The Engineering Sketch Pad (ESP) Report

IISc

Abhigyan Roy

Undergraduate B. Tech Student, Department of Aerospace Engineering, Indian Institute of Technology Madras (IITM)

Monday July 17, 2023

Contents

Introduction

The current use of geometry models in design processes is ad hoc and often consists of various tools. Conceptual Design tools suggest the shape but do not realize 3D geometry, while early-stage specialized tools build 3D parametric models without considering the larger multidisciplinary design. Some disciplines remain "geometry free" during design, resulting in the final realized geometry being different from the analyzed.

A single, multidisciplinary geometry modelling system is needed to support complex geometric requirements in later stages. This system should generate geometries that do not need to be fixed or repaired, making it suitable for automation within the design process. The system should also output the designed geometry so that a commercial CAD system can import the model without loss of accuracy in the shapes represented. Relegating commercial CAD to the end of the design process may be surprising, but its focus has been, and is, manufacturing, not analysis. The current state of affairs is partly due to the focus on manufacturing rather than analysis. Attempting to use these systems within the early phases of design is partially responsible for the current state of affairs. These large, monolithic software systems are difficult to connect to and use as a service, and an effective geometry system for analysis needs to provide its output seamlessly to the rest of the design process without translation.

"Data" that current MDAO frameworks handle are "point" quantities (possible in "small" arrays) like geometric parameters (length, thickness, camber, etc), operating conditions (speed, load, etc) and performance value (cost, efficiency, range, etc). No current framework handles "field" data directly, i.e. copy (same as for "point" data), interpolate/evaluate, integrate and supply the derivative. Also, multi-disciplinary coupling in current frameworks requires that the user supplies custom pairwise coupling routines.

This report is on ESP and the CAPS framework which are softwares developed to solve the above issues.

1 Role of Geometry Models

Use in Meshing for Analysis

A gap exists between creating geometry and analyzing the geometry. This is due to the process of first creating the geometry, then transferring the geometry information from a CAD-based geometry to a meshing tool, and then finally to the appropriate analysis tools. The usual connection between fidelity analysis codes and geometry is performed by a grid generator. The 3D meshing software requires topological information to realise a closed "watertight" model for it to generate the appropriate surface and volume meshes.

Generation of design parameter sensitivity derivatives is required for gradient-based optimization. The individual instance of a Master-Model is the subject of analysis, while it is the Master-Model together with its parametric sensitivities that are the subject of optimization. Each instance of the Master-Model results in a BRep (Master-Model implies feature tree).
Boundary Representations (BReps) are the standard data model that holds both the geometric and topological entities that support the concept of a "solid". The BRep includes the model's topology which collects the geometric entities into their topological equivalent and provides the connectivity information. The topology is directly related to both the design intent of the Feature Tree and the construction methods of the underlying CAD system.
These BReps have a tolerance that determines the meaning of "closure" for connected entities. By tolerance of closure of connected objects, it means that the lower dimension entity is within a certain range of the higher dimension entity so that the face(model) created sits on the surface of the geometry. Usually the precision in the model is higher than tolerance, so either needs to be 'fixed' . This is for some part due to unintended geometry construction artefacts that plague BReps commonly manipulated through Mechanical Computer-Aided Design (MCAD) systems.

To deal with gaps and overlaps without a program halt which then requires intervention, we can either "fix" the geometry or the BReps created. Currently, most applications (being BRep-based) "fix" the geometry by representing the geometry definition in a different manner. This causes various side effects like inconsistencies, errors, complexity and this process itself is not automatic, hence it needs to be done manually and thus, is time consuming. Essentially, this process typically breaks the relation between the discrete representation and the original geometry model.

Creation and Collaboration

Commercial shape control systems, both CAD-based and CAD-free, have drawbacks. Commercial CAD systems may hide core geometric components, making it difficult to compute parametric sensitivities. Customization can be difficult, and the number of available licenses can be a practical issue. CAD-free systems have limited intuitive shape design parameters, requiring time-consuming manipulation.
Free form parameterization methods can streamline optimization but may hinder design engineers' understanding of results being purely mathematical. B-splines can bridge parametric solid modeling with free form shape design.

Aerodynamic shape optimization has been a subject of extensive research, with high-fidelity techniques and tools ranging from simple airfoil geometries to complex components. However, these tools have not been widely adopted among designers due to the required depth of knowledge among various tool-sets. Computational Fluid Dynamics (CFD) steps in solving shape optimization problems include geometry parameterization, surface and volume meshing, flow analysis, and optimization itself. This requires extensive knowledge of the design problem and setup, especially in optimization problems with larger sets of design variables. Automation of this process offers some relief and decreases manual setup times. However, integrating all the disparate codes involved in each step is challenging. Historically, the wide array of codes involved in shape optimization stems from the geometric modelers used, with custom in-house discrete mesh-based tools being common in the aerospace design community. Commercial CAD systems may hide core geometric components, making it difficult to compute values like parametric sensitivities.

ESP provides an integrated design environment which is a powerful tool that allows for collaboration among team members during the design process. Traditionally, design has been a solitary activity, with each engineer working independently on their part of the project. However, as designs become more complex, it has become increasingly important for engineers to work together to ensure that the final product is of necessary specifications. It is particularly useful for models that evolve over time due to changing requirements. The integrated design environment allows for real-time collaboration, enabling team members to work together on the same project simultaneously. This approach has several benefits, including shared ownership of the model, fewer shortcuts, and a lower error rate. In this way, the integrated design environment is a valuable tool for collaborative modeling and design.

2 A CAD based Approach

Processes in Current CAD systems

The traditional design process starts from a conception stage where no actual geometry may be specified, to a final design where the part is fully realized down to the finest details. When working in a multidisciplinary design environment, one discipline may establish some parameters before transferring its knowledge to another. The design won't be fully developed until a more thorough examination is necessary and requires equivalent geometric qualities.

The design process in CAD systems is consistent. Each phase in the design process uses the same suite (or a subset of the suite) of parameters.The design can be monitored and maintained using the CAD system and Product Data Management (PDM) or Product Lifetime Management (PLM) software associated with it. Additionally, since the design is in the CAD system from the start, manufacturing-related concerns can be easily addressed early on and unrealistic expressions are prevented from entering the design space.
With respect to MDAO, defeaturing for design progression involves adjusting the Feature Tree to represent the design process stages. During preliminary design, most branches are suppressed, but as the design progresses, more details are expressed by unsuppressing them. This approach helps match geometry fidelity to analysis, making it simpler and more rigorous than manually modifying fully expressed parts.

Traditional serial design settings, where one discipline performs its design and passes the results to the next, lacks the ability to easily recover from a conflict between the current discipline and the state of the geometry. This usually requires restarting the entire process. For example, aerodynamics designs a wing that does not contain enough space to support the structures. With an integrated continuous view of design and the appropriate parameterization, the shape can be modified at any time so that the aerodynamics team does not need to be explicitly involved in redesigning the wing.

