Biomimetic Nose Cone Design for Aerodynamic Optimization of Model Rockets

Table of Contents

Abstract 3

1. Introduction 3

    1.1 Research Motivation 3

    1.2 Research Objectives 3

2. Literature Review 4

3. Research Methodology 8

    3.1 Design and Modeling 8

    3.2 Simulation and Fabrication 15

    3.3 Physical Testing 16

4. Future Work 19

5. References 20

Abstract

This study integrates biomimetics, curve fitting, and fluid dynamics to explore naturally occurring shapes that exhibit low drag and high stability, and applies these findings to rocket nose cone design. We selected streamlined biological structures —such as the heads of sharks, the beaks of swallows, and the heads of sailfish and eagles—and reconstructed them into biomimetic models using tools such as Open3D and curve fitting techniques. Computational fluid dynamics (CFD) simulations were performed in OpenFOAM, followed by experimental validation through wind tunnel tests, structural strength measurements, and prototype launches. These tests aim to verify the real-world performance of biomimetic nose cones and assess their aerodynamic efficiency.

1. Introduction

1.1 Research Motivation

I participated in the Taiwan Rocket Cup organized by TASA. As members of the structural design group, I'm responsible for the overall rocket structure. During the design process, I became intrigued by the variety of nose cone shapes. Beyond the commonly used forms—such as conical, ogive, ellipsoid, power series, parabolic, and Haack shapes frequently seen in our design software *OpenRocket*—I began to wonder whether alternative designs might achieve higher efficiency and greater altitude. I'm also questioned why these traditional shapes were the dominant choice in modern rocketry.

Inspired by high-speed biological organisms, I hypothesized that integrating features from animals capable of rapid flight or swimming might improve rocket performance. Specifically, I considered whether the unique body shapes of these species could be applied to reduce aerodynamic drag and enhance stability. This curiosity drove us to initiate a research project exploring the feasibility of such biomimetic designs.

1.2 Research Objectives

The primary aim of this study is to identify the most effective nose cone design. Our initial literature review revealed that “effectiveness” encompasses multiple dimensions and should not be measured solely by aerodynamic drag. Drawing from the aerodynamic design principles of high-speed trains, I identified three core performance indicators for evaluating nose cone designs:

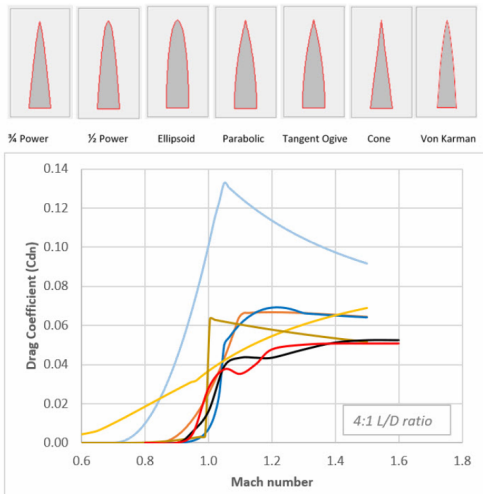
1. **Drag Coefficient (Cd):** I hypothesize that biomimetic nose cones—particularly those with blending ratios α = 0.6–0.8—will better guide airflow, reduce pressure concentration, and minimize flow separation, potentially lowering Cd by 10–20% compared to conventional ogive designs.
2. **Flow Attachment:** CFD simulations and smoke flow visualization in wind tunnel tests are expected to show that high-α biomimetic designs (e.g., swift and shark) allow smoother airflow attachment and smaller vortices, resulting in improved stability.
3. **Pitch Stability:** Due to smoother curvature transitions at the rear of biomimetic nose cones, I anticipate reduced sudden changes in pitching moments during flight, thereby improving trajectory stability.

These indicators will serve as the main basis for simulation and experimental analysis. Even at the conceptual and modeling stage, I predict that biomimetic designs with α = 0.6–0.8 will significantly outperform the traditional ogive shape in aerodynamic performance, a hypothesis to be tested in subsequent experiments.

2. Literature Review

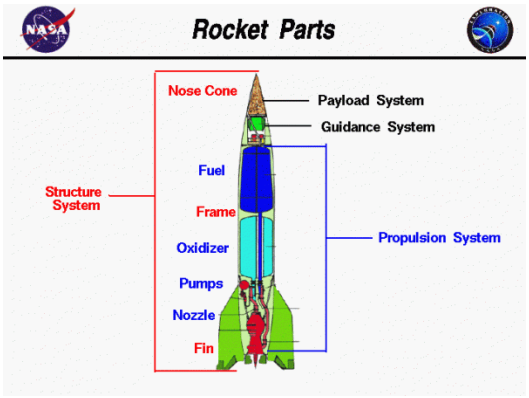
In the design of high-speed vehicles such as rockets, trains, and missiles, the shape of the nose cone has a decisive influence on overall aerodynamic performance. Traditional nose cone designs often employ simplified geometric profiles—such as ogive, conical, or parabolic shapes—optimizing drag reduction primarily by

adjusting length-to-diameter ratios and curvature radii (Barrowman, 1967). However, under high-speed conditions, purely parameterized designs often fail to simultaneously achieve lift stability, wake control, and adaptability to varying flow regimes.



