Shape Evolution in Computational Fluid Dynamics

Trainee Student: Cristian C. Rendón Cardona

Laboratorio de CAD CAM CAE
Universidad EAFIT

May 14, 2018

Course Coordinator
Prof. Dr. Mauricio Toro Bermúdez
ST0245 - Estructura Datos y Algoritmos
mtorobe@eafit.edu.co

Course Coordinator
Prof. Dr. Luis A. Mora Gutiérrez
IM0244 - Anteproyecto
lmora@eafit.edu.co

Research Supervisor
Prof. Dr. Oscar E. Ruiz Salguero
Laboratorio de CAD/CAM/CAE
oruiz@eafit.edu.co

Glossary

1. Ω: Rectangular orthogonal simulation domain with center $\in$ (0,0) and $x \in$ [-L,L] and $y \in$ [-D,D].
2. Γ: PL simple closed curve $\in R^2$ (CCW with respect to Z) with center of gravity $\in$ (0,0).
3. F_L: Lift force, generated by the difference of pressures across a object and acting perpendicular to flow direction.
4. F_D: Drag force, generated by the relative movement between a object and a fluid and acting parallel to flow direction.

Outline

1. Introduction

In nature, constant perturbations make objects change their shape in order to maximize a biological function and evolve. Thanks to actual computational power this shape evolution can be applied in engineering to optimize machines performance by maximizing a mathematical function.

Aeronautics industry is one of the field where shape optimization is widely used in the design of optimal Airfoils for specific functions. This project, whit help of Computational Fluid Dynamics (CFD), studies the shape evolution of a wing submitted to air flow maximizing the lift force to identify which parameters have more influence in the Airfoil shape.

Outline

2. Literature Review

1. Bernaulli theorem dictates that the decrease of pressure of a moving fluid is due to its speed increase [**?**].
2. Von Kármán effect is a repeating pattern of vortices caused by an unsteady separation of the flow [**?**].
3.

Outline

3.1. Problem Specification

Given:

1. A incompressible Newtonian fluid $\in R^2$.
2. Constant density and viscosity.
3. Steady laminar flow.
4. $v(x = \text{-L}) = (V_\infty, 0)$.
5. Γ_0: Initial object, submerged in the fluid domain Ω.
6. A target F_L.

Goal:

1. To obtain a Γ_f by modifying its shape such that generates the target F_L and minimizes F_D.

3.2. Evolution Diagram

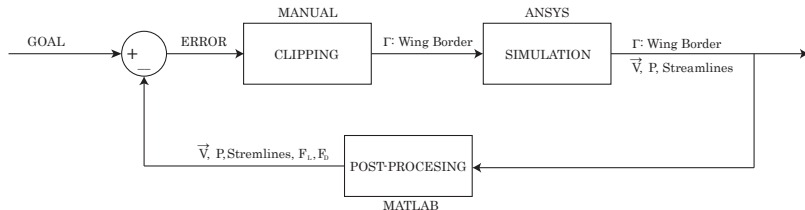Figure 1 shows the evolution diagram for the object's shape

Figure 1: Shape evolution diagram. The output is the optimized object Γ

3.3. Simulation Parameters

Geometry	
L	35 m
D	30 m
b	3 m
h	1.5 m

Table 1: Geometry for Ω and Γ_0

Mesh	
Max element size	0.3 m
Min element size	0.1 m

Table 2: Meshing parameters. Figure 3 shows the generated mesh.

FLUENT (Solver)	
Fluid	air
Density ρ	1.225 kg/m^3
Viscosity μ	1.789
Velocity V_∞	80 m/s
Flow model	Viscous Laminar
Constant density	
Constant viscosity	

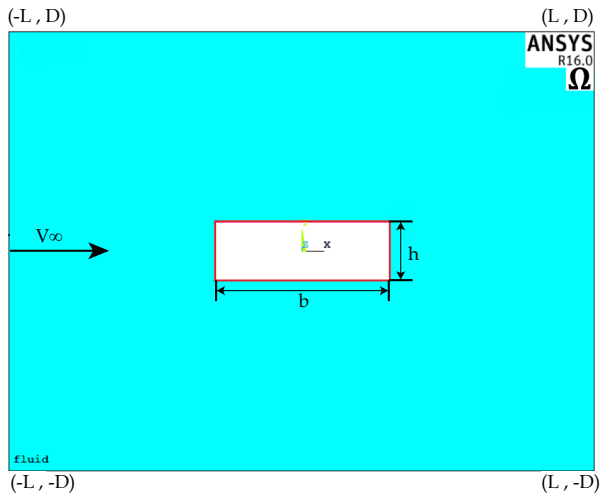Table 3: Solver model and boundary conditions