CAD systems are large scale complex software environments. Not only are the software licenses expensive, but the training costs for the learning of parametric CAD are often far greater. There is little training for using Parametric CAD in MDAO.
Not all parametric CAD systems are created equal, and the suite of features differ for each. It requires a great deal of specific CAD training to be able to compose the series of operations driven by the parameter set to form a particular design. In some instances it may not be possible to fully realize the desired parameterization due to a feature mismatch or a fundamental difference in construction or modification of geometry.
The various analysis methods that make up the geometry definition can be found across numerous disciplines and at various levels of accuracy within a discipline. For instance, the Panel approach for intermediate analysis requires the full outer-mold line (OML), whereas the Vortex-Lattice method only needs the surface planform outlines and airfoil camber lines. The redesigned geometry alterations based on one technique are likely to conflict with the definitions of the other geometries, which ultimately requires human intervention in a typically ad hoc way. If a 3D model incorporates a low and high-fidelity parameterization that is consistent with needed geometry requirements, then higher fidelity analysis becomes possible alongside low-fidelity tools.
The geometric sensitivity derivatives are elusive when the geometry is defined by complex and often proprietary CAD software. With proprietary systems, software source code is unavailable and therefore differentiation of the source is not possible.

Specialized construction is hindered as the standard geometry file formats do not support these unusual geometry types. The lofting procedures needed to generate a wing in a particular way may not exist in a particular CAD system. So to perform specialized constructions it would be necessary to convert (by additional fitting or approximating) to types such as BSplines or NURBS (Non-Uniform Rational B-Splines) for transmittal to other parts of the MDAO process.
A fully defined system has the proper number of constraints and is solvable, as in no singularities exist, resulting in one unambiguous solution for the sketch. An over-constrained system has too many constraints and those constraints may or may not properly define a sketch. An under-constrained system does not have enough constraints to uniquely define the sketch and constraints must be added. An improperly constrained system has the correct number of constraints but there may be conflicting constraints that cause a singularity, and certain coordinates to be unsolvable. Over-constrained sketches are the only state that all commercial CAD systems tested acknowledge and offer some advice to repair.

The concept of parameters was added to “CAD” systems, making it possible to create a model whose dimensions could be easily modified. “Computer-Aided Design” systems took the perspective of adding,

removing, or morphing some shape, users of the systems are encouraged to "make your models mimic the manufacturing process". If this is done, the generation of manufacturing processes can be automated, hence we classify these "CAD" systems as "mCAD" systems. But for the (optimal) design of complex systems, the analytical designer wants to think about the function and performance of the device being generated, requiring the generation of a separate "aCAD" model. Sometimes the systems that produce analysis-ready models are thought of as "pre-preprocessing", referring to the order utilized in the larger design/manufacturing process ("aCAD" before "mCAD"). Almost all "aCAD" systems are (and have always been) parametric but "mCAD" has only recently become parametric. The modelling techniques supported by "aCAD" and "mCAD" are rather dissimilar, and so transfer between them is done by limited translators or by starting over. This one-way path from "aCAD" to "mCAD" leads to a design process where real feedback is not really possible.

File Formats

Each CAD system or geometry kernel uses a different mathematical formulation to represent the same types of surfaces, and also have different tolerances for closure. Therefore, while transferring data it may be found that the model may be open now again and need patching.

- IGES file format contains data that is defined as disjoint and unconnected surfaces and curves, with no explicit notion of topology. This is not suitable for meshing since we need such topological information to realise a closed "watertight" model for meshing.
- The STEP file format supports topology as well as geometry. This is therefore the preferable file type to use for the transmission of closed models. This format is seldom used in practice as constructing a STEP reader is complex and requires a complete solid modelling geometry kernel to deal with the data.
- STL combines a discretized view of the Brep as well as its geometry and topology to provide a complete, and easier to use, access point. [1]
- EGADS
- NMB file type supports database import in the proprietary format of Fidelity Pointwise's geometry kernel and can be used to exchange database information with it in the kernel's grid-neutral format.

3 Objectives

- – ***To support the entire Multi-Disciplinary Analysis and Optimization (MDAO) process from conceptual to detail design in a seamless manner, as well as multi fidelity analysis***

  – ***To make the process user-friendly and support automation to minimise unnecessary or repetitive human effort.***

Current methods to achieve these objectives is Constructive Solid Geometry (CSG) using Conceptual Design geometry tools(CDGT) are the natural foundation on which the modelling techniques are based, allowing flexibility in creating complex structures from simple components. CSG is based on a master model format in which objects can be combined through boolean operations. The build recipe is stored as a Feature tree (a binary tree). CDGT build up individual 3D components when sizing geometry and are parametric, allowing for rapid and interactive control of shape.
Next, a combination of 2 approaches is used - CAD systems and their "feature" based view of construction,

[1] Further research needed as to why this not used in the MDAO process

and Bottom-Up methods which generate solid "components". Bottom-Up methods do not have the turn-key features of commercial CAD systems, but it provides geometric design-gradient information is required for optimization, giving it flexibility and potential open nature.
To realise the MDAO objectives via the Bottom-Up approach, a new software suite, the Electronic Geometry Aircraft Design System (EGADS) which has been developed is used on top of OpenCASCADE, the solid modelling geometry kernel. To further enhance the abilities by enabling creation or modification externally by other programs, to be extensible via user-defined primitives and directly provides sensitivities, OpenCSM, an open source software is built on EGADS. Finally, Engineering Sketch Pad (ESP), an open source, MDAO oriented, collaborative, fully-parametric, feature-based solid-modelling system that is web-enabled making it interactive is used. It supports different disciplines such as both structural and aerodynamic analyses from the same geometry definition.

4 Parts of ESP Architecture

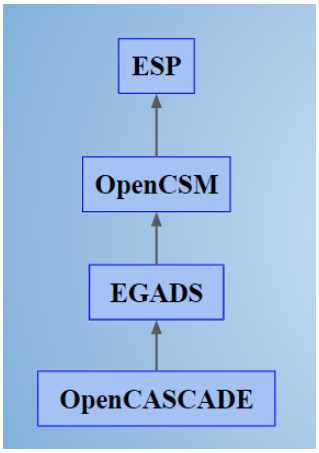

Figure 1: Layers of ESP

4.1 OpenCASCADE - geometry primitives, BRep model

OpenCASCADE is a fully functional open-source solid modeling geometry kernel that can be used to create and manipulate 3D models. It has the ability to perform bottom-up construction, supports manifold and non-manifold geometry, and has both CSG operations and other abstract feature-like construction methods, unlike drafting mode in early CAD softwares. OpenCASCADE can read and write IGES, STEP, and native file formats, and is a fully object-oriented C++ API with about 17,000 methods in 2+ million lines of code. It is often used as a basis for developing software applications that require solid modeling capabilities.

4.2 EGADS - persistent attribution, watertight tessellation

EGADS (the Electronic Geometry Aircraft Design System) is a software subsystem built on OpenCASACDE, reducing the level of programming complexity and its the huge suite of methods. This procedural-based API is an open source, object-based software built on top of OpenCASCADE, thus it has CSG operations and Bottoms up construction. It has full support for current platforms (hardware and software) and for maximum flexibility can be driven by either C, C++, or FORTRAN user applications (for other types, those architectures can be ported and used). Also it outputs STEP file, other than its native EGADS format (no loss of geometric data or fidelity but the design intent is lost) so design can be used in other CAD systems. The scope of an attribute depends on the class of the owning object. Only those attributes on Topology will

be persistent across EGADS sessions and this will only take place if the Model data is written out in the native EGADS manner.

