Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Algorithmic Synthesis and Parametric Modeling of Dual Bio-Inspired Airfoils Featuring Leading-Edge Tubercles and Trailing-Edge Serrations

MD. Rokibul Hasan

rokibulhasan9102@gmail.com

Military Institute of Science and Technology https://orcid.org/0009-0002-7996-6619

Method Article

Keywords: leading-edge tubercles, trailing-edge serrations, NACA 4-digit airfoil, parametric CAD, Grasshopper, passive flow control, aerodynamic noise reduction

Posted Date: August 19th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10731961/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Algorithmic Synthesis and Parametric Modeling of Dual Bio-Inspired Airfoils Featuring Leading-Edge Tubercles and Trailing-Edge Serrations

Md. Rokibul Hasan *

Department of Naval Architecture and Marine Engineering
Military Institute of Science and Technology (MIST)
Dhaka 1216, Bangladesh

rokibulhasan9102@gmail.com
ORCID: 0009-0002-7996-6619

*Corresponding author

Submitted: August 2026

Abstract: This paper describes a parametric modeling framework for building three-dimensional airfoil blades that incorporate leading-edge tubercles and trailing-edge serrations — two bio-inspired modifications known to improve stall behaviour and reduce aerodynamic noise. The workflow combines Grasshopper (Rhino 3D) with an embedded Python 3 script that evaluates NACA 4-digit profiles using cosine-spaced discretization and a tunable trailing-edge thickness coefficient. Chord length varies sinusoidally along the span to produce the characteristic humpback-whale-like waviness at the leading edge, and a Boolean subtraction step cuts the serration notches into the trailing edge. The paper walks through the governing equations, the full annotated Python source code, the Grasshopper component layout, a single-variable sensitivity study, and practical notes on preparing the geometry for CFD meshing. The intent is that a reader could reproduce the entire workflow from scratch using only what is presented here.

Keywords: leading-edge tubercles; trailing-edge serrations; NACA 4-digit airfoil; parametric CAD; Grasshopper; passive flow control; aerodynamic noise reduction

Nomenclature

Symbol	Description	Unit
A_{tub}	Tubercle depth amplitude	mm
A_{ser}	Trailing-edge serration depth	mm
a_4	4th-order coefficient of NACA thickness polynomial	–
C_0	Base/root airfoil chord length	mm
$c(z)$	Spanwise local chord length function	mm
H	Total blade span height	mm
m	Maximum camber ratio ($M/100$)	–
N_{cycles}	Number of tubercle wave cycles along span	–
n_{pts}	Number of chordwise discretization points	–
p	Location of maximum camber ($P/100$)	–
$\mathbf{P}_{\text{LE}}$, $\mathbf{P}_{\text{TE}}$	Leading and trailing-edge coordinates	mm
t	Maximum thickness ratio ($T/100$)	–
te_{ratio}	Trailing-edge thickness ratio	–
x_c	Non-dimensional chordwise coordinate ($x/c \in [0, 1]$)	–
x_L, y_L	Lower surface coordinates	mm
x_U, y_U	Upper surface coordinates	mm
y_c	Mean camber line ordinate	mm
y_t	Half-thickness envelope ordinate	mm
z	Spanwise elevation coordinate	mm
Greek Symbols		
δ	Tubercle wave phase shift	rad
λ	Leading-edge tubercle wavelength	mm
θ	Camber line surface normal angle	rad
Acronyms		
BREP	Boundary Representation solid geometry	–
CAD	Computer-Aided Design	–
CFD	Computational Fluid Dynamics	–
NACA	National Advisory Committee for Aeronautics	–
SPL	Sound Pressure Level	–

1. Introduction

The humpback whale (*Megaptera novaeangliae*) has unusual bumps called tubercles running along the leading edges of its pectoral flippers, and for a long time marine biologists assumed they were scars or parasitic growths. It turned out they are aerodynamically purposeful: Fish and Battle showed that the bumps produce stream-wise vortices that keep the boundary layer attached longer, pushing stall to steeper angles of attack than a smooth flipper would allow [1]. Wind tunnel experiments by Miklosovic et al. reproduced this on a scaled model and found lift improvement of about 6% post-stall alongside a narrower drag bucket [2]. Later, Johari et al. and Hansen et al. tested the concept on standard NACA profiles and broadly confirmed that tubercle protuberances delay flow separation, though the magnitude varies considerably with profile shape and Reynolds number [3, 4].

A related observation comes from nocturnal owls, whose leading-edge feather serrations scatter trailing-edge noise into incoherent higher-frequency content that is less audible [6]. This acoustics motivation fed directly into wind-turbine research: analytical models [8] and field trials [9] both showed that sawtooth trailing-edge add-ons can knock several decibels off broadband noise at relatively modest aerodynamic cost. The depth and pitch of the serrations control which frequency band benefits most [7].