3.3. Geometry

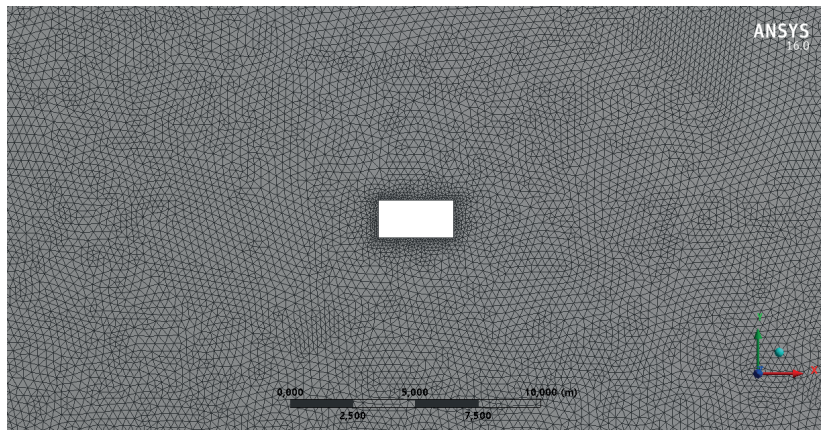Figure 2: Geometry for initial conditions

3.3. Mesh

Figure 3: Mesh for initial conditions

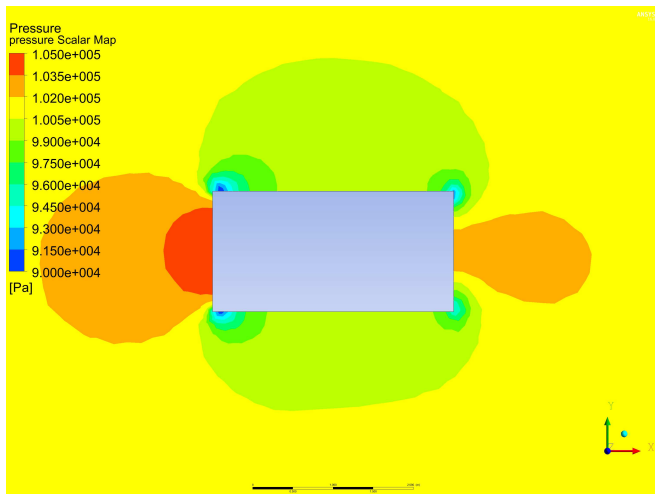# 3.4. Simulation Results

Figure 4: Pressure scalar map. Due to horizontal simmetry $F_L = 0$
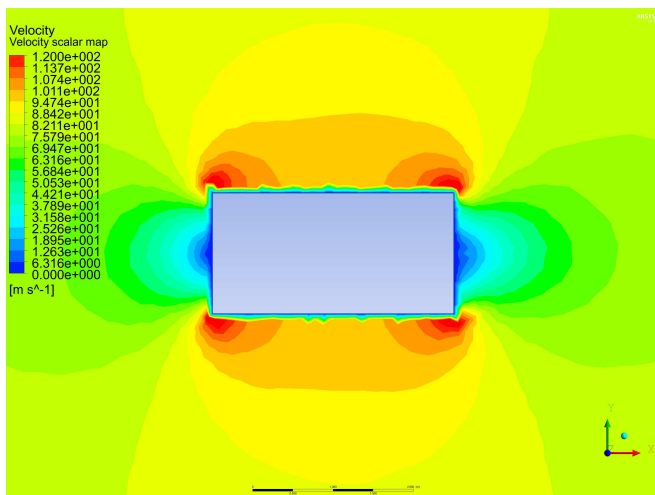
3.4. Simulation Results