圖三 資料來源（Richard Nakka’s Experimental Rochetry Web Site）

In terms of rocket structure, typical designs include a **nose cone**, **payload bay**, **body tube**, **fins**, and a **propulsion system**. Separation mechanisms (e.g., shear pins) between the nose cone and the body, along with precise stability adjustments based on center of gravity and center of pressure, are critical for mission success.



圖一 資料來源（NASA Research Center）

In recent years, **biomimetic engineering** has been increasingly applied in aerodynamics, with researchers drawing inspiration from high-speed organisms in

nature to develop more optimized external forms. For example, the streamlined shape of a shark has been shown to reduce turbulence and pressure drag (Bechert et al., 2000), while the head profiles of flying birds such as swallows and eagles provide excellent stability and lift control during diving and gliding.

Shape blending or **morphing techniques** have also been widely used in the development of wings, vehicle bodies, and structural components. By blending multiple geometric models through weighted interpolation, designers can generate continuously variable intermediate shapes and explore optimal design spaces (Wang et al., 2016). In the field of 3D modeling, freeform surface technologies such as Bézier curves and subdivision surfaces are widely adopted to control smoothness and local curvature, providing powerful tools for fitting and reproducing biomimetic geometries.

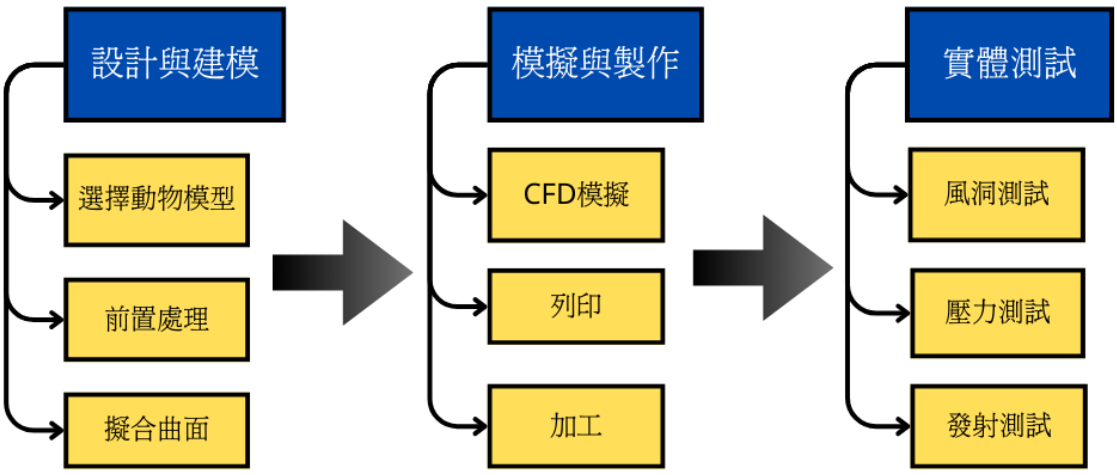
In addition, **wind tunnel testing** has long been a core method for visualizing airflow interaction with aerodynamic models, often using smoke filaments or particle tracking to observe flow attachment, separation, and wake formation. Low-speed wind tunnels are widely used for preliminary aerodynamic assessments, and CFD simulations often complement wind tunnel experiments to validate results.

圖四 資料來源：維基百科

In summary, integrating biomimetic profiles into nose cone design—combined with shape blending techniques and parameterized control—offers the potential to balance aerodynamic performance with innovative form. However, most existing

studies focus on aircraft, high-speed train noses, and vehicle shells, while biomimetic-blended designs specifically for rocket nose cones remain scarce. This study therefore seeks to address this research gap by focusing on biomimetic nose cone design, combining modeling techniques with performance analysis to explore its aerodynamic potential.