Applying either modification to a real blade requires a geometry that can be varied and rebuilt quickly. Standard CAD tools can handle individual surface patches but they do not naturally link a profile equation to a spanwise sinusoidal sweep and an automated cutting operation. Grasshopper, the visual programming plugin for Rhinoceros 3D [12], fits this need well because it can wire analytical expressions directly into solid modelling operations. The goal of this work is a fully parametric pipeline where adjusting a single slider in Grasshopper instantly regenerates a geometrically correct, watertight blade ready for CFD import — without any manual rebuilding.

The rest of the paper is organised as follows. Section 2 works through the governing equations. Section 3 gives the Python script. Section 4 describes the Grasshopper component layout. Section 5 reports a single-variable sensitivity study, and Section 6 notes practical considerations for CFD mesh generation.

2. Governing Mathematical Equations

2.1. Modified NACA 4-Digit Thickness Envelope

The NACA 4-digit half-thickness as a function of non-dimensional chord position $x_c \in [0, 1]$ is

$$y_t(x_c) = 5t \left(0.2969\sqrt{x_c} - 0.1260\, x_c - 0.3516\, x_c^2 + 0.2843\, x_c^3 + a_4\, x_c^4\right) \tag{1}$$

where $t = T/100$ is the thickness-to-chord ratio. The classical value $a_4 = -0.1036$ gives a sharp trailing edge. Here, a_4 is back-calculated to enforce a user-specified trailing-edge thickness

$te_{\text{ratio}} = TE_{\text{Thickness}}/100$. Setting $y_t(1) = te_{\text{ratio}}/2$ and solving gives

$$a_4 = \frac{te_{\text{ratio}}/2}{5t} - 0.1036 \tag{2}$$

Setting $te_{\text{ratio}} = 0$ recovers the sharp-edge original.

2.2. Camber Line Equations

The mean camber line $y_c(x_c)$ and its slope depend on maximum camber $m = M/100$ and camber location $p = P/100$.

2.2.1. Forward region ($0 \le x_c \le p$)

$$y_c = \frac{m}{p^2}\left(2p\,x_c - x_c^2\right), \qquad \frac{dy_c}{dx_c} = \frac{2m}{p^2}\left(p - x_c\right) \tag{3}$$

2.2.2. Aft region ($p < x_c \le 1$)

$$y_c = \frac{m}{(1-p)^2}\left[(1-2p) + 2p\,x_c - x_c^2\right], \qquad \frac{dy_c}{dx_c} = \frac{2m}{(1-p)^2}\left(p - x_c\right) \tag{4}$$

For symmetric profiles, $m = 0$ gives $y_c = dy_c/dx_c = 0$ everywhere.

2.3. Upper and Lower Surface Coordinates

The surface normal angle is

$$\theta(x_c) = \begin{cases} 0 & x_c < 10^{-6} \\ \arctan\left(\dfrac{dy_c}{dx_c}\right) & x_c \ge 10^{-6} \end{cases} \tag{5}$$

Scaling by local chord c, the absolute surface coordinates are

$$x_U = c\left(x_c - y_t \sin\theta\right), \qquad y_U = c\left(y_c + y_t \cos\theta\right) \tag{6}$$

$$x_L = c\left(x_c + y_t \sin\theta\right), \qquad y_L = c\left(y_c - y_t \cos\theta\right) \tag{7}$$

2.4. Cosine Spacing and Boundary Anchoring

Points are clustered near the leading edge using cosine spacing:

$$x_c(i) = \tfrac{1}{2}\left[1 - \cos\left(\frac{\pi i}{n_{\text{pts}} - 1}\right)\right], \quad i = 0, \ldots, n_{\text{pts}} - 1 \tag{8}$$

After computing all intermediate points, the leading and trailing edges are pinned exactly:

$$\mathbf{P}_{\text{LE}} = (0,\, 0,\, 0)^\top, \qquad \mathbf{P}_{\text{TE}} = (c,\, 0,\, 0)^\top \tag{9}$$

This prevents floating-point drift from breaking the curve closure condition.

2.5. Sinusoidal Spanwise Chord Modulation

The leading-edge tubercles arise from varying local chord $c(z)$ sinusoidally along the blade span $z \in [0, H]$:

$$c(z) = C_0 + A_{\text{tub}} \sin\left(\frac{2\pi z}{\lambda} + \delta\right) \tag{10}$$

Here C_0 is the root chord, A_{tub} the tubercle amplitude, λ the wavelength, and $\delta = \Delta_{\text{deg}} \cdot \pi/180$ converts the phase angle from degrees to radians. The total number of complete sine cycles is