4.3 OpenCSM - design parameters, feature tree, UDP, sensitivities

OpenCSM is built on EGADS. The parameters created, each having unique names are all stored as two-dimensional arrays of floating point numbers; these parameters can either be external or internal. External parameters are those that are exposed to the outside world and can be modified programmatically in OpenCSM via external calls to OpenCSM's API. The values of these external parameters are always specified as numbers. Internal parameters are those used during the execution of a model. In a sense, internal parameters can be thought of as multi-valued temporary variables.
As such, internal parameters can either be specified as a number or as a MATLAB-like expression based upon the current values of external and other internal parameters; these specifications can only be done by altering the feature tree. One unique feature about external parameters in OpenCSM is that, in addition to a value, *external parameters can have a "dot" associated with them which is used in the calculation of sensitivities.*
Feature trees are defined in terms of a binary-like tree of branches. Each branch has an associated type which describes the operation to be performed.Branches can have zero to two parents. Parents in branch tress - primitive solids have no parents since they start the branch of a tree; transformation branches (such as rotatex) have one parent while Boolean operations (such as intersect) have two parents. For each parent link, there is a reciprocal child link. Any branch without a child link corresponds to a solid body (orBRep) that is the ultimate product of the build operation in OpenCSM. The API of OpenCSM currently consists of 31 functions that are callable from C. While not strictly object-oriented, the API has been designed in an object-based manner, making it directly interfacable with object-oriented systems written in Python, Tcl/Tk, C++, or Java.

4.4 ESP

The Engineering Sketch Pad is the final piece of this architecture to build the geometry that utilises the abilities of all these elements to their maximum, acting as the system driver enabling smoother user interaction with the graphic objects.

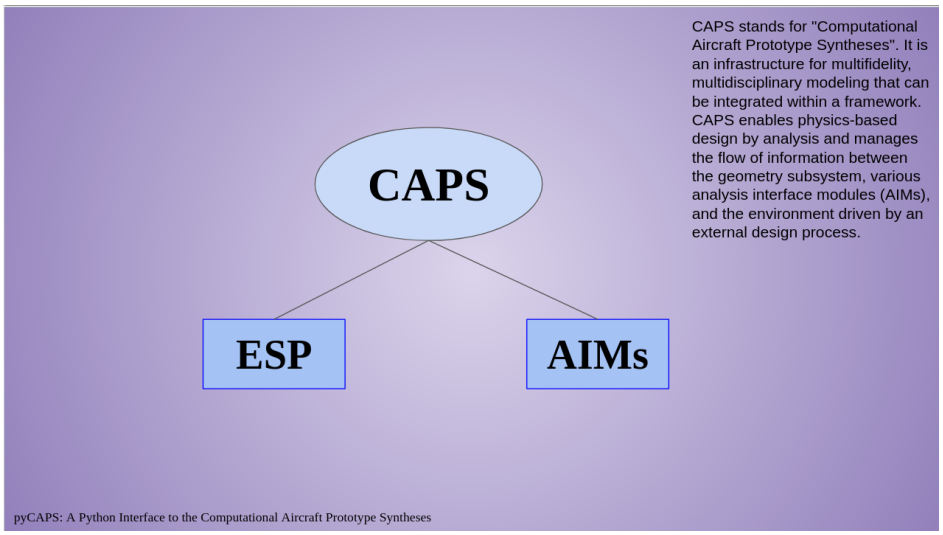

Figure 2: ESP in CAPS Framework

4.5 CAPS

The Computational Aircraft Prototype Syntheses (CAPS), which combines geometry, meshing, and analysis model generation into a single context, has the ability to support multi-disciplinary, multi-fidelity analysis from a single geometric source for the design of aerospace vehicles. To balance solution accuracy and cost, several modelling considerations related to geometric representation, physics modelling, interdisciplinary coupling, and numerical error control must be made from the standpoint of computer simulations.

5 Engineering Sketch Pad (ESP)

ESP is a browser-based system at the top of the architecture which provides the user the ability to interact with a configuration by building and modifying the design parameters and feature tree that define the configuration. It captures the design intent in a simple, easy to read, nonproprietary text file (ASCII) that is easily modifiable by any other component in an MDAO environment, viewed in a web-based GUI. It is open-source and is not encumbered with any licensing restrictions.

The traditional approach to engineering design splits the workflow into sequential stages of conceptual, preliminary and detailed design. Design is not a single-pass process, but frequently involves exploration of alternatives, some of which might be abandoned, but all of which should be documented in some way. Initially, conceptual design translates a set of design requirements into simple configurations that are sized and evaluated at high level using engineering intuition, empirical models and low fidelity analysis. As the design progresses into later stages, the focus shifts to refining the geometry and increasing fidelity analysis. One way to approach the design complexity problem is to view the design process as a decision tree where each node represents the work necessary to answer a specific design question. This view breaks the overall design process into manageable chunks. This process allows the designer to trace design decisions up to specific nodes and progress the design from any node in the tree, enable non-sequential design workflows. ESP supports the generation of multiple models from the same parameterization. For example, one can easily generate a beam, built-up element, or fully expressed solid model (or assembly) of an aircraft structural model. Simultaneously, one can use the same parameters to generate mid-surface aerodynamic models as well as full solid models for CFD analysis. It also provides attribution at all levels, which is an essential capability when one wants to connect (in a multi-disciplinary way) the various parts of the various representations. Each node in the decision tree is a Phase in ESP. Each Phase can be driven by any workflow which allows each Phase to be streamlined in answering the design question of interest. A Phase is essentially defined by two input files: a CSM file that captures the parametric geometry variables, and a pyCAPS script that notes the intent of the current Phase, constructs and solves the optimization problem, and updates the geometry with the optimal values.

ESP allows for a collaborative environment as in the MDAO process, it is near impossible for one individual to understand all the intricacies and tactic assumptions in the design and implementation of the code. In pair programming, one individual (typically called the driver) sits at the keyboard and actually types the program, that is, makes the tactical decisions as to the naming of variables and the exact structure of the statement in the program. The other individual (typically called the navigator) sits and watches the overall process, typically thinking at a higher (strategic) level about whether or not the whole approach is going to work, about situations that might cause the program to fail, or ways of simplifying the approach. After doing this for a short period of time (a few hours), the individuals switch roles, which has been found to be extremely important to keep the pair functioning at a high level. Patterned after the pair-programming paradigm, control of ESP can easily be switched to another user by "passing the ball". Several benefits include shared-ownership of the model, boosts efficiency through collaboration, knowledge is shared, a tendency to take fewer short-cuts, and a lower error rate.

ESP supports easy integration into a larger process by allowing geometry import/export using file stan-

dards (either discrete or analytic) and direct connections to a number of 3D grid generators, preventing loss of critical information when the standard does not support the data in the current available systems. It generates watertight models which are imperative for 3D mesh generators to carry out its functionality efficient

It provides analytic parameter sensitivity (for much of the build), making it suitable for the gradient-based optimization processes that are frequently used in MDAO environments. The sensitivity of any part of a configuration with respect to any design parameter.