3. Research Methodology



圖五 自行繪製

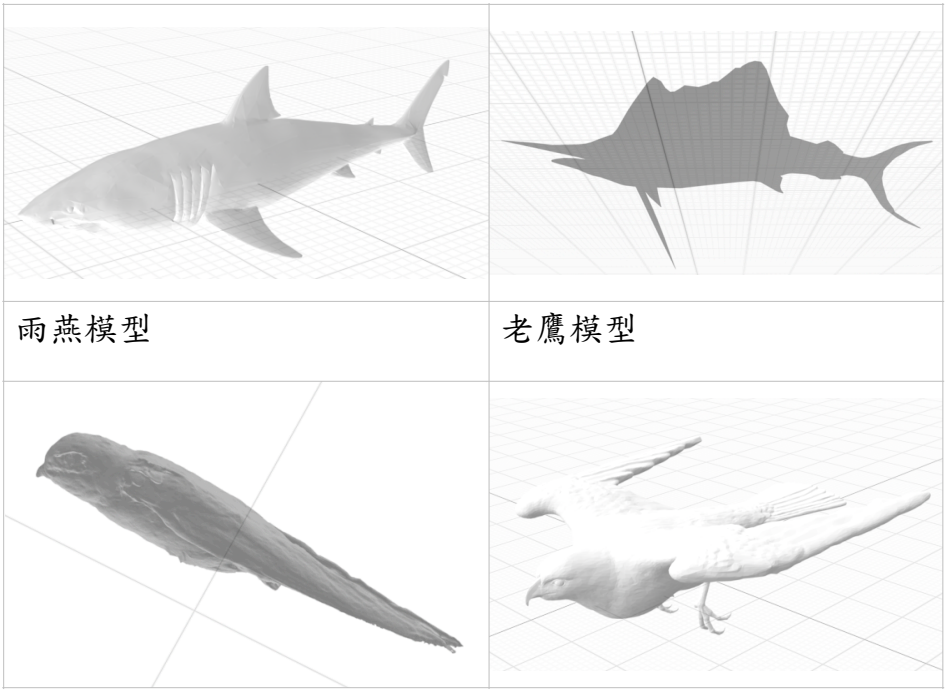
3.1 Design and Modeling

(1) Selection of Biological 3D Models

Four biomimetic sources were chosen based on their streamlined head or beak geometries: **shark**, **sailfish**, **swallow**, and **eagle**. These shapes are known for their low-drag profiles and stability in high-speed movement. 3D models were obtained from open-source databases (e.g., Sketchfab).

(Figure 5. Selected 3D models of shark, sailfish, swallow, and eagle used as biomimetic references.)

鯊魚模型	槍魚模型



雨燕模型	老鷹模型

表一 初始動物模型

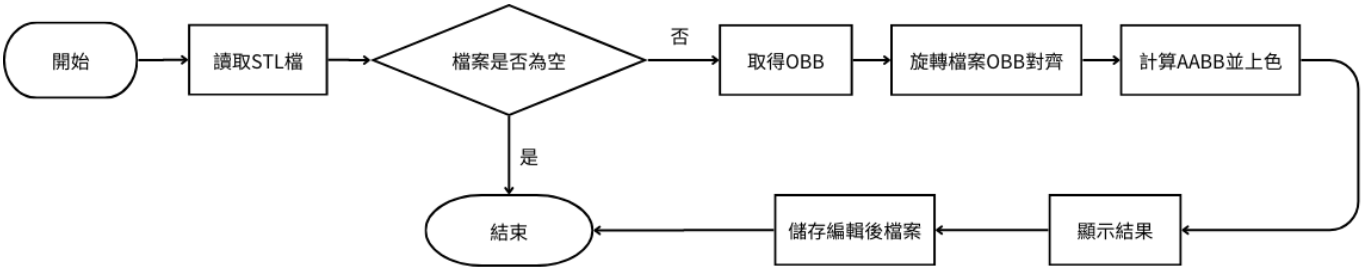
模型來源：SKETCHFAB.COM

(2) Model Preprocessing

(a) Orientation Alignment (Turning)

A Python script using the **Open3D** library was developed to automatically align STL models to the global coordinate system. The script:

1. Reads the STL mesh and verifies successful loading.
2. Computes the oriented bounding box (OBB) to determine the model's principal orientation.
3. Applies an inverse rotation to align the main axis with the global axes.
4. Recalculates the axis-aligned bounding box (AABB) for visual inspection.
5. Computes vertex normals for improved rendering and simulation performance.
6. Outputs the aligned STL file for subsequent processing.
   (Figure 6. Flowchart of the orientation alignment process implemented in Python using Open3D.)



圖六 程式架構圖（自行繪製）

(b) Head Section Extraction (Cutting)

To isolate the nose or head geometry for independent aerodynamic analysis, a second Python script crops the head section based on user-defined AABB boundaries.

- The cropped mesh undergoes **Laplacian smoothing** (10 iterations) to remove surface noise and jagged edges, improving geometric continuity.

$$v_i^{(\text{new})} = v_i^{(\text{old})} + \lambda \sum_{j \in N(i)} \left(v_j - v_i^{(\text{old})} \right)$$

$\vec{v}_i^{\text{old}}$：頂點原始位置　　$\vec{v}_i^{\text{new}}$：頂點新位置　　$\vec{v}_j$：鄰近點位置

$\vec{v}_j - \vec{v}_i^{\text{old}}$：差直向量　　λ：平滑係數

- Vertex normals are recalculated, and the head section is saved as a new STL file for fitting.
  (Figure 7. Cropped head sections of the four biomimetic models after Laplacian smoothing.)