$$N_{\text{cycles}} = H/\lambda \tag{11}$$

3. Algorithmic Source Code

Listing 1 shows the Python 3 script inside the `Airfoil Generator Code` Grasshopper component. It takes the list of modulated chord values from upstream, generates a closed B-spline for each, and passes the list forward to the Loft component.

```
import Rhino.Geometry as rg
import math

# Inputs: M (camber %), P (position %), T (thickness %),
#         chords (list), nPts (int), TE_thick_per (%)
m        = max(0.0,   min(float(M)            / 100.0, 0.20))
p        = max(0.01,  min(float(P)            / 100.0, 0.99))
t        = max(0.001, min(float(T)            / 100.0, 0.40))
te_ratio = max(0.0,       float(TE_thick_per) / 100.0)
nPts     = max(20, int(nPts))

te_coeff = (te_ratio / 2.0) / (5.0 * t) - 0.1036

def half_thickness(xc):
    if xc <= 0.0:
        return 0.0
    return 5.0 * t * (
          0.2969 * math.sqrt(xc)
        - 0.1260 * xc
        - 0.3516 * xc ** 2
        + 0.2843 * xc ** 3
        + te_coeff * xc ** 4
    )

def camber_and_gradient(xc):
    if m == 0.0:
        return 0.0, 0.0
    if xc <= p:
        yc     = (m / p ** 2) * (2.0 * p * xc - xc ** 2)
        dyc_dx = (2.0 * m / p ** 2) * (p - xc)
    else:
        yc     = (m / (1.0 - p) ** 2) * ((1.0 - 2.0 * p) + 2.0 * p * xc - xc ** 2)
```

```
        dyc_dx = (2.0 * m / (1.0 - p) ** 2) * (p - xc)
    return yc, dyc_dx

def generate_airfoil(chord):
    if chord <= 0:
        return None
    xc_vals = [0.5 * (1.0 - math.cos(math.pi * i / (nPts - 1))) for i in range(nPts)]
    upper_pts, lower_pts = [], []
    for xc in xc_vals:
        x  = xc * chord
        yt = half_thickness(xc) * chord
        yc, dyc_dx = camber_and_gradient(xc)
        yc_mm = yc * chord
        theta = 0.0 if xc < 1e-6 else math.atan(dyc_dx)
        upper_pts.append(rg.Point3d(x - yt * math.sin(theta),
                                    yc_mm + yt * math.cos(theta), 0.0))
        lower_pts.append(rg.Point3d(x + yt * math.sin(theta),
                                    yc_mm - yt * math.cos(theta), 0.0))
    # Pin leading and trailing edges exactly
    upper_pts[0]  = rg.Point3d(0.0,   0.0, 0.0)
    lower_pts[0]  = rg.Point3d(0.0,   0.0, 0.0)
    upper_pts[-1] = rg.Point3d(chord, 0.0, 0.0)
    lower_pts[-1] = rg.Point3d(chord, 0.0, 0.0)
    upper_crv = rg.Curve.CreateInterpolatedCurve(upper_pts, 3)
    lower_crv = rg.Curve.CreateInterpolatedCurve(lower_pts, 3)
    if not upper_crv or not lower_crv:
        return None
    lower_crv.Reverse()
    joined = rg.Curve.JoinCurves([upper_crv, lower_crv], chord * 0.001, True)
    if joined and len(joined) > 0 and joined[0].IsClosed:
        return joined[0]
    return None

airfoilCurves = []
for c in chords:
    crv = generate_airfoil(float(c))
    if crv and crv.IsValid and crv.IsClosed:
        airfoilCurves.append(crv)
```

Listing 1: Airfoil Generator Code Component.

4. Grasshopper CAD Architecture

Table 1 lists all the user-adjustable sliders. Figure 1 maps the data flow from those sliders to the final solid.

The build sequence follows six steps:

1. **Parameter initialisation.** Number sliders define C_0, H, t, m, p, and te_{ratio}. These feed downstream through persistent wires so any change propagates immediately.

2. **Sinusoidal chord sweep.** An Expression Node evaluates Eq. (8) at each spanwise station, giving the array of local chord lengths $c(z_k)$. The amplitude slider A_{tub} and wavelength slider λ jointly shape the wave; the phase slider δ positions it so the designer can choose whether a crest or trough sits at mid-span.

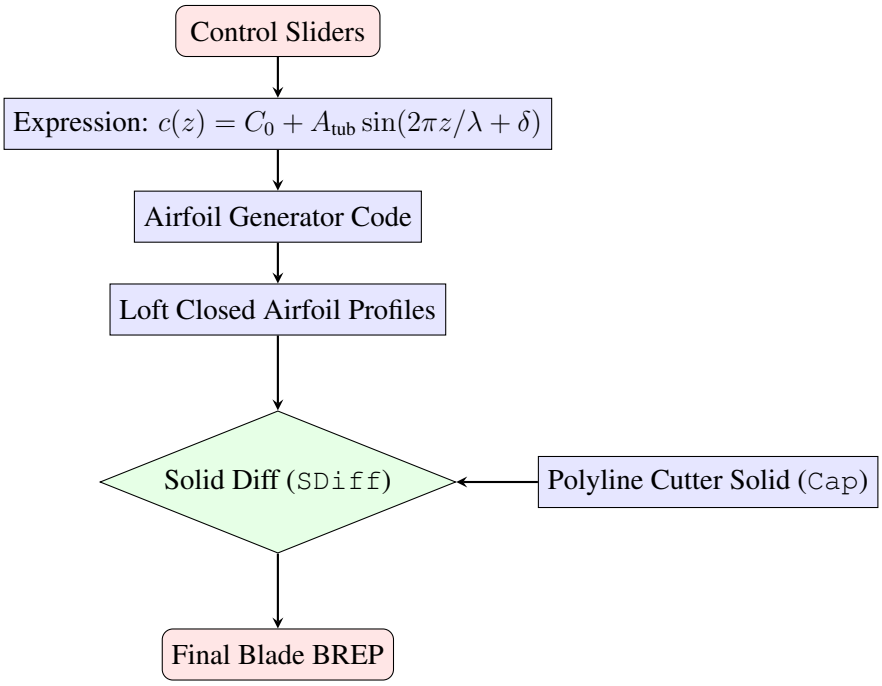


Figure 1: Data flow through the Grasshopper pipeline, from user sliders to final blade solid.