ESP has the ability to easily add custom features to the system in the form of UDPs. Thus it allows for parameterised components to be generated by the user of unique shapes. For example, it is quite difficult to generate something simple like a NACA 4−digit wing in a commercial CAD system. With the OpenCSM user- defined primitive, this only requires a small amount of bottom-up EGADS programming.

6 Computational Aircraft Prototype Syntheses (CAPS)

Computational Aircraft Prototype Syntheses is an MDAO framework augmenting MDA with richer geometric information via OpenCSM, enhance automation by tightly coupling analysis with geometry and allow interdisciplinary analysis with "field" data transfer while also not replacing optimization algorithms. It provides the tools and techniques for generalizing analysis coupling like in multidisciplinary coupling (aeroelastic, FSI) and multi-fidelity coupling (conceptual and preliminary design). It provides the tools and techniques for rigorously dealing with geometry (single and multi-fidelity) in a design process where OpenCSM connects design parameters to geometry and CAPS itself connects geometry to analysis tools. It also aims to input and attribution driven automated (not automatic) meshing.
ESP is setup with CAPS and visualizes bodies used by CAPS but cannot change parameters or attributes (only their values), surface meshing AIMs and data transfer setup. (Volume mesh visualization is not supported in ESP, instead Paraview can be used)

Analysis Interface Module (AIM)

AIMs are used to interface between CAPS framework and analysis tools. It hides all of the individual analysis details (and peculiarities) but does not make analysis tool a "black box". It consists of shared libraries written in C/C++ which are loaded at runtime as plugins. It defines analysis input parameters and outputs, where the inputs include attributed BRep with geometric-based information.
Also AIMs inputs/outputs can be linked to transfer simple or rich data (e.g. meshes) between AIMs, , essentially creating a data flow. The user defines bounds on geometry to connect 'field' data, which AIMs instances "field" are coupled and iteration loops in the code (Python script). The AIM Developer functions to interpolate and/or integrate discrete data (consistent with solver) and to reverse differentiated interpolate and integrate to facilitate conservative transfer optimization. The CAPS Framework performs the "field" data transfer (interpolate or conservative) which is automatically initiated in a lazy manner.

Low Fidelity	Structural Analysis	Meshing	3D CFD
- AWAVE - FRICTION - AVL - XFoil	- Masstran - MYSTRAN - NASTRAN - ASTROS - TACS	Surface - Native EGADS - AFLR4 Volume - TetGen - AFLR3 - Pointwise	- Cart3D - Fun3D - SU2

7 The Entire MDAO Framework

In the Multidisciplinary Analysis and Optimization (MDAO) framework, the user has complete control over execution. The process is simple and straightforward. Load Geometry - Create AIM - Set Geometry Parameter - Set Analysis Parameter - Execute Analysis - Retrieve Analysis Outputs In such a way, an entire database can be constructed, by performing analysis on each analysis parameter for each geometry parameter.

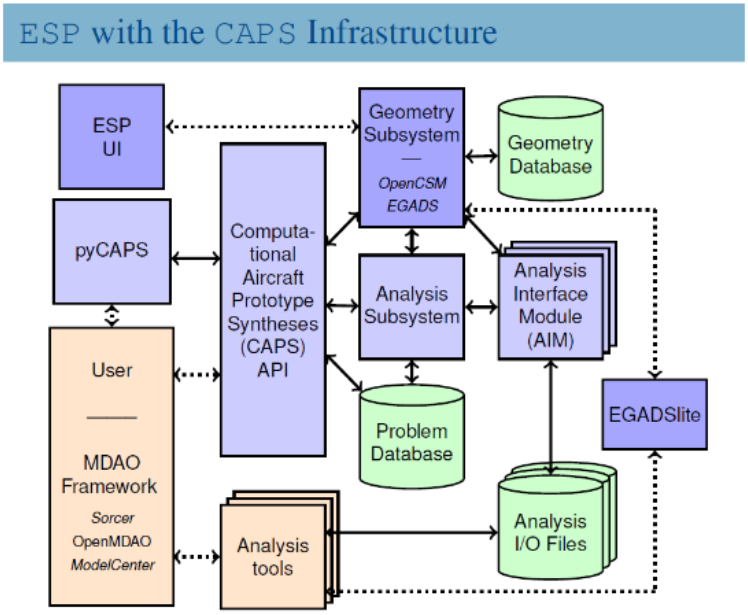

Figure 3: ESP with CAPS Framework

7.1 Demonstration of the ESP Integrated Environment

In this section, the designing of a wing is shown using ESP in the CAPS framework. Starting from conceptual design to preliminary structure and then to the final flow analysis of the wing, usage of selected softwares from the various available AIMs is shown.

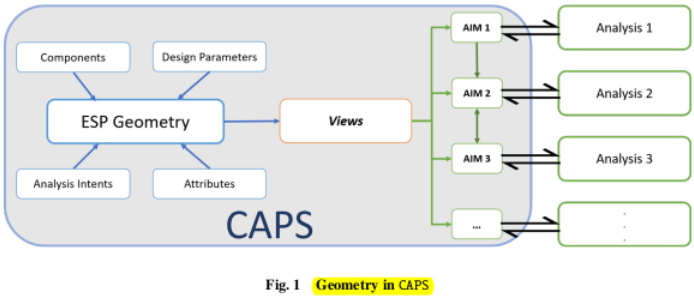

Figure 4: Workflow of the example case

Geometry Creation using Low Fidelity Analysis tools

7.1.1 Airfoil Section using XFoil

XFOIL is an interactive program for the design and analysis of subsonic isolated airfoils. It consists of a collection of menu-driven routines. The main goal was to combine the speed and accuracy of high-order panel methods with the new fully-coupled viscous/inviscid interaction method used in the ISES code.
XFOIL uses a linear-vorticity stream function panel method for predicting airfoils with different angles of attack. It uses a Karman-Tsien compressibility correction and has two types of inverse methods: Full-Inverse and Mixed-Inverse. Viscous formulation describes boundary layers and wake with a two-equation lagged dissipation integral BL formulation and envelope e^n transition criteria.

Using Xfoil, a naca airfoil is generated of appropriate characteristics based on Cl/Cd ratio from low fidelity analysis.

Firstly, a general naca airfoil is created.