Figure 5: Velocity magnitude scalar map arround Γ_0. There is low pressure for high velocity magnitudes and high pressure for low velocity magnitude
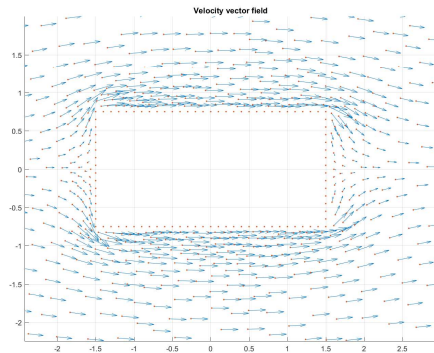
3.4. Simulation Results

Figure 6: Velocity vector field exported from ANSYS and ploted with Matlab

3.4. Simulation Results

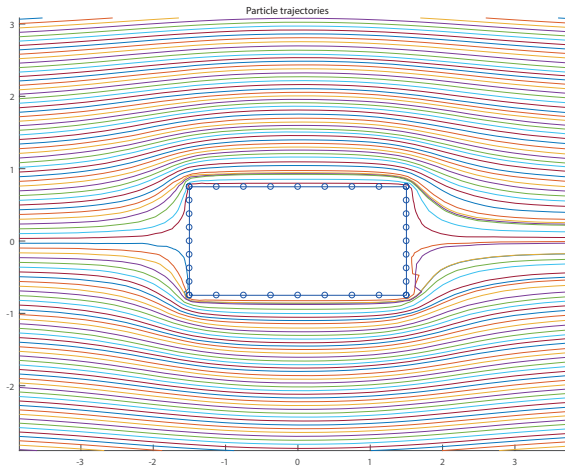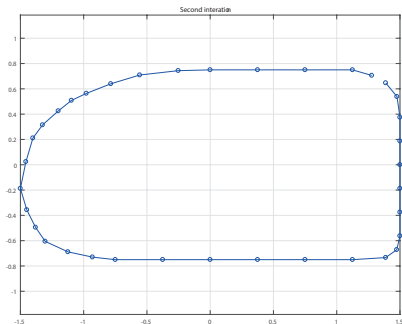Figure 7: Particle trajectory exported from ANSYS and ploted with Matlab. Changes in paths curvature produces change in velocity magnitude.
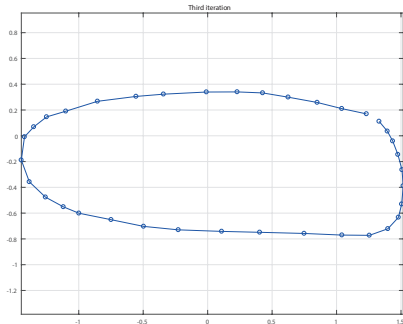
3.5. Wing Border Clipping

1. **Goal:** Satisfy a minimal Lift force F_L.
2. Tuning variable: Γ
3. **Criteria:** Produce a high pressure underneath the wing by reducing there the stream velocity. Avoid zero velocities around the wing
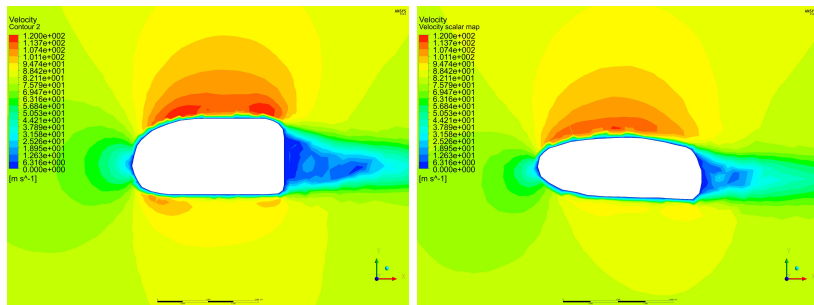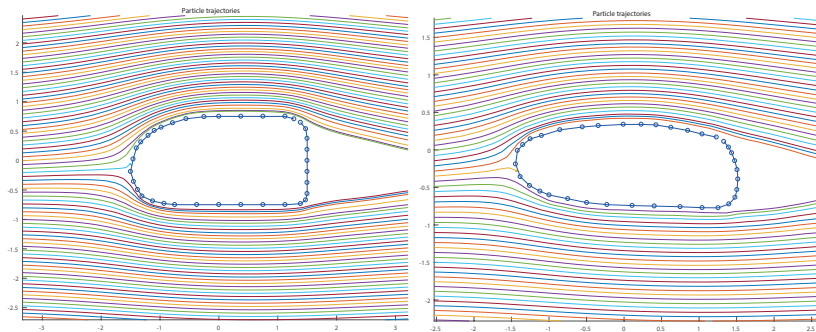