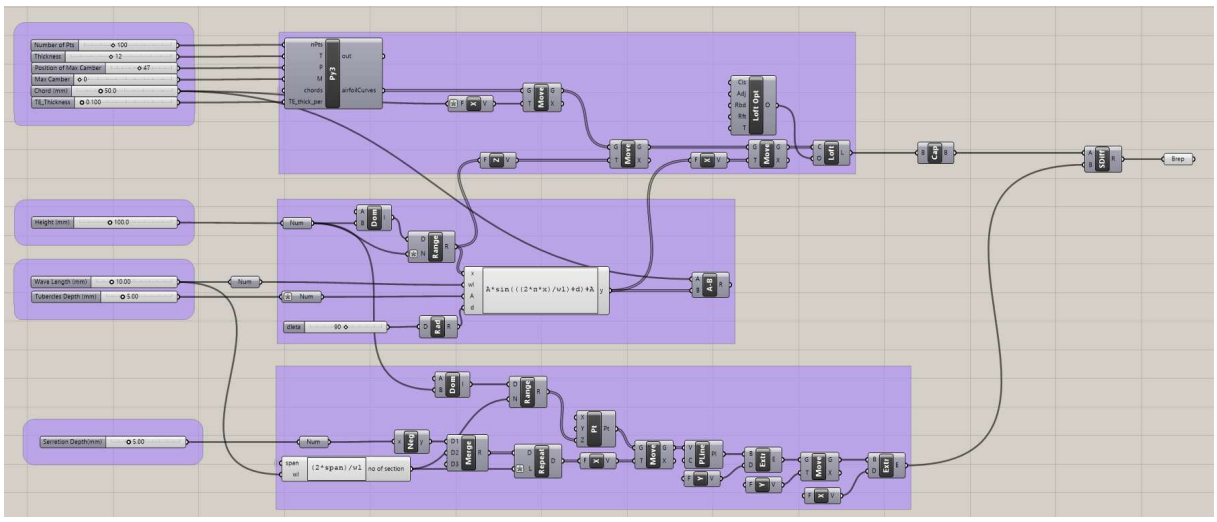


Figure 2: The Grasshopper canvas showing parameter sliders (left), the Airfoil Generator Code Python component (centre), the sinusoidal chord expression, and the Boolean subtraction that cuts the serration notches.

Table 1: Parametric slider matrix in Grasshopper.

Parameter Name	**Symbol**	**Default**	**Range**	**Unit**
Base Chord	C_0	100.0	1–200	mm
Number of Points	n_{pts}	100	20–300	–
Thickness Ratio	t/c	12.0	0.1–30	%
Camber Position	p/c	47.0	1–99	%
Max Camber	m/c	0.0	0–20	%
TE Thickness Ratio	te_{ratio}	0.1	0–5	%
Blade Height	H	100.0	1–300	mm
Tubercle Wavelength	λ	50.0	1–50	mm
Tubercle Depth	A_{tub}	20.0	0–20	mm
Serration Depth	A_{ser}	5.0	0–20	mm
Phase Shift Angle	δ	90.0	0–180	deg

3. **Airfoil profile generation.** The chord array enters `Airfoil Generator Code`, which runs the Python script and returns a list of closed $\mathcal{C}^2$ B-spline curves, one per spanwise section.

4. **Spatial placement and lofting.** A Move component raises each curve to its correct z height. A Loft component fits a smooth surface through all cross-sections, producing the blade volume V_{blade}.

5. **Serration cutter construction.** A sawtooth polyline is drawn at the root plane, extruded across the full span, and capped to form the closed solid V_{cutter}.

6. **Boolean subtraction.** SDiff computes $V_{\text{serrated}} = V_{\text{blade}} \setminus V_{\text{cutter}}$, leaving the sawtooth notches in the trailing edge.