老鷹

```
# 設定頭部的範圍（根據 AABB 來微調）
min_bound = np.array([aabb.min_bound[0] + 220, aabb.min_bound[1] + 80, aabb.min_bound[2] + 550])
max_bound = np.array([aabb.max_bound[0] - 220, aabb.max_bound[1] - 20, aabb.max_bound[2]])

```

槍魚

```
# 設定頭部的範圍（根據 AABB 來微調）
min_bound = np.array([aabb.min_bound[0] + 220, aabb.min_bound[1] + 80, aabb.min_bound[2] + 700])
max_bound = np.array([aabb.max_bound[0] - 220, aabb.max_bound[1] - 20, aabb.max_bound[2]])
```

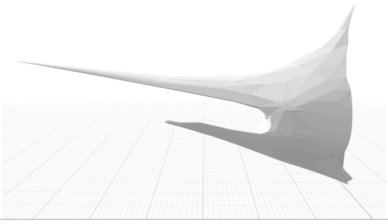
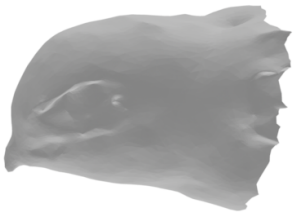
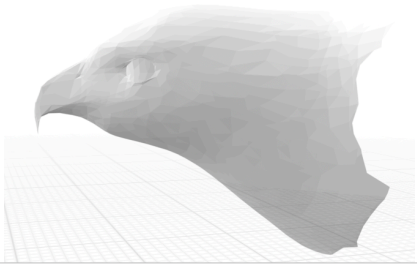
雨燕

```
# 設定頭部的範圍（根據 AABB 來微調）
min_bound = np.array([aabb.min_bound[0]+0.13, aabb.min_bound[1], aabb.min_bound[2] ])
max_bound = np.array([aabb.max_bound[0], aabb.max_bound[1] , aabb.max_bound[2]])
```

鯊魚

```
# 設定頭部的範圍（根據 AABB 來微調）
min_bound = np.array([aabb.min_bound[0] + 220, aabb.min_bound[1] , aabb.min_bound[2] + 320])
max_bound = np.array([aabb.max_bound[0] - 220, aabb.max_bound[1] - 20, aabb.max_bound[2]])
```

表二 模型頭部裁切參數

鯊魚	槍魚
	
雨燕	老鷹
	

表三 模型裁切後頭部

(3) Surface Fitting and Model Blending

Tools Used:

- **Open3D** – 3D data processing and visualization.
- **PyCharm** – Python development IDE.
- **Inventor / Fusion 360** – CAD modeling and parametric control.
- **Anaconda** – Environment and package management for Python.
- **Blender** – Freeform surface modeling, mesh refinement, and visualization.

Procedure:

1. Parametric Shape Blending

- Two STL models are used: a standard ogive nose cone and the biomimetic head section.
- Both are scaled to the same length along the Z-axis.
- Surface sampling is applied to convert them into point clouds with matched point counts.
- A blending parameter **α** (0.2, 0.4, 0.6, 0.8) is applied for linear interpolation between the two shapes:

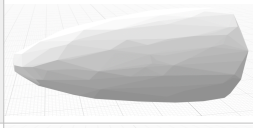
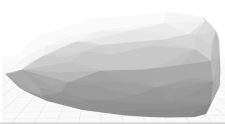
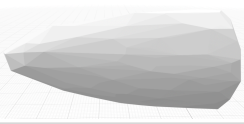
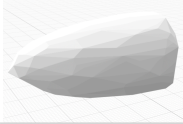
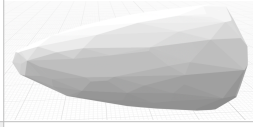
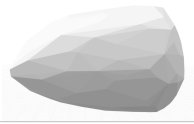
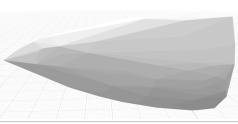
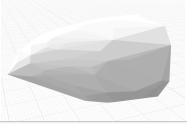
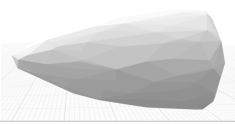
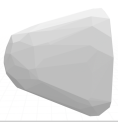
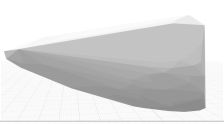
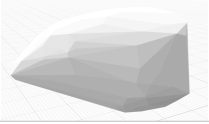
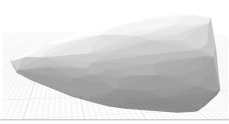
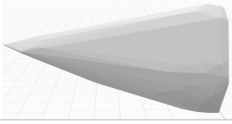
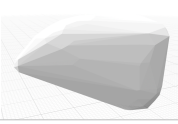
$$\vec{p}_{\text{blend}} = (1-\alpha)\cdot\vec{p}_{\text{standard}} + \alpha\cdot\vec{p}_{\text{bio}}$$

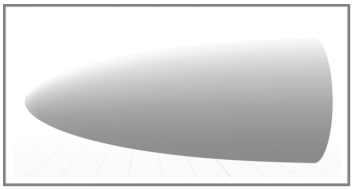
$\vec{p}_{\text{standard}}$：傳統鼻錐對應點位置　　$\vec{p}_{\text{bio}}$：仿生鼻錐模型對應點位置

