system and four flexible blades. Each blade is discretized into 21 three-node beam elements (Refs. 8,14) and attached to the hub via three coincident revolute hinges positioned at 0.05%R (blade radius). These hinges represent the flap, lag, and pitch degrees of freedom. The pitch rotation is restricted by a flexible pitch link that connects the blade root node to the rotating swashplate. To eliminate singularities in the flap and lag hinges, viscous dampers are incorporated into their respective joints.

Stiffness matrix In MBDyn, the beam stiffness matrix is positioned in this reference system:

- The X-axis is along the beam feathering axis and goes from the beam root to the tip.
- The Y-axis coincides with the section chord from the trailing edge to the leading edge.
- The Z-axis is determined by the right-hand rule.
- The origin is the point of intersection between the section plane and the feathering axis.

When considering an isotropic beam section, the constitutive matrix 6×6 can always be written in terms of elementary stiffness and geometrical properties. In isotropic uniform beam sections, the properties can be split into two groups: those related to shear stress and strain, and those related to axial stress and strain. The stiffness properties of the beam are evaluated at two stations: $P_e = \pm\ell/(2\sqrt{3})$ from the mid-node.

The axial properties can be transformed from the elastic reference frame to the section reference frame by means of the following rotor-translation transformation.

$$\mathbf{K}_{ax} = \mathbf{T}_{ax}\mathbf{R}_{ax} \begin{bmatrix} EA_e & 0 & 0 \\ 0 & EJY_e & 0 \\ 0 & 0 & EJZ_e \end{bmatrix} \mathbf{R}_{ax}^T \mathbf{T}_{ax}^T \tag{9}$$

Here, the rotation matrix is given by angle α_e = rotan:

$$\mathbf{R}_{ax} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha_e) & -\sin(\alpha_e) \\ 0 & \sin(\alpha_e) & \cos(\alpha_e) \end{bmatrix} \tag{10}$$

and the translation matrix is:

$$\mathbf{T}_{ax} = \begin{bmatrix} 1 & 0 & 0 \\ z_{na_e} & 1 & 0 \\ -y_{na_e} & 0 & 1 \end{bmatrix} \tag{11}$$

The same roto-translation transformation can also be applied:

$$\mathbf{K}_{sh} = \mathbf{T}_{sh}\mathbf{R}_{sh} \begin{bmatrix} GAY_e & 0 & 0 \\ 0 & GAZ_e & 0 \\ 0 & 0 & GJ_e \end{bmatrix} \mathbf{R}_{sh}^T \mathbf{T}_{sh}^T \tag{12}$$

Any value was provided for the shear stiffness, therefore was assumed to be equal to:

$$GAY = GAZ = GA = \frac{EA}{2(1+\nu)} \qquad \nu = 0.33 \tag{13}$$

The rotation matrix is the same as the axial one: $\mathbf{R}_{sh} = \mathbf{R}_{ax}$, and the translation matrix is

$$\mathbf{T}_{sh} = \begin{bmatrix} 1 & 0 & 0 \\ z_{ct_e} & 1 & 0 \\ -y_{ct_e} & 0 & 1 \end{bmatrix} \tag{14}$$

Finally, the fully coupled stiffness matrix 6×6 can then be obtained as

$$\mathbf{K} = \begin{bmatrix} K_{ax_{11}} & 0 & 0 & 0 & K_{ax_{12}} & K_{ax_{13}} \\ 0 & K_{sh_{11}} & K_{sh_{12}} & K_{sh_{13}} & 0 & 0 \\ 0 & K_{sh_{21}} & K_{sh_{22}} & K_{sh_{23}} & 0 & 0 \\ 0 & K_{sh_{31}} & K_{sh_{32}} & K_{sh_{33}} & 0 & 0 \\ K_{ax_{21}} & 0 & 0 & 0 & K_{ax_{22}} & K_{ax_{23}} \\ K_{ax_{31}} & 0 & 0 & 0 & K_{ax_{32}} & K_{ax_{33}} \end{bmatrix} \tag{15}$$