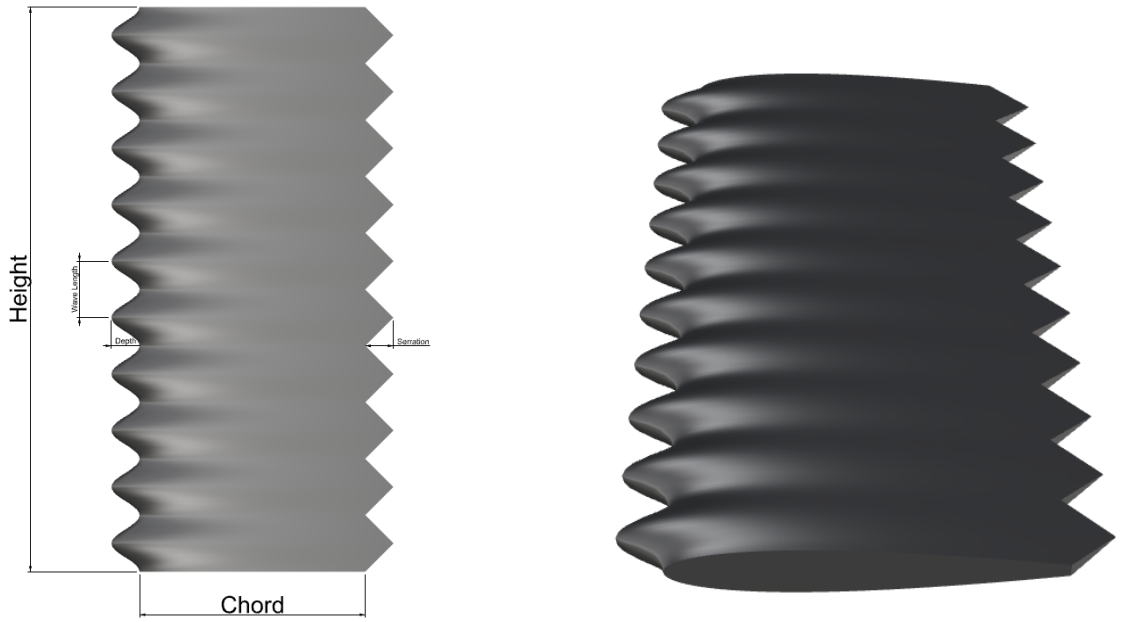
5. Parametric Sensitivity Study

To understand how each design knob affects the blade shape, a one-at-a-time study was run over the three main bio-inspired parameters. All other sliders were held at their defaults (Table 1).

A few things stand out:

- **Tubercle depth (A_{tub}).** Raising the amplitude widens the chord envelope linearly — the spread is $\Delta c = 2A_{\text{tub}}$. The cycle count does not change at all. At high amplitudes ($A_{\text{tub}} > 0.2\,C_0$), the narrowest cross-sections get quite thin and the lofter can struggle; this is worth checking case by case.

- **Tubercle wavelength (λ).** Shortening the wavelength packs more cycles into the same span without touching the chord amplitude. Fine wavelengths (below about 10 mm at $C_0 = 100$ mm) need a higher point count n_{pts} to prevent faceting.



(a) Annotated orthographic schematic. (b) Isometric 3D solid model.

Figure 3: The synthesised blade with leading-edge tubercles and trailing-edge serrations.



Figure 4: Close-up labelling tubercle wavelength λ, tubercle depth A_{tub}, and serration depth A_{ser}.

Table 2: Geometric response to changes in the three primary bio-inspired parameters. Baseline: $C_0 = 100\,\text{mm}$, $H = 100\,\text{mm}$, $t = 12\%$, $m = 0\%$, $n_{\text{pts}} = 100$.

Parameter	**Value**	$\boldsymbol{c_{\min}}$ (mm)	$\boldsymbol{c_{\max}}$ (mm)	$\boldsymbol{\Delta c}$ (mm)	$\boldsymbol{N}_{\text{cycles}}$
A_{tub} (mm)	0	100.0	100.0	0.0	2.0
	5	95.0	105.0	10.0	2.0
	10	90.0	110.0	20.0	2.0
	20	80.0	120.0	40.0	2.0
λ (mm)	10	80.0	120.0	40.0	10.0
	25	80.0	120.0	40.0	4.0
	50	80.0	120.0	40.0	2.0
	100	80.0	120.0	40.0	1.0
A_{ser} (mm)	0	–	–	–	–
	5	Notch depth = 5 mm; chord envelope unchanged			
	10	Notch depth = 10 mm; chord envelope unchanged			
	20	Notch depth = 20 mm; chord envelope unchanged			

- **Serration depth (A_{ser}).** This parameter affects only the Boolean cutter and has no effect on the chord distribution. The practical upper limit is roughly $A_{\text{ser}} < 0.5\,C_0$; deeper cuts risk punching through the profile interior.