α：混合比例參數　　$\vec{p}_{\text{blend}}$：內差後混合模型對應點位置

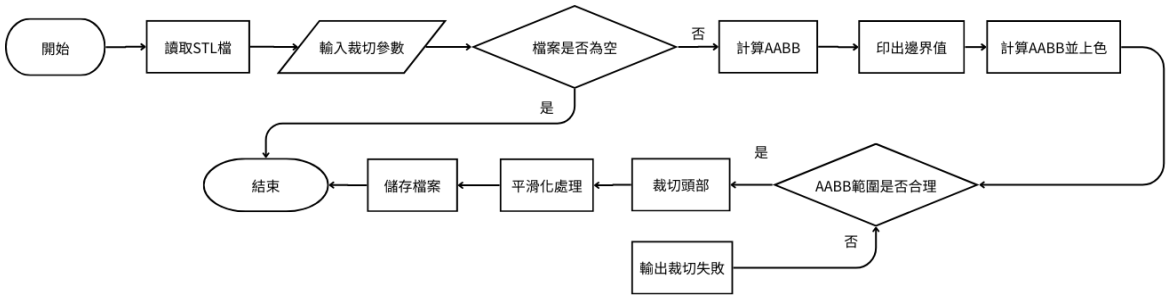
- The blended point cloud is reconstructed into a closed mesh using a convex hull algorithm.

(Figure 8. Examples of blended nose cone geometries for different α values.)

內差比/樣式	鯊魚	雨燕	槍魚	老鷹
0.2				
0.4				
0.6				
0.8				



表五 仿生鼻錐模型樣本



圖七 程式架構圖（自行繪製）

3.2 Simulation and Fabrication

(1) CFD Simulation

- **Software:** OpenFOAM v12 (solver), ParaView v5.13 (visualization).
- **Flow Conditions:** Mach 0.8, 1.0, 1.2.
- **Boundary Conditions:** No-slip walls for the nose cone surface; inlet and outlet set for steady-state airflow.
- Procedure:
    1. Import final STL into OpenFOAM.
    2. Generate computational mesh for the surrounding air domain.
    3. Run simulations to analyze pressure distribution, streamline patterns, and potential flow separation zones.

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \vec{u} + \vec{g}$$

ρ：流體密度　　$\vec{u}$：速率向量　　$\frac{\partial \vec{u}}{\partial t}$：局部加速度　　$\vec{u} \cdot \nabla \vec{u}$：對流加速度

$-\nabla p$：壓力梯度力　μ：動黏滯係數　　$\mu \nabla^2 \vec{u}$：黏性力項（內部摩擦力）

$\vec{f}$：體積力　　∇：向量微分算子　∇^2：拉普拉斯算子

4. Adjust nose cone curvature based on results to minimize drag and stabilize airflow.
   (Figure 10. Example of CFD pressure distribution and streamline visualization in ParaView.)

(2) 3D Printing

- Material: PLA.
- **Post-processing:** Sanding to remove print layer lines.
- **Integration:** Models mounted on 2-inch diameter PP tubes (wall thickness 4.5 mm) with custom fins.
- **Pressure Measurement Preparation:** Drill pressure ports at locations of significant pressure variation as indicated by CFD.
  (Figure 11. 3D-printed nose cone prototypes prepared for testing, with drilled pressure ports.)

3.3 Physical Testing

(1) Wind Tunnel Testing

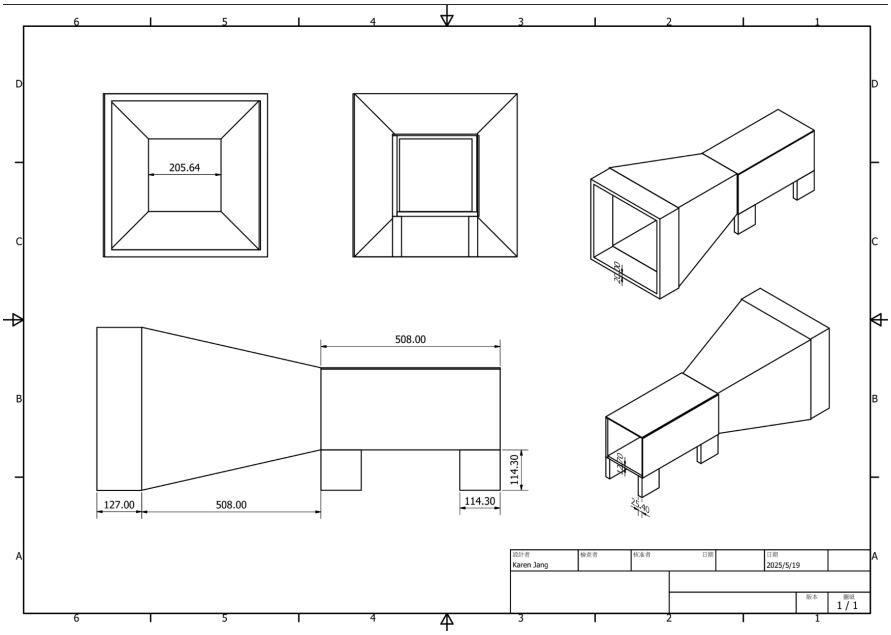
A three-section low-speed wind tunnel was designed and built:

1. **Drive Section:** Electric fan generates airflow.
2. **Flow Straightener Section:** Honeycomb straws and dual-layer nylon mesh reduce turbulence.
3. **Test Section:** Transparent acrylic for visual observation.