Mass matrix The inertial properties of each beam are introduced by lumped masses placed at each MBDyn beam node defined in the following reference system:

- The X-axis is along the elastic axis of the beam and goes from the root of the beam to the tip.
- The Y-axis is rotated by an angle with respect to the Y section axis.
- The Z-axis is determined by the right-hand rule.
- The origin is in the center of mass of the body at a distance X_{CG}, Y_{CG}, Z_{CG} from the center of the beam section.

The mass lumping procedure follows:

- Each beam is divided into three segments: the first and third segments have a length of $1/2(1-1/\sqrt{3})\ell_i$, while the second (the mid-length) has a length of $(1\sqrt{3})\ell_i$, where ℓ_i is the beam's length.
- Mass and inertia are obtained by exploiting the trapezoidal integration rule (considering x as the beam axis):

$$M^{i,i+1} = \int_{s^i}^{s^{i+1}} m_{2D}\, dx = \frac{(m_{2D}^{i} + m_{2D}^{i+1})(s^{i+1} - s^{i})}{2} \tag{16}$$

$$I_{xx}^{i,i+1} = \int_{s^i}^{s^{i+1}} I_{xx}^{2D}\, dx \tag{17}$$

$$I_{yy}^{i,i+1} = \int_{s^i}^{s^{i+1}} I_{yy}^{2D}\, dx + \frac{M^{i,i+1}(s^{i+1} - s^{i})^2}{12} \tag{18}$$

$$I_{zz}^{i,i+1} = \int_{s^i}^{s^{i+1}} I_{zz}^{2D}\, dx + \frac{M^{i,i+1}(s^{i+1} - s^{i})^2}{12} \tag{19}$$

where m_{2D}^{i} and $I_{jj}^{2D^i}$ are the density of the section and the moment of sectional inertia around the axis jj evaluated at the station s^i.

- The position of the center of mass of the body is determined as follows.

$$\mathbf{P}_{CG}^{i,i+1} = \frac{m_{2D}^{i+1} \cdot \mathbf{P}_{CG}^{i+1} + m_{2D}^{i} \cdot \mathbf{P}_{CG}^{i}}{m_{2D}^{i} + m_{2D}^{i+1}} \tag{20}$$

DUST aerodynamic model The DUST aerodynamic model discretizes the rotor into 30 uniformly distributed lifting line elements. Four RC-series airfoils are positioned along the span, following the geometry outlined in (Ref. 6). Aerodynamic coefficients for these sections are derived from `.c81` lookup tables, with values between airfoil stations obtained through linear interpolation. The `.c81` tables are generated using NeuralFoil (Ref. 15). NeuralFoil is an open-source tool that provides fast and accurate predictions of airfoil aerodynamic performance. It is based on a machine learning model trained on a large database of XFoil simulations, and it is designed as a drop-in replacement for traditional tools in preliminary design and optimization tasks. The model takes as input the airfoil shape, angle of attack, Reynolds number, and Mach number, and outputs aerodynamic coefficients. Unlike XFoil, NeuralFoil is significantly faster, offering speedups of one to three orders of magnitude, while maintaining high accuracy even in post-stall or transitional flow regimes. Its smooth and differentiable outputs make it especially useful in optimization workflows.

RESULTS AND DISCUSSION

To verify the dynamic stability of the structural model, the rotor's fan plot in vacuo (Fig. 5) is calculated, tracking the frequencies of the first seven rotor modes as a function of rotor speed. At 100% of the RPM, none of the modes cross the resonance frequencies. The first two modes correspond to the rigid lag and rigid flap, respectively, whereas the remaining ones are related to the beam flexible modes. Figure 6 shows the components of the second flap mode. From the mode shape, it is clear that strong pitch flap coupling is present on the blade: when the blade flaps up, then the sections pitch down.