(a) Second state   (b) Third state

Figure 8: Manual pruning applied to Γ₀

3.5.1. Velocity Results

(a) Low velocities after Γ generate vac- (b) Γ tends to have a tail on order to
uum reduce the low velocity area
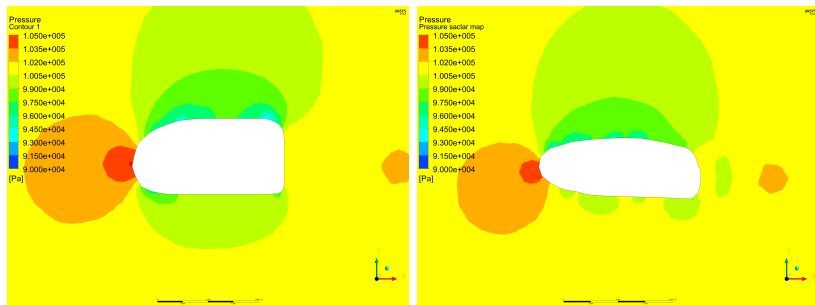
Figure 9: Velocity magnitude for second (a) and third (b) states

3.5.2. Particle trajectories

(a) Trajectories present high curvature values they meet with Γ

(b) Curvatures above the object are reduced in order to have a higher velocity than below it

Figure 10: Particle trajectories for second (a) and third (b) states

3.5.3. Pressure scalar map

(a) Pressure in front of Γ is higher than second state generating more drag

(b) Pressure difference is augmented by decreasing of velocity under Γ

Figure 11: Pressure scalar maps for second (a) and third (b) states

3.5.4. Final state

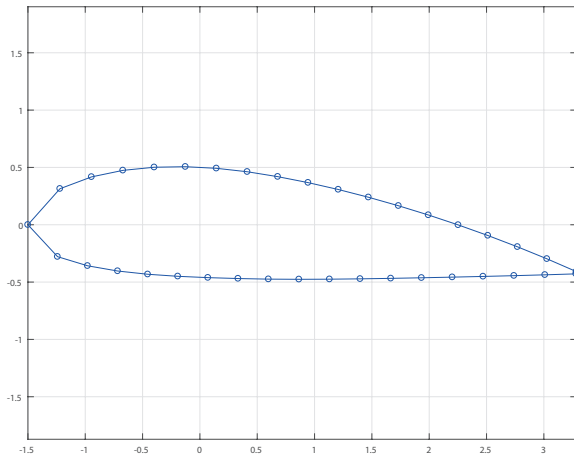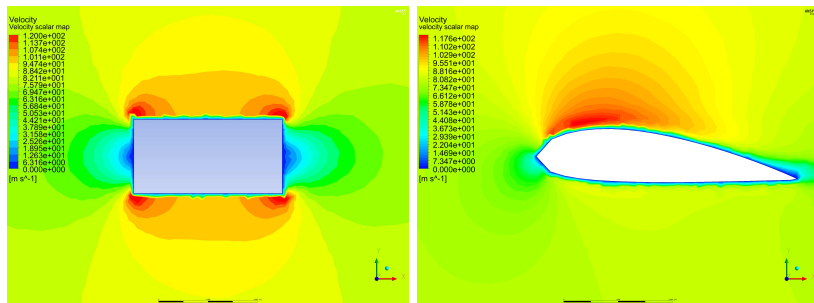Figure 12: Final state of the Pruning