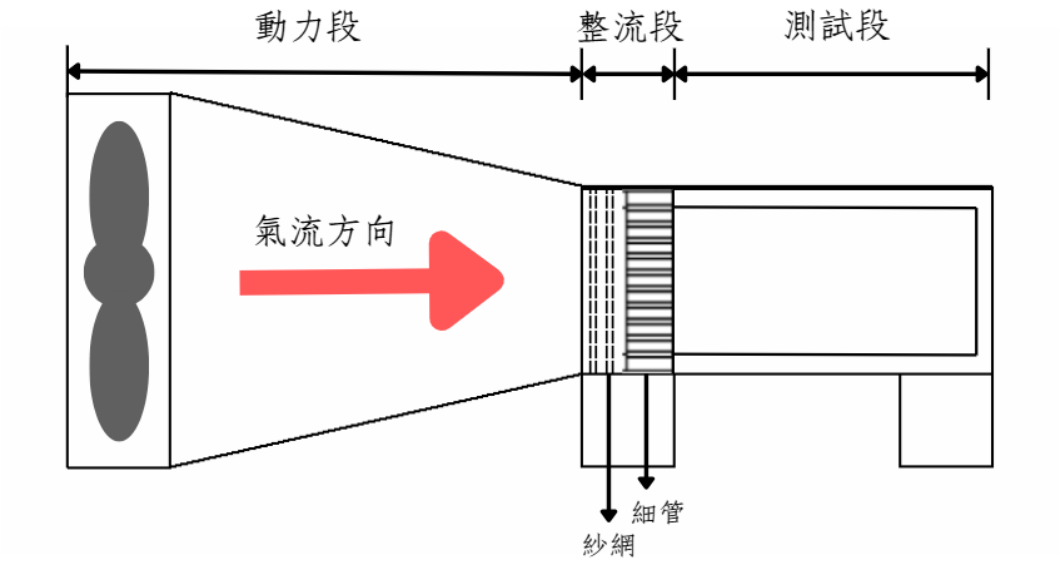
Visualization: Smoke filaments (incense sticks) used to observe flow attachment and separation.

(Figure 12. Custom-built low-speed wind tunnel with transparent test section.)

名稱	規格
壓克力板*3	厚4/長508/寬241.3（mm）
塑膠瓦愣版*5	A3
吸管*100	細吸管
紗網*1	高透尼龍60cm
電風扇	長400/寬400（mm）



圖九 風洞初步設計圖（自行繪製）



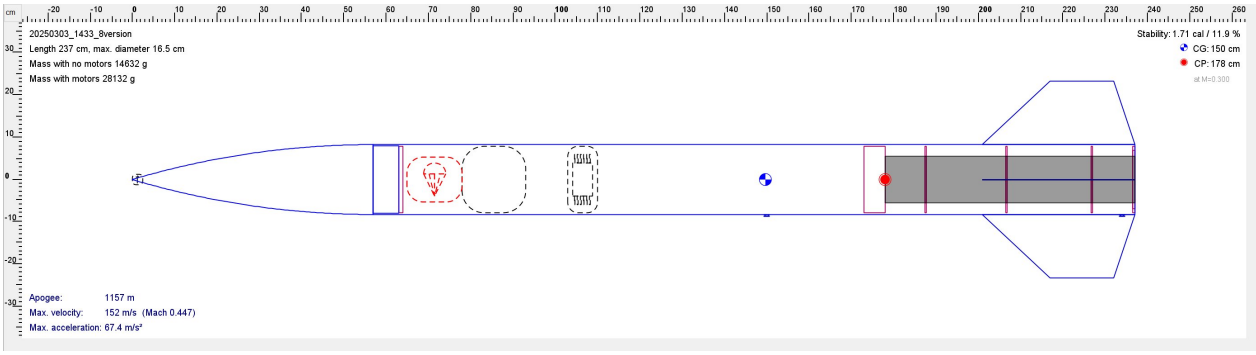
圖十 風洞運作示意圖（自行繪製）

(2) Surface Pressure Measurement

- Pressure ports connected to tubing and pressure sensors (BMP388) record real-time surface pressure under various flow speeds.
- Data compared with CFD predictions to validate simulation accuracy.
  (Figure 13. Pressure measurement setup with tubing and sensors connected to the nose cone model.)