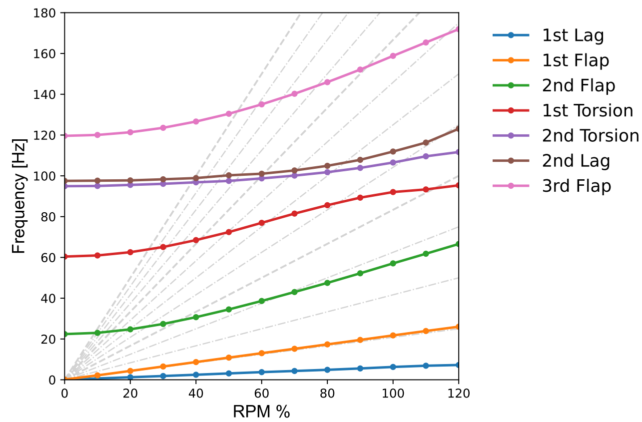

Figure 5: Rotor Fan plot at 0 deg collective

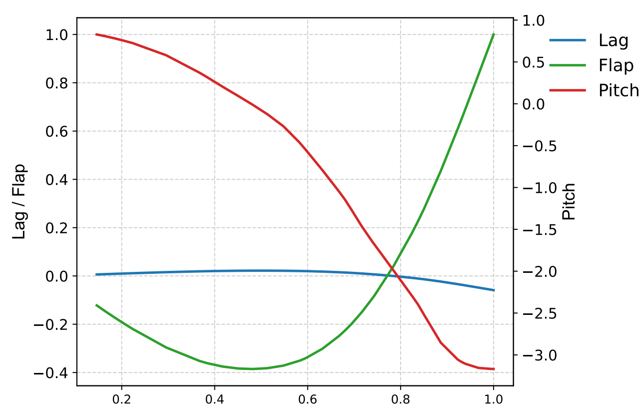

Figure 6: Second flap mode shape

Rotor Validation

Two models were analyzed: the first is fully rigid and uses the DUST aerodynamic model; the second is an aeroelastic model that couples the DUST aerodynamic solver with the MBDyn structural solver. Rotor performance was assessed by comparing thrust and torque predictions from the rigid DUST model against experimental data from NASA's hover testbed (RUN 30), across a collective pitch range of 4° to 13°. For each point, 10 revolutions were used to reach convergence, using a time-step of 120 steps per rotor revolution. Each point, for the aeroelastic model, requires 30 minutes using the CPU computation, while using the hybrid approach CPU/GPU, the simulation time drops to 4 minutes.

The results, shown in Figs. 7, indicate overall agreement between the numerical models and the experimental data. The rigid DUST model consistently overpredicts thrust by approximately 5% across the entire collective range, while the aeroelastic model provides a closer match to the measurements. As shown in Fig. 8, the rigid model tends to overestimate torque at high thrust levels, whereas the aeroelastic model slightly underpredicts it. These trends are further reflected in the Figure of Merit vs. thrust coefficient plot (Fig. 9), where the aeroelastic model demonstrates improved consistency with experimental observations. In summary, a satisfactory agreement is achieved, particularly when blade flexibility is taken into account.

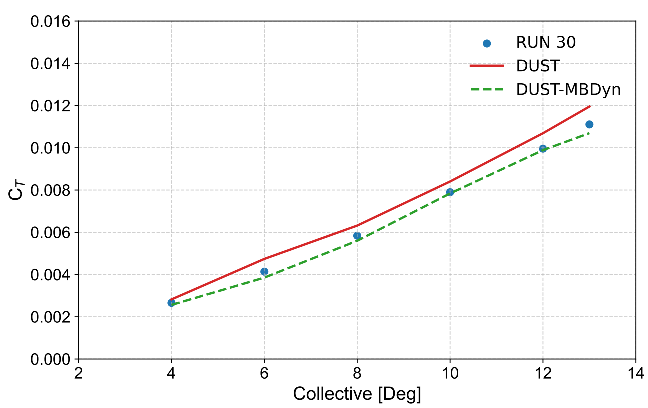

Figure 7: Comparison of thrust coefficient vs imposed collective between the numerical models and the experimental results (RUN 30)