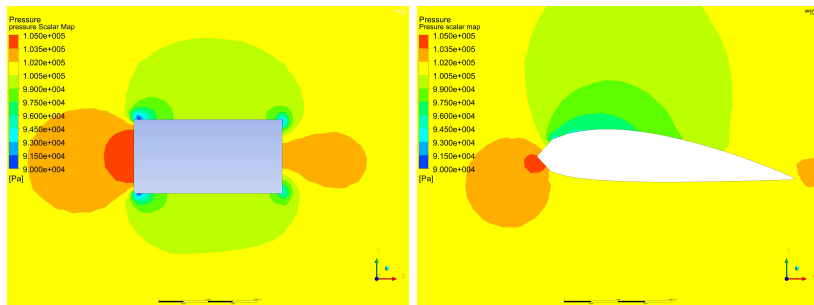# 3.5.5. Initial and final states

(a) Velocity presents horizontal symmetry

(b) There is no horizontal, therefore difference of pressure in Γ is not 0

Figure 13: Velocity magnitude for first (a) and final states

3.5.5. Initial and final states

(a) As shown in velocity scalar maps, Symmetry produces no lift

(b) In final state Γ presents higher pressure under it that produces lift

Figure 14: Pressure scalar maps for first (a) and final (b) states
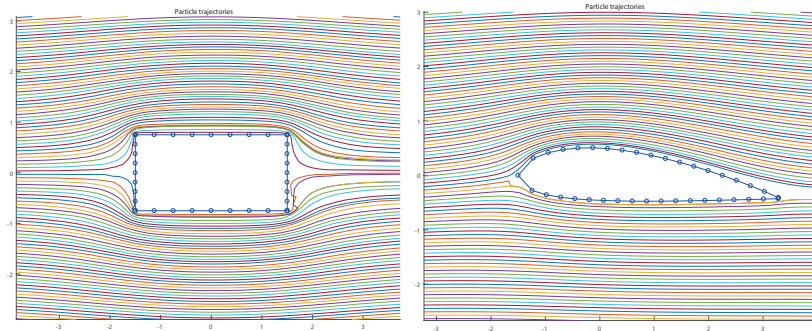
(a) Trajectories present high curvature (b) Curvatures are significantly reduced values they meet with Γ

Figure 15: Particle trajectories for first (a) and final (b) states

Outline

4.1. Matlab Data Structure

- ▶ Structure array of Matlab is a data type that groups related data containers called fields. This structure is used to storage the the obtained informations by the simulation.
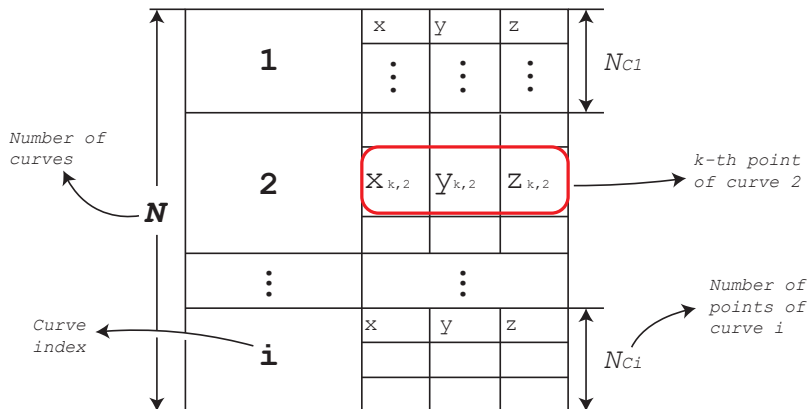

Figure 16: Matlab Struct Array for particle pathlines

4.2. Use of the structure

Being A the structure array that contains the streamlines information.

1. A(i).curve = number of the curve i of the structure A.
2. A(i).streamline(k,:) = k-th point of i-th streamline of structure A

4.3. Complexity calculation

1. To calculate the complexity of the algorithms it is necessary to traduce the measure variable to an input variable of the problem. In this case that variable is N_e the number of the mesh elements .