(3) Launch Testing

- Conducted using commercially available fireworks rockets as propulsion (due to regulatory limits).
- Slow-motion video and altimeter data used to evaluate flight stability and altitude.
- Results compared with CFD and wind tunnel data to confirm aerodynamic performance.
  (Figure 14. Field launch testing of rocket prototypes equipped with biomimetic nose cones.)



圖八 openrocket設計圖

4. Future Work

4.1 Early Stage: Nose Cone Modeling and CFD Analysis

We will use **Fusion 360** to construct multiple nose cone designs, including a conventional ogive and four biomimetic variants (shark, sailfish, swallow, eagle).

Each biomimetic shape will be generated with four blending ratios (**α = 0.2, 0.4, 0.6, 0.8**), resulting in a total of **17 models**.

- All models will be 3D printed in **PLA**, post-processed by sanding, and drilled with pressure ports for wind tunnel testing.
- The models will then be paired with a full rocket body (including fins) and imported into **OpenFOAM** for CFD simulation at Mach 0.8, 1.0, and 1.2.
- Using **ParaView**, we will visualize streamlines and pressure concentration zones, then calculate drag coefficients (**Cd**) for comparative analysis.

4.2 Mid Stage: Wind Tunnel Testing, Data Collection, and Design Optimization

We will build a three-section wind tunnel (drive, flow straightener, and test sections) and use smoke visualization to evaluate flow attachment and separation on the nose cone surface.

- Pressure sensors will measure localized pressure at multiple points on the surface, and results will be compared against CFD predictions.
- If significant discrepancies are found, the geometry will be modified, reprinted, and re-tested to refine performance.

4.3 Final Stage: Flight Testing and Data Recording

The best-performing nose cone designs will be mounted on rockets for actual flight trials.

- Due to propulsion restrictions, commercially available fireworks rockets will be used.
- Flight trajectories will be recorded with slow-motion cameras and altimeters to assess stability and altitude.
- Results will be compared with both wind tunnel and CFD data to confirm aerodynamic improvements.

5. References

1. Chang, C.-H. (2015). *Linear regression and surface fitting*. Retrieved from http://mirlab.org/Jang/Books/Matlabprogramming4guru/10-2_regressionLin4surfaceFitting.asp
2. Wikipedia contributors. (2024, April). *Nose cone design*. In *Wikipedia*. Retrieved from https://en.wikipedia.org/wiki/Nose_cone_design
3. Nakka, R. (2023). *Nosecone Design Considerations*. Richard Nakka's Experimental Rocketry. Retrieved from https://www.nakka-rocketry.net/RD_nosecone.html
4. Off We Go Rocketry. (2011). *Nose Cone & Fin Optimization* [PDF]. Retrieved from https://www.offwegorocketry.com/userfiles/file/Nose%20Cone%20%26%20Fin%20Optimization.pdf
5. Crowell, R. (1996). *The Descriptive Geometry of Nose Cones* [PDF]. Universidade Federal do Paraná. Retrieved from https://servidor.demec.ufpr.br/CFD/bibliografia/aerodinamica/Crowell_1996.pdf
6. Barrowman, J. S. (1967). The practical calculation of the stability derivatives of an idealized slender body of revolution. NASA Technical Note D-3547.
7. Bechert, D. W., Bruse, M., & Hage, W. (2000). Experiments with three-dimensional riblets as an idealized model of shark skin. *Experiments in Fluids*, 28(5), 403–412. https://doi.org/10.1007/s003480050400
8. Wang, C., Zhang, J., & Wang, Y. (2016). Morphing design and optimization of aircraft structures. *Chinese Journal of Aeronautics*, 29(1), 1–12.

9. Ulu, E., McCann, J., & Kara, L. B. (2019). Structural design using Laplacian shells. *arXiv preprint arXiv:1906.10669*. Retrieved from https://arxiv.org/abs/1906.10669

10. ESDU. (n.d.). Normal-force-curve and pitching-moment-curve slopes of forebody-cylinder combinations at zero angle of attack for Mach numbers up to 5. Retrieved from https://www.esdu.com/cgi-bin/ps.pl?t=doc&p=esdu_89008c

11. Apogee Components. (2010). *Build Your Own Inexpensive Wind Tunnel*. Retrieved from https://www.apogeerockets.com/education/downloads/Newsletter252.pdf

12. NASA contributors. (2023). *Description of rocket parts*. NASA Glenn Research Center. Retrieved from https://www1.grc.nasa.gov/beginners-guide-to-aeronautics/rocket-parts/

13. 3Dscan.guide. (2018). *Inventor 3D scanning tutorial* [Video]. Retrieved from https://youtu.be/opBe9FafBHA