Figure 10 shows the wake evolution for the aeroelastic model, visualized using q-criterion iso-surfaces colored by velocity magnitude. The visualization reveals the presence of tip vortex leapfrogging. Additionally, a secondary inner vortex structure can be observed originating near the blade root.

Figures 11 and 12 show the blade twist deformation and vertical displacement for three collective pitch angles. The nodal positions of the first blade were projected into the local frame of the first beam node, which is rigidly attached to the pitch hinge, at each time step. The results presented correspond to the average displacements over the final rotor revolution.

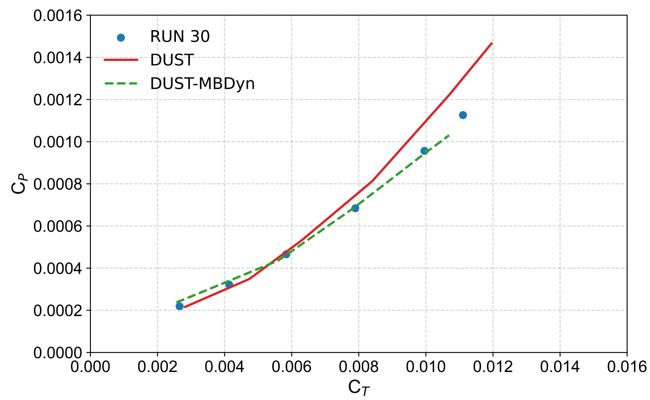

Figure 8: Comparison of torque coefficient vs thrust coefficient between the numerical models and the experimental results (RUN 30)

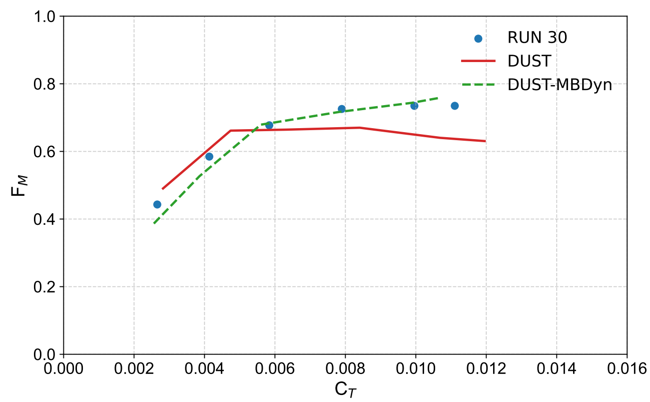

Figure 9: Comparison of Figure of Merit vs thrust coefficient between the numerical models and the experimental results (RUN 30)

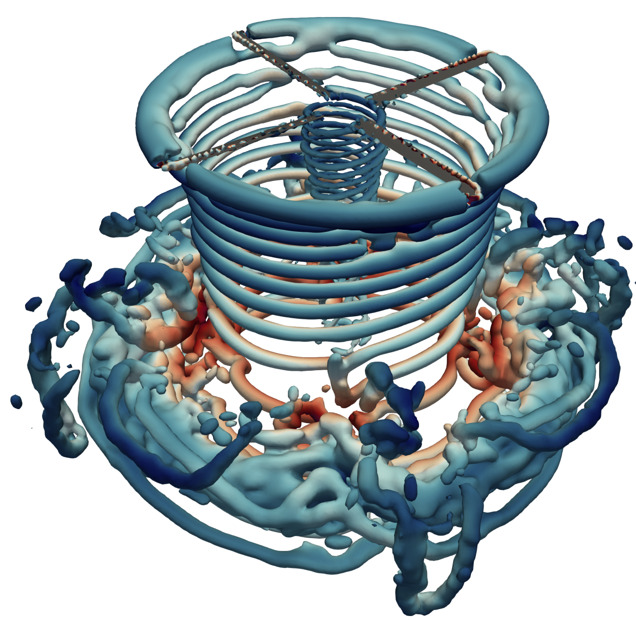

Figure 10: Wake visualization by means of q-criterion (Q=1500) for a collective of 8 deg