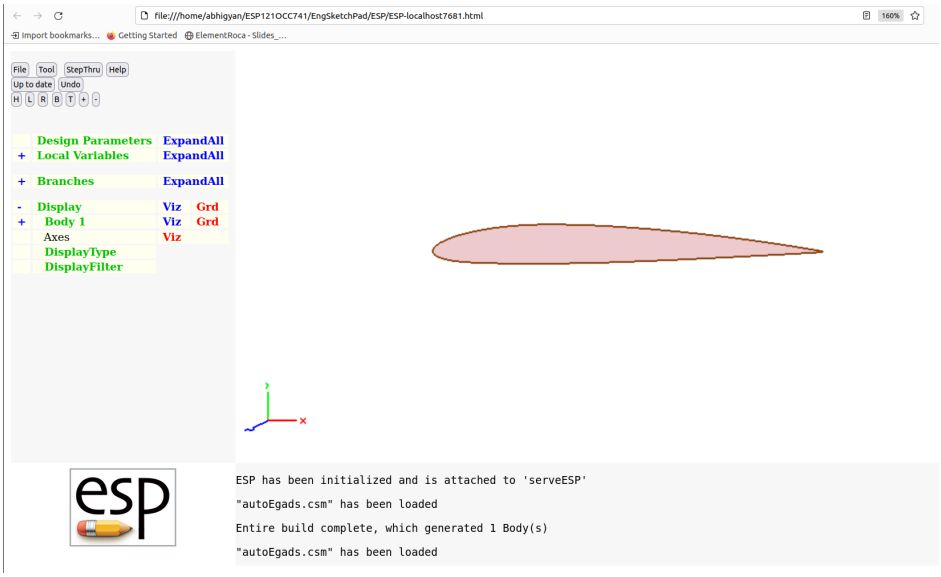

Figure 5: Conceptual Design

Next, calculations are performed by XFoil to obtain appropriate maximum camber, the thickness as well as the location of the maximum camber. This is done in 2 iterations; the first iteration is to get an idea of the range of where the values of the parameter should lie and the second to get a more accuracy in the found range. The modified geometry is shown after calculation for each parameter.