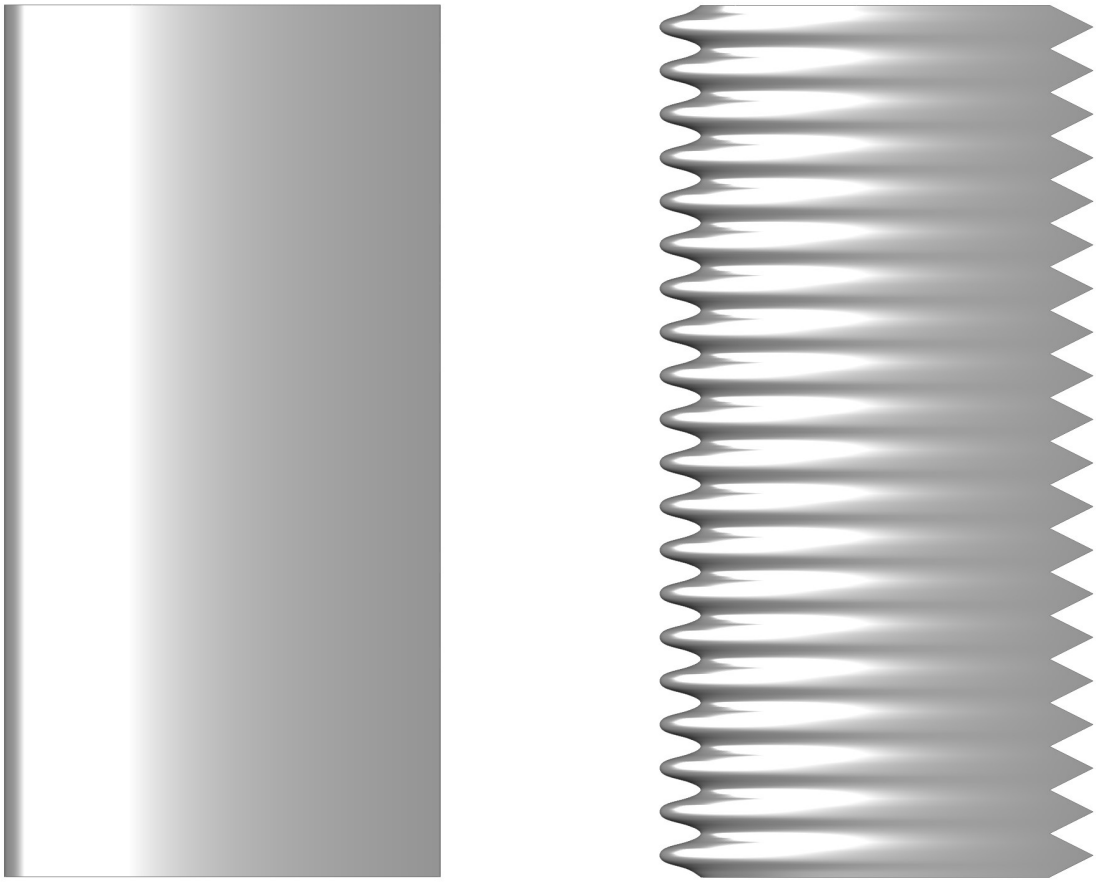
5.1. Visual Geometry Gallery

Figures 5 and 6 show rendered frontal views of eight configurations exported from Rhino. Sub-captions follow the convention $(\lambda,\ A_{\text{tub}},\ A_{\text{ser}})$ in mm.

6. CFD Pre-Processing Notes

Once the blade is built and visually checked, a few things are worth addressing before generating a CFD mesh:

- **Surface continuity.** The lofted surface is $\mathcal{C}^2$ across all cross-sections. After export and re-import, it is worth checking that no gaps or overlapping patches appeared at the import step.
- **Boundary layer grid.** The tubercle troughs and crests are where spanwise secondary vortices roll up [13]. Prismatic inflation layers with $y^+ < 1$ should cover the entire leading edge, not just mid-span.
- **Export format.** STEP (ISO 10303) and Parasolid `x_t` both export cleanly from Rhino and open without topology loss in ANSYS Fluent, OpenFOAM, and Star-CCM+.
- **Domain size.** A C-type or O-type far-field extending roughly $20c$ from the blade surface is generally sufficient to avoid blockage effects on integrated force coefficients [10].

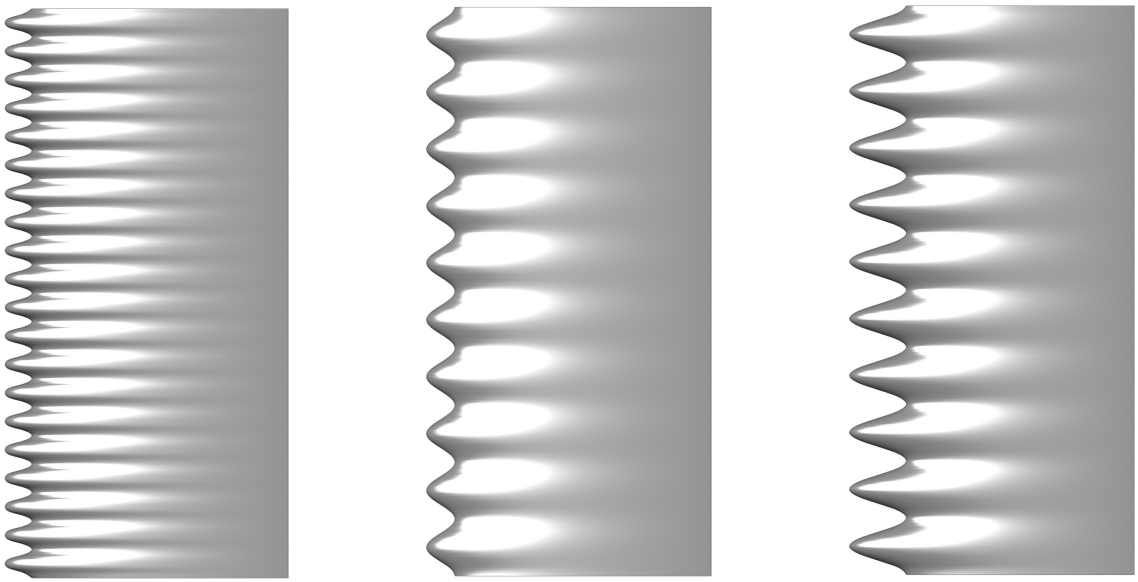


(a) Smooth baseline. $A_{\text{tub}} = 0\,\text{mm}$, $A_{\text{ser}} = 0\,\text{mm}$.