附錄

（一）turning 程式碼

```
import open3d as o3d


stl = o3d.io.read_triangle_mesh("swift_3d.stl")
if stl.is_empty():
    print("STL 檔案讀取失敗！")
    exit()


obb = stl.get_oriented_bounding_box()
stl.rotate(obb.R.T, center=obb.center)
# 重新計算 AABB
```

```
aabb = stl.get_axis_aligned_bounding_box()
aabb.color = (1, 0, 0)  # 紅色

stl.compute_vertex_normals()
#  顯示結果
o3d.visualization.draw_geometries([stl, obb, aabb], window_name="旋轉對齊後的 AABB")

o3d.io.write_triangle_mesh("aligned_swift_3d.stl", stl)
print("STL 已儲存")
```

（二）cutting 程式碼

```
import open3d as o3d
import numpy as np

# 讀取 STL 模型
stl = o3d.io.read_triangle_mesh("aligned_swift_3d.stl")

if stl.is_empty():
    print("STL 檔案讀取失敗！")
    exit()

# 計算 AABB (Axis-Aligned Bounding Box)
aabb = stl.get_axis_aligned_bounding_box()
aabb.color = (1, 0, 0)  # 設定邊界框顏色為紅色

# 打印 AABB 範圍 (確認整體尺寸)
print("AABB 最小值:", aabb.min_bound)
print("AABB 最大值:", aabb.max_bound)
```

```
# 設定頭部的範圍 (根據 AABB 來微調)
min_bound = np.array([aabb.min_bound[0]+0.13, aabb.min_bound[1], aabb.min_bound[2] ])
max_bound = np.array([aabb.max_bound[0], aabb.max_bound[1] , aabb.max_bound[2]])

# 創建頭部的 AABB
head_box = o3d.geometry.AxisAlignedBoundingBox(min_bound, max_bound)
head_box.color = (0, 1, 0)  # 設定頭部範圍為綠色

# 使用 crop() 來裁切頭部
head_part = stl.crop(head_box)

# 確保裁切後的結果不是空的
if head_part.is_empty():
    print("頭部裁切失敗，請檢查範圍設定！")
    exit()

# 平滑化處理
head_part = head_part.filter_smooth_laplacian(number_of_iterations=10)
head_part.compute_vertex_normals()

# 顯示裁切後的頭部
o3d.visualization.draw_geometries([head_part, head_box], window_name="老鷹頭部")

o3d.io.write_triangle_mesh("swiftdone_3d.stl", head_part)
print(“保存成功")
```

（三）fitting 程式碼

```
import numpy as np
```

```
import trimesh


# === Revolve 成鼻錐 mesh ===
def generate_revolved_mesh(z_vals, r_vals, segments=100):
    theta = np.linspace(0, 2 * np.pi, segments)
    vertices = []

    for z, r in zip(z_vals, r_vals):
        for t in theta:
            x = r * np.cos(t)
            y = r * np.sin(t)
            vertices.append([x, y, z])

    vertices = np.array(vertices)
    faces = []
    rings = len(z_vals)
    pts_per_ring = segments

    for i in range(rings - 1):
        for j in range(pts_per_ring):
            next_j = (j + 1) % pts_per_ring
            a = i * pts_per_ring + j
            b = i * pts_per_ring + next_j
            c = (i + 1) * pts_per_ring + j
            d = (i + 1) * pts_per_ring + next_j
            faces.append([a, b, d])
            faces.append([a, d, c])

    return trimesh.Trimesh(vertices=vertices, faces=faces)
```

```
# === 載入兩個 STL 模型 ===
nose_cone = trimesh.load("BNC-20B_Nose_Cone_No_Shoulder.stl")
falcon_cone = trimesh.load("Sailfishdone_3d.stl")

# === 自動將 falcon nose cone 縮放成與 BNC 同樣 Z 軸長度 ===
z_nose_min, z_nose_max = nose_cone.bounds[:, 2]
z_falcon_min, z_falcon_max = falcon_cone.bounds[:, 2]
length_nose = z_nose_max - z_nose_min
length_falcon = z_falcon_max - z_falcon_min
scale_factor = length_nose / length_falcon
falcon_cone.apply_scale(scale_factor)

#  ===  重取樣為相同點數  ===
def sample_mesh_uniform(mesh, n_points=10000):
    points, _ = trimesh.sample.sample_surface_even(mesh, n_points)
    if len(points) < n_points:
                #  隨機補足不足點數（可以重複）
                additional = points[np.random.choice(len(points), n_points - len(points),
replace=True)]
        points = np.vstack([points, additional])
    return points

n_samples = 10000
alpha = 0.2  # 0 = 完全鼻錐，1 = 完全老鷹
points_nose = sample_mesh_uniform(nose_cone, n_samples)
points_falcon = sample_mesh_uniform(falcon_cone, n_samples)

# === 對應點混合 ===
blended_points = (1 - alpha) * points_nose + alpha * points_falcon
```

```
# === 建立點雲，並使用凸包重建外型 ===
blended_pointcloud = trimesh.points.PointCloud(blended_points)
blended_mesh = blended_pointcloud.convex_hull

# === 匯出融合結果 ===
blended_mesh.export("Sailfishdone_3d_blended_0.2.stl")
print("融合鼻錐已輸出")
```