Broadly, the model captures the general trends of both torsion and bending deformation. However, it tends to overestimate the tip displacement by approximately 15% and underestimate the tip torsion. These discrepancies at the blade tip suggest localized modeling errors, such as an insufficient number of elements at the tip, though global performance remains well captured.

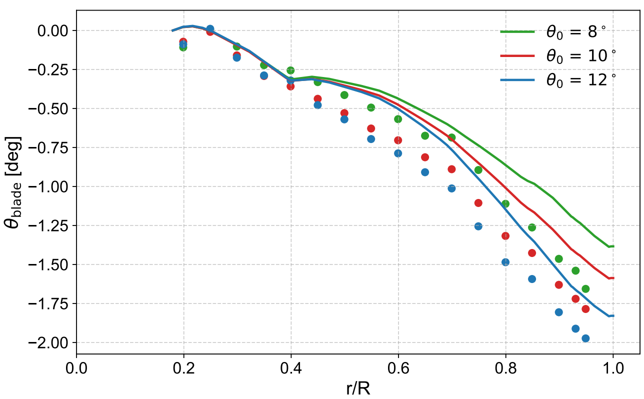

Figure 11: Comparison of the blade twist deformation at various collective angles

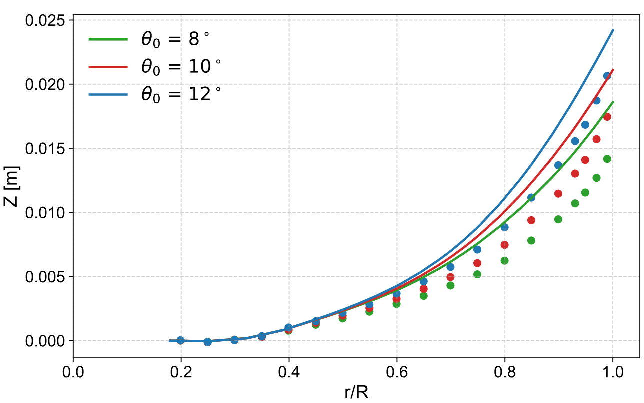

Figure 12: Comparison of the blade bending deformation at various collective angles

CONCLUSIONS

In this study, the DUST-MBDyn aeroelastic tool was employed to assess its capability in predicting rotor hover performance. The NASA hover testbed configuration was replicated within the simulation framework to enable validation.

The results demonstrate that the tool can reliably capture the overall rotor performance in hover. While some discrepancies remain, particularly in torque prediction, these can be largely attributed to sensitivities in the drag coefficients of the airfoil profiles. The inclusion of blade flexibility through the aeroelastic model notably improves agreement with experimental data, highlighting the importance of accounting for structural deformation in articulated rotor analyses.

Given its relatively low computational cost and the quality of both structural and aerodynamic modeling, the DUST-MBDyn tool is well-suited for integration into rotor blade optimization frameworks. Furthermore, the incorporation of machine-learning tools, such as NeuralFoil, offers good opportunities to enhance the speed and accuracy of the design loop by providing rapid aerodynamic evaluations of the airfoil coefficients.

REFERENCES

1. ACARE — Report of the Group of Personalities. European aeronautics: a vision for 2020, January 2001.

2. R. P. Narducci, R. Jain, J. Abras, and N. Hariharan. Hvab rotor hover computational prediction: A comparative study using overflow and hpcmp create-av helios. In *AIAA 2021 SciTech Forum*, 2021.