(b) Fully bio-inspired: $\lambda = 5\,\text{mm}$, $A_{\text{tub}} = 5\,\text{mm}$, $A_{\text{ser}} = 5\,\text{mm}$.

Figure 5: Left: plain NACA baseline. Right: both tubercles and serrations active.

(a) Tubercle-only sweep ($A_{\text{ser}} = 0\,\text{mm}$)

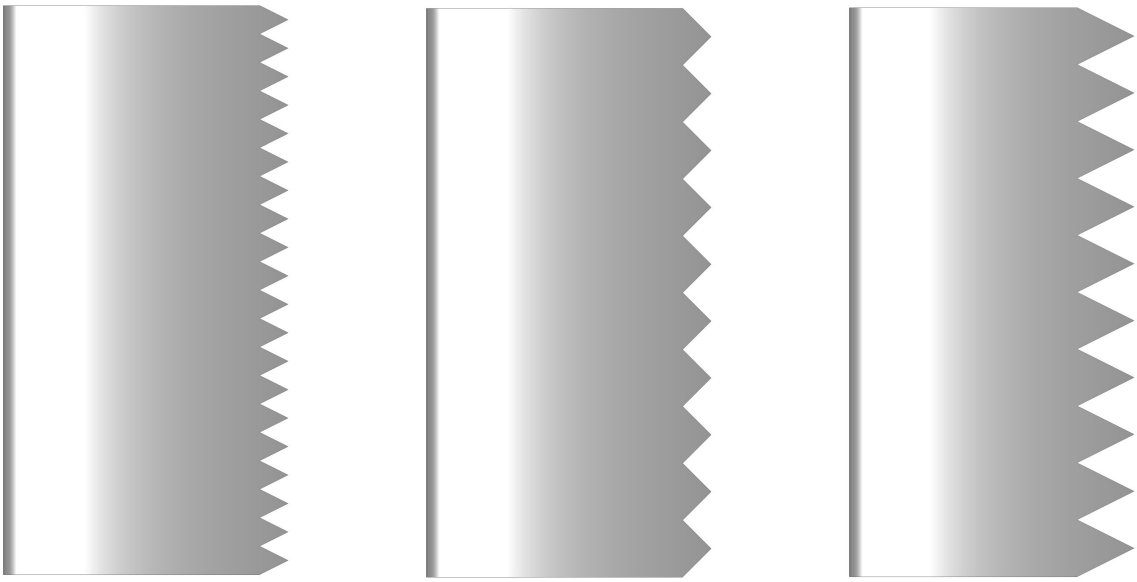


(a) $\lambda = 5\,\text{mm}$, $A_{\text{tub}} = 5\,\text{mm}$, $A_{\text{ser}} = 0\,\text{mm}$

(b) $\lambda = 10\,\text{mm}$, $A_{\text{tub}} = 5\,\text{mm}$, $A_{\text{ser}} = 0\,\text{mm}$

(c) $\lambda = 10\,\text{mm}$, $A_{\text{tub}} = 10\,\text{mm}$, $A_{\text{ser}} = 0\,\text{mm}$

(b) Serration-only sweep ($A_{\text{tub}} = 0\,\text{mm}$)



(d) $\lambda = 5\,\text{mm}$, $A_{\text{tub}} = 0\,\text{mm}$, $A_{\text{ser}} = 5\,\text{mm}$

(e) $\lambda = 10\,\text{mm}$, $A_{\text{tub}} = 0\,\text{mm}$, $A_{\text{ser}} = 5\,\text{mm}$

(f) $\lambda = 10\,\text{mm}$, $A_{\text{tub}} = 0\,\text{mm}$, $A_{\text{ser}} = 10\,\text{mm}$

Figure 6: Parametric geometry sweeps. **(a) Tubercle-only** ($A_{\text{ser}} = 0$): shorter λ increases undulation count; larger A_{tub} deepens each lobe. **(b) Serration-only** ($A_{\text{tub}} = 0$): shorter λ gives more teeth per span; larger A_{ser} cuts deeper notches.

7. Limitations and Future Work

The current implementation works well for the geometries it targets, but several boundaries are worth being clear about:

1. **No sweep or twist.** Cross-sections are generated in planes orthogonal to the Z-axis. Blades with geometric sweep, dihedral, or spanwise twist need additional transformation steps not yet in the pipeline.