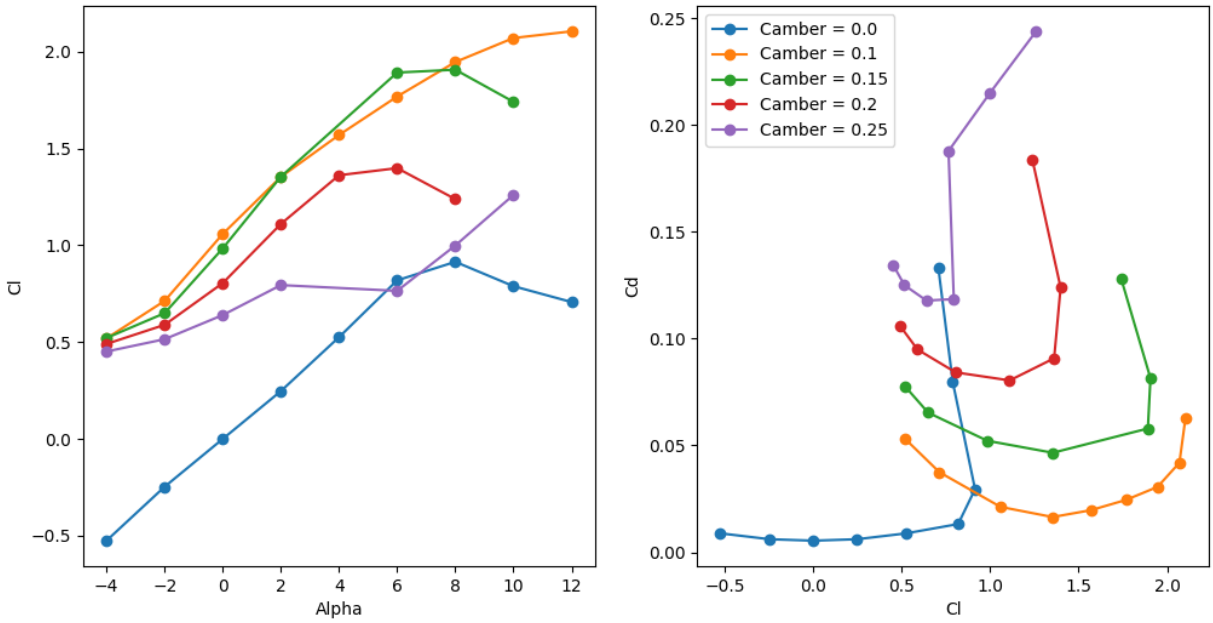

Figure 6: Maximum Camber Analysis 1

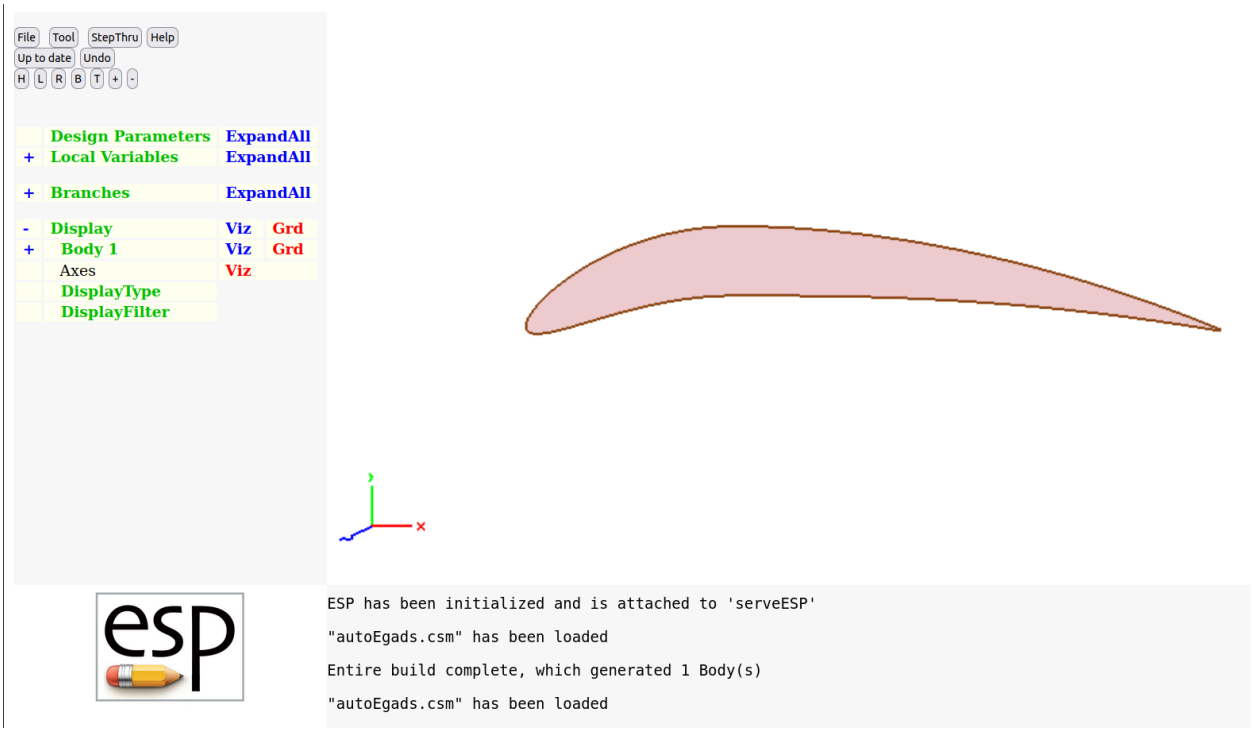

Figure 7: Geometry after Maximum Camber Analysis 1

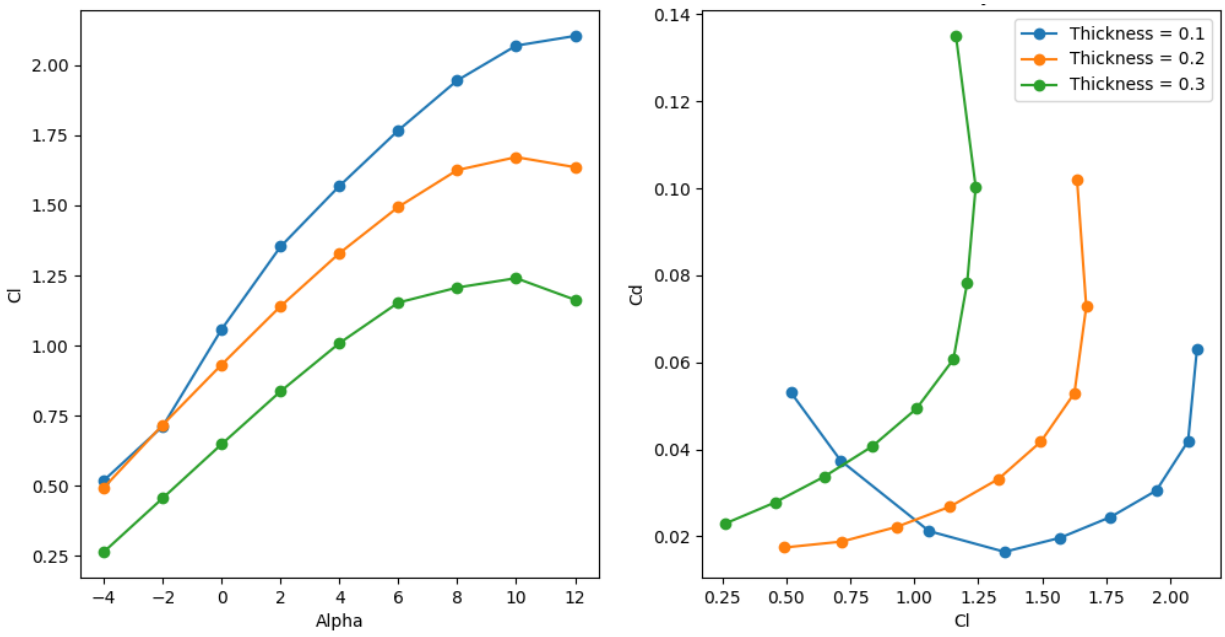

Figure 8: Maximum Thickness Analysis 1

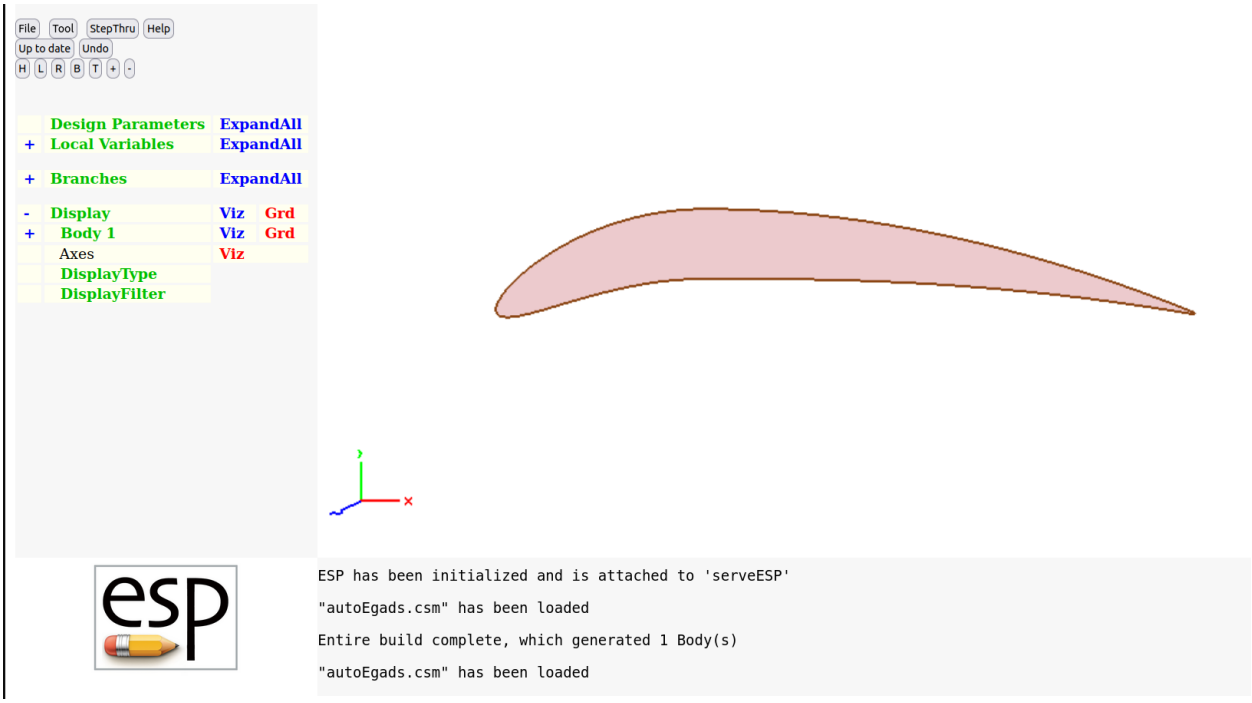