3. Alberto Savino, Alessandro Cocco, Alex Zanotti, Matteo Tugnoli, Pierangelo Masarati, and Vincenzo Muscarello. Coupling mid-fidelity aerodynamics and multibody dynamics for the aeroelastic analysis of rotary-wing vehicles. *Energies*, 14(21):6979, 2021.

4. Matteo Tugnoli, Davide Montagnani, Monica Syal, Giovanni Droandi, and Alex Zanotti. Mid-fidelity approach to aerodynamic simulations of unconventional vtol aircraft configurations. *Aerospace Science and Technology*, 115:106804, 2021.

5. Pierangelo Masarati, Marco Morandini, and Paolo Mantegazza. An Efficient Formulation for General-Purpose Multibody/Multiphysics Analysis. *Journal of Computational and Nonlinear Dynamics*, 9(4), 07 2014.

6. Thomas R. Norman, Norman W. Schaeffler, Austin D. Overmeyer, Michelle Dominguez, James T. Heineck, Lauren N. Wagner, Manikandan Ramasamy, Edward T. Schairer, Gloria K. Yamauchi, Christopher G. Cameron, and Alex L. Sheikman. Fundamental test of a hovering rotor: Comprehensive measurements for cfd validation. *VFS 79th Annual Forum*, pages 1–27(27), 2023.

7. P. Masarati and M. Morandini. Intrinsic deformable joints. *Multibody System Dynamics*, 23(4):361–386, 2010. doi:10.1007/s11044-010-9194-y.

8. Gian Luca Ghiringhelli, Pierangelo Masarati, and Paolo Mantegazza. A multi-body implementation of finite volume C^0 beams. *AIAA Journal*, 38(1):131–138, January 2000. doi:10.2514/2.933.

9. Marco Morandini and Pierangelo Masarati. Implementation and validation of a 4-node shell finite element. In *ASME IDETC/CIE 2014*, Buffalo, NY, August 17–20 2014. DETC2014-34473.

10. Roy R. Craig, Jr. and Mervyn C. C. Bampton. Coupling of substructures for dynamic analysis. *AIAA Journal*, 6(7):1313–1319, July 1968. doi:10.2514/3.4741.

11. Hans-Joachim Bungartz, Florian Lindner, Bernhard Gatzhammer, Miriam Mehl, Klaudius Scheufele, Alexander Shukaev, and Benjamin Uekermann. preCICE – a fully parallel library for multi-physics surface coupling. *Computers and Fluids*, 141:250–258, 2016. Advances in Fluid-Structure Interaction.

12. Martin Goland. The flutter of a uniform cantilever wing. *Journal of Applied Mechanics-Transactions of the Asme*, 12(4):A197–A208, 1945.

13. Eduardo Alvarez and Andrew Ning. Development of a vortex particle code for the modeling of wake interaction in distributed propulsion. In *AIAA Applied Aerodynamics Conference, Atlanta, GA*, 06 2018.

14. Olivier A. Bauchau, Peter Betsch, Alberto Cardona, Johannes Gerstmayr, Ben Jonker, Pierangelo Masarati, and Valentin Sonneville. Validation of flexible multibody dynamics beam formulations using benchmark problems. *Multibody System Dynamics*, 37(1):29–48, 2016. doi:10.1007/s11044-016-9514-y.

15. Peter Sharpe. NeuralFoil: An airfoil aerodynamics analysis tool using physics-informed machine learning. https://github.com/peterdsharpe/NeuralFoil, 2023.

*

ACKNOWLEDGMENTS The work in this study done by Alessandro Cocco was carried out within MOST— Sustainable Mobility National Research Center and received funding from the European Union Next-Generation EU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.4 – D.D. 1033 17/06/2022, CN00000023). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

CLASSIFICATION & COPYRIGHT