2. **Purely sinusoidal leading edge.** The chord modulation is a single-harmonic sine. Non-sinusoidal or asymmetric planforms — sawtooth or truncated-cosine tubercles [3] — cannot be reproduced without modifying the Expression Node directly.

3. **NACA 4-digit profiles only.** The thickness polynomial is fixed to the 4-digit series. Switching to a 5-digit, 6-series, or custom profile requires replacing the polynomial or reading coordinates from an external file.

4. **No built-in solver.** The pipeline stops at solid geometry. Mesh generation, boundary conditions, and solver configuration are all handled outside Grasshopper [13].

5. **Rectangular serration cutters.** Trailing-edge teeth are straight-sided. Real owl-feather serrations have oblique tooth angles and micro-serration sub-features [6] not reproduced by the current planar polyline cutter.

The most natural next steps are adding spanwise twist interpolation, supporting imported profile coordinates from the UIUC database [10], and scripting a direct OpenFOAM mesh-export routine.

Acknowledgements

The author thanks the Department of Naval Architecture and Marine Engineering at the Military Institute of Science and Technology (MIST), Dhaka, for access to computing facilities and the supportive research environment. The Grasshopper plugin developed by Robert McNeel & Associates is the foundation of the computational workflow described here.

8. Conclusions

This paper described a working parametric pipeline for building serrated airfoil blades with leading-edge tubercles. The approach ties together a NACA profile generator written in Python, a sinusoidal chord-modulation expression, a Grasshopper loft, and a Boolean cutting step for the trailing-edge serrations. All equations are derived from first principles and the full source code is included, so the workflow should be straightforward to reproduce. The sensitivity study showed

that tubercle depth and wavelength each control a distinct geometric feature — amplitude and cycle count, respectively — and do not interfere with each other, making the parameter space relatively easy to navigate. Serration depth is geometrically independent of the chord modulation, so the two features can be tuned separately without affecting each other.

The work stops short of aerodynamic predictions; CFD results and experimental validation are planned as follow-on work once the geometry and meshing workflow are finalised for a chosen set of test cases.

References

[1] F. E. Fish and J. M. Battle, "Hydrodynamic design of the humpback whale flipper," *Journal of Morphology*, vol. 225, no. 1, pp. 51–60, 1995.

[2] D. S. Miklosovic, M. M. Murray, L. E. Howle, and F. E. Fish, "Leading-edge tubercles delay stall on humpback whale (*Megaptera novaeangliae*) flippers," *Physics of Fluids*, vol. 16, no. 5, pp. L39–L42, 2004.

[3] H. Johari, C. Henoch, D. Custodio, and A. Levshin, "Effects of leading-edge protuberances on airfoil performance," *AIAA Journal*, vol. 45, no. 11, pp. 2634–2642, 2007.

[4] K. L. Hansen, R. M. Kelso, and B. B. Dally, "Performance variations of leading-edge tubercles for distinct airfoil profiles," *AIAA Journal*, vol. 49, no. 1, pp. 185–194, 2011.

[5] M. J. Stanway, "Hydrodynamic effects of leading-edge tubercles on control surfaces and in flapping foil propulsion," M.S. thesis, Massachusetts Institute of Technology, 2008.

[6] M. Weger and H. Wagner, "Morphological variation of leading-edge serrations in owls (*Strigiformes*)," *PLOS ONE*, vol. 11, no. 3, p. e0149236, 2016.

[7] M. Gruber, P. F. Joseph, and T. P. Chong, "Aerofoil self-noise and directivity," *Journal of Sound and Vibration*, vol. 330, no. 16, pp. 3835–3856, 2011.

[8] M. S. Howe, "Noise produced by a sawtooth trailing edge," *Journal of the Acoustical Society of America*, vol. 90, no. 1, pp. 482–487, 1991.

[9] S. Oerlemans, M. Fisher, T. Maeder, and K. Kögler, "Reduction of wind turbine noise using optimized airfoils and trailing-edge serrations," *AIAA Journal*, vol. 47, no. 6, pp. 1470–1481, 2009.

[10] M. S. Selig, J. J. Guglielmo, A. P. Broeren, and P. Giguere, *Summary of Low-Speed Airfoil Data*, vol. 1. SoarTech Publications, Virginia Beach, VA, 1995.

[11] M. Drela, "XFOIL: An analysis and design system for low Reynolds number airfoils," in *Low Reynolds Number Aerodynamics*, Springer, Berlin, 1989, pp. 1–12.

[12] D. Rutten, "Grasshopper: Algorithmic modelling for Rhino," Robert McNeel & Associates, Seattle, WA, 2010. [Online]. Available: `https://www.grasshopper3d.com`

[13] N. L. Guerreiro and J. M. Sanz-Andres, "Effects of leading-edge tubercles on the aerodynamic performance of small aspect ratio wings at low Reynolds numbers," *Journal of Fluids Engineering*, vol. 134, no. 4, p. 041201, 2012.

[14] E. N. Jacobs, K. E. Ward, and R. M. Pinkerton, "The characteristics of 78 related airfoil sections from tests in the variable-density wind tunnel," NACA Technical Report TR-465, 1933.