Figure 9: Geometry after Maximum Thickness Analysis 1

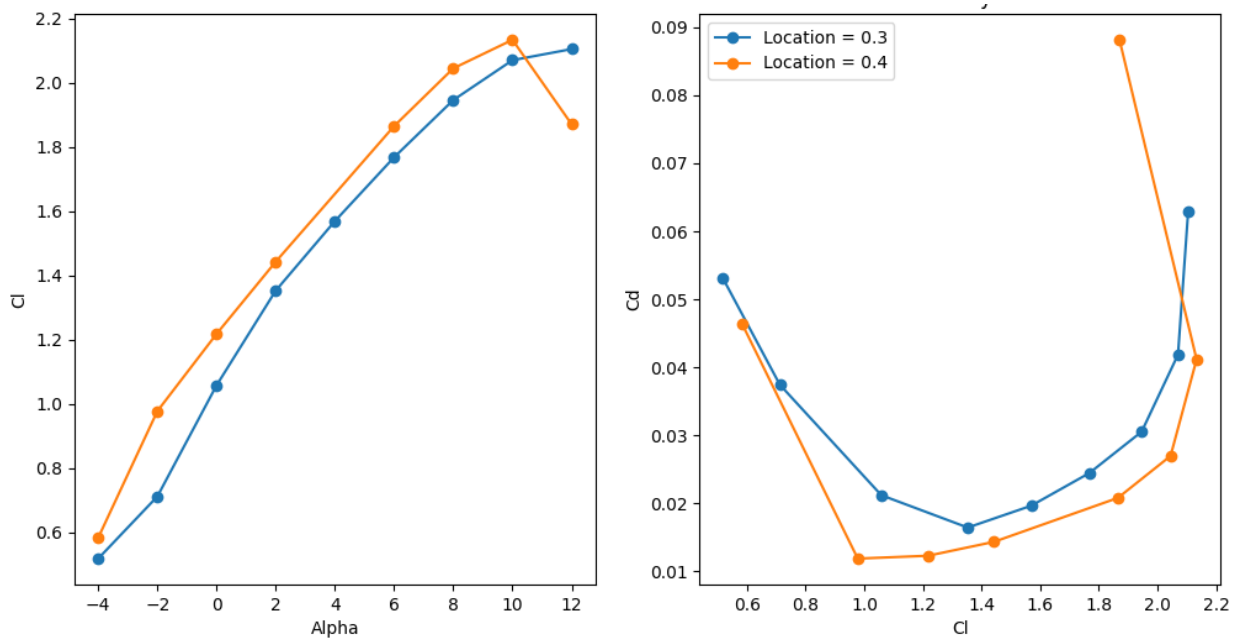

Figure 10: Location of Maximum Camber Analysis 1

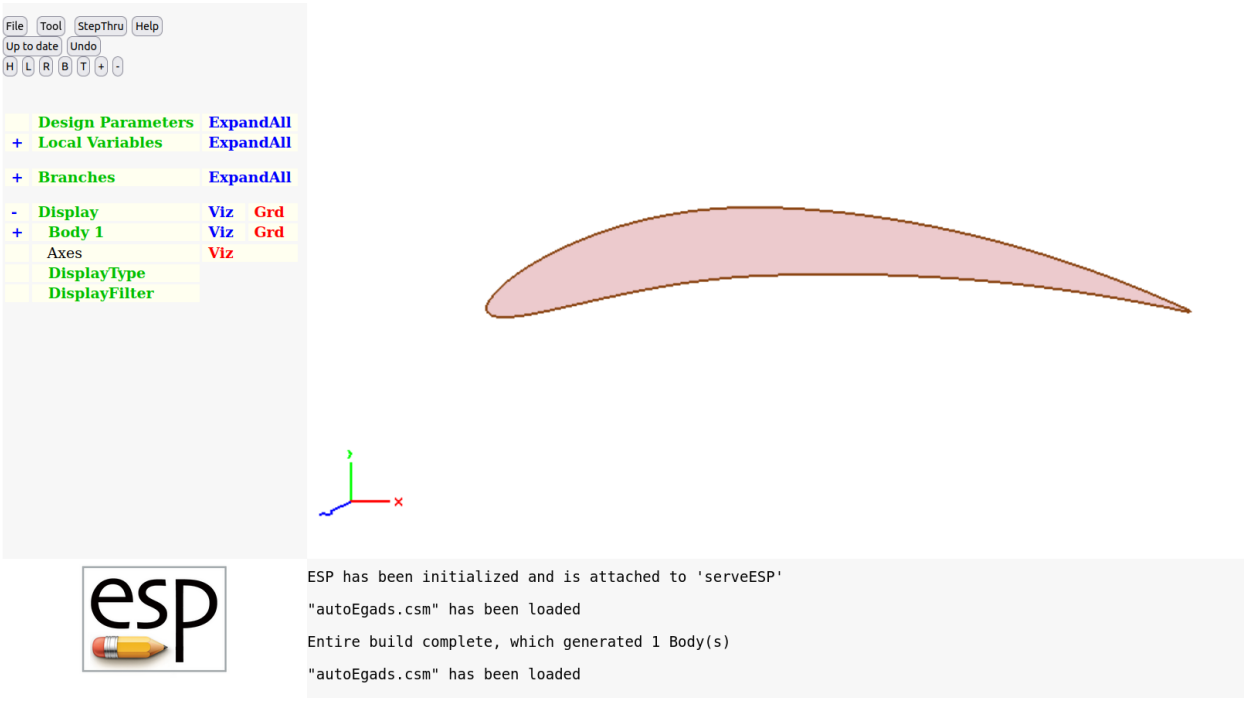

Figure 11: Geometry after Location of Maximum Camber Analysis 1

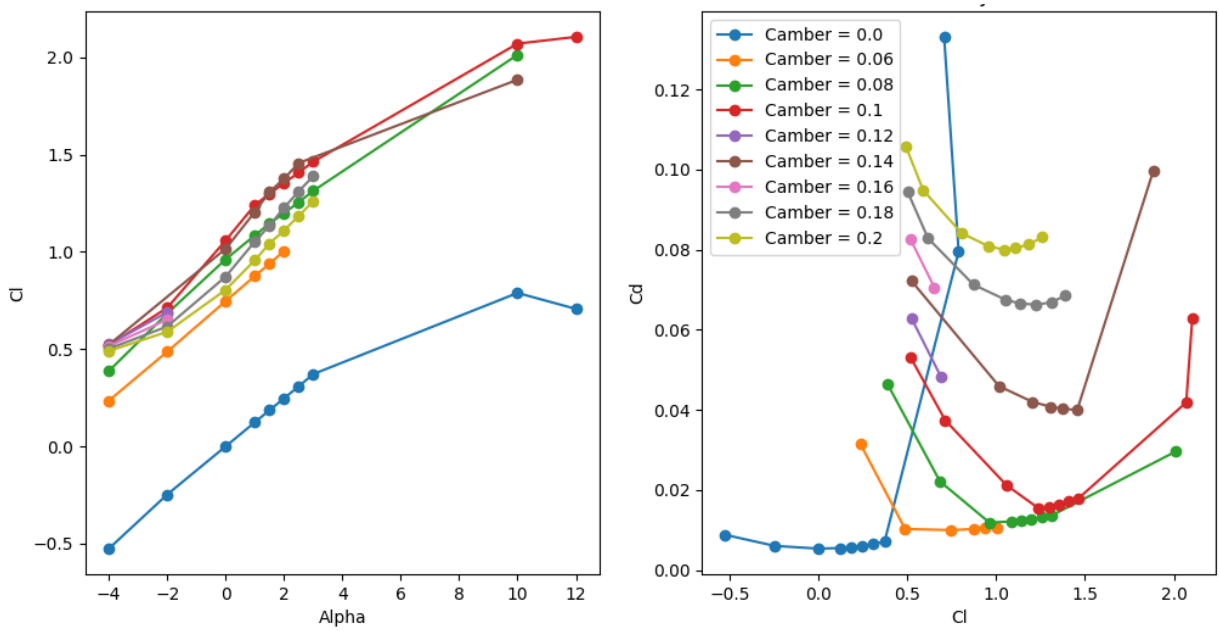

Figure 12: Maximum Camber Analysis 2

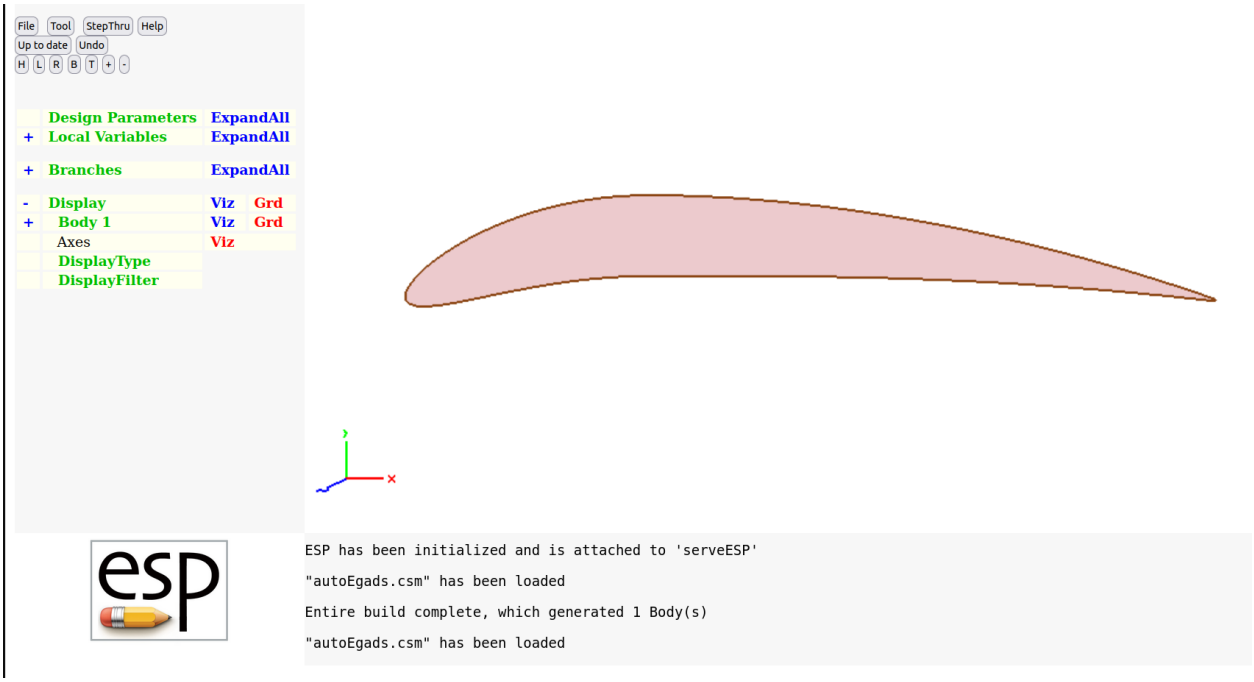

Figure 13: Geometry after Maximum Camber Analysis 2

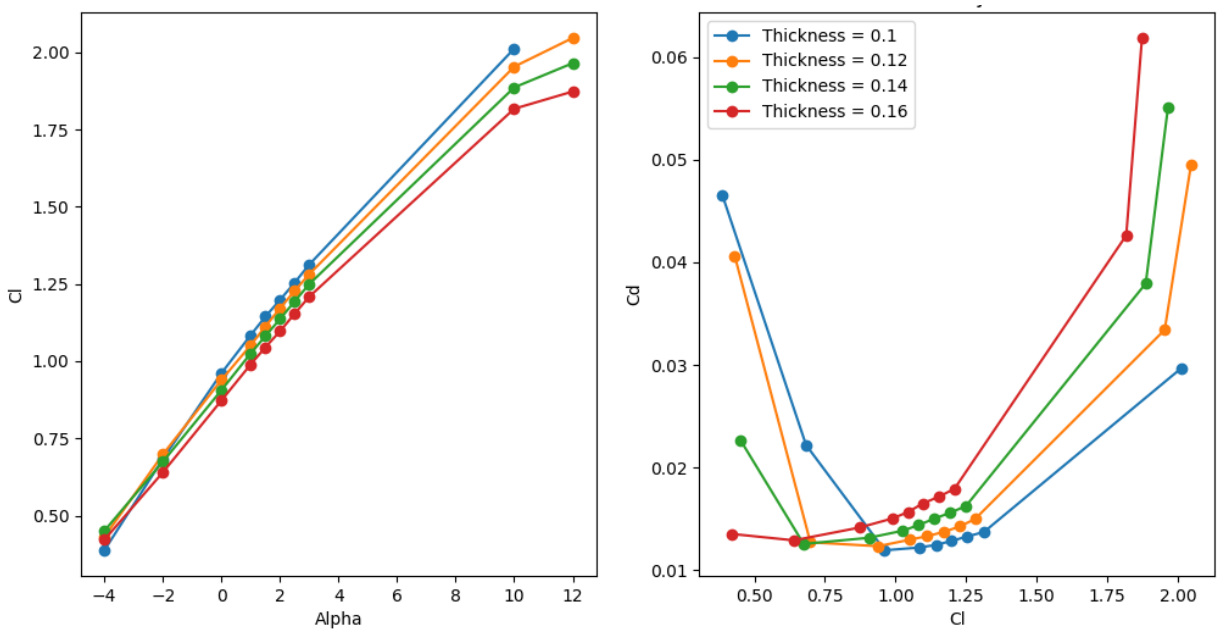

Figure 14: Maximum Thickness Analysis 2

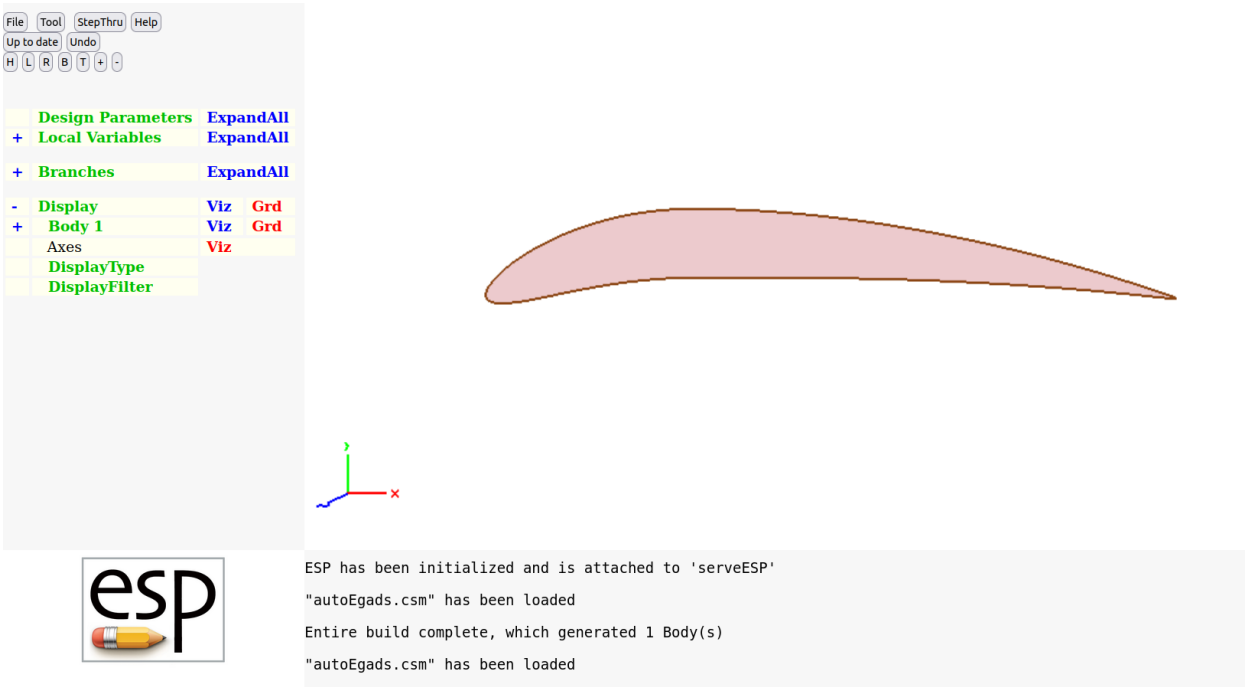

Figure 15: Geometry after Maximum Thickness Analysis 2