Assumptions

1. The number of streamlines is $O(\sqrt{N_e})$.
2. The number of points of the streamlines is $O(\sqrt{N_e})$.
3. n is the number of points of Γ
4. The number of elements is greater than the number of points of the polyline Γ.
5. Basic operations refers to Matlab functions and operations that are $O(1)$. (*e.g* sums, multiplications, normalizations, extract a value from an array, etc).
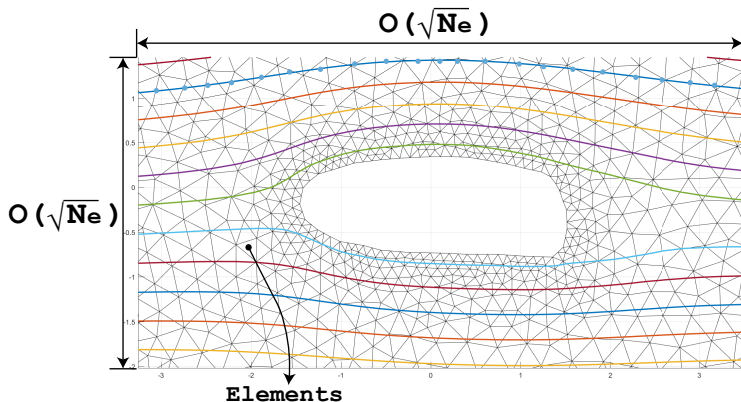
4.3. Complexity calculation

Figure 17: Relation between the number of elements, streamlines and points of the streamlines
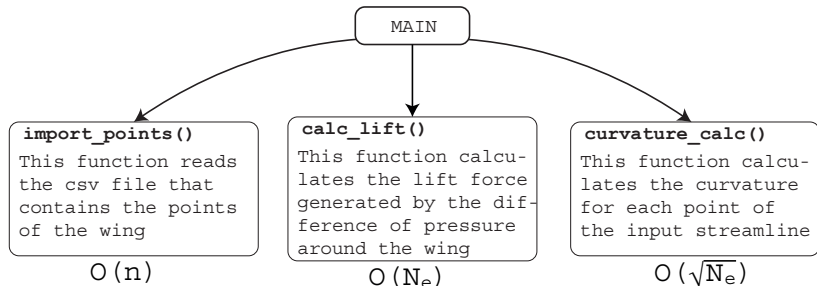
4.3. Complexity calculation

Figure 18: Functions called by the Main program

1. Matlab functions plot(), quiver(), load() and csvread() have $O(m)$ where m id the size of the input variable.

4.3.1. Curvature calculation of Streamlines

```
1  function curvature_calc(streamline)
2      for i = 2 to number of points - 1     O(√N_e)
3          basic operations                   O(1) * O(√N_e)
3      end for
4      return curvature                       O(1)
```

▶ Therefore the complexity of the function is

$$O(\sqrt{N_e} + \sqrt{N_e} + 1) = O(\sqrt{N_e}) \qquad (1)$$

4.3.2. Lift Force Calculation

```
1  function calc_lift(presure,gamma)
2     for i = 2 to number of points - 1    O(n)
3     basic operations                     O(1) * O(n)
4        for j = 1 number of elements      O(Nₑ) * O(n)
5           basic operations               O(Nₑ) * O(n)
6        end for
7     end for
8     return lift                          O(1)
```

▶ Therefore the complexity of the function is

$$O(n + N_e + N_e * n) = O(N_e, n)$$
$$N_e >> n \therefore \text{complexity is } O(N_e) \tag{2}$$

4.3.3. Main algorithm

```
1    import wing points                    O(n)
2    import velocity vector field          O(Nₑ)
3    import pressure information           O(Nₑ)
4    calc_lift()                           O(Nₑ)
5    plot velocity vectorfield             O(Nₑ)
6    plot wing border                      O(n)
7    import streamlines                    O(Nₑ)
8    find separations of streamlines       O(Nₑ)
9    for i = 1 to number of streamlines    O(√Nₑ)
10       plot i-th streamline              O(√Nₑ) * O(√Nₑ)
11       curvature_calc()                  O√Nₑ) * O(√Nₑ)
13   end for
```

▶ Therefore, the complexity of the function is

$$O(7N_e + n) = O(N_e, n)$$
$$N_e >> n \therefore \text{complexity is } O(N_e)$$

(3)

Outline

5. Conclusions

1. To reduce Drag force is necessary to allow smooth flow as possible in front of the wing. This can be made by modifying its frontal curvature.

2. The angle of attack is a very sensible parameter in the lift generation.

3. The wing tends to have a tail in order to avoid zero velocities after the it.

4. The calculation of lift force is the function that takes more execution time, followed by the functions that import the data.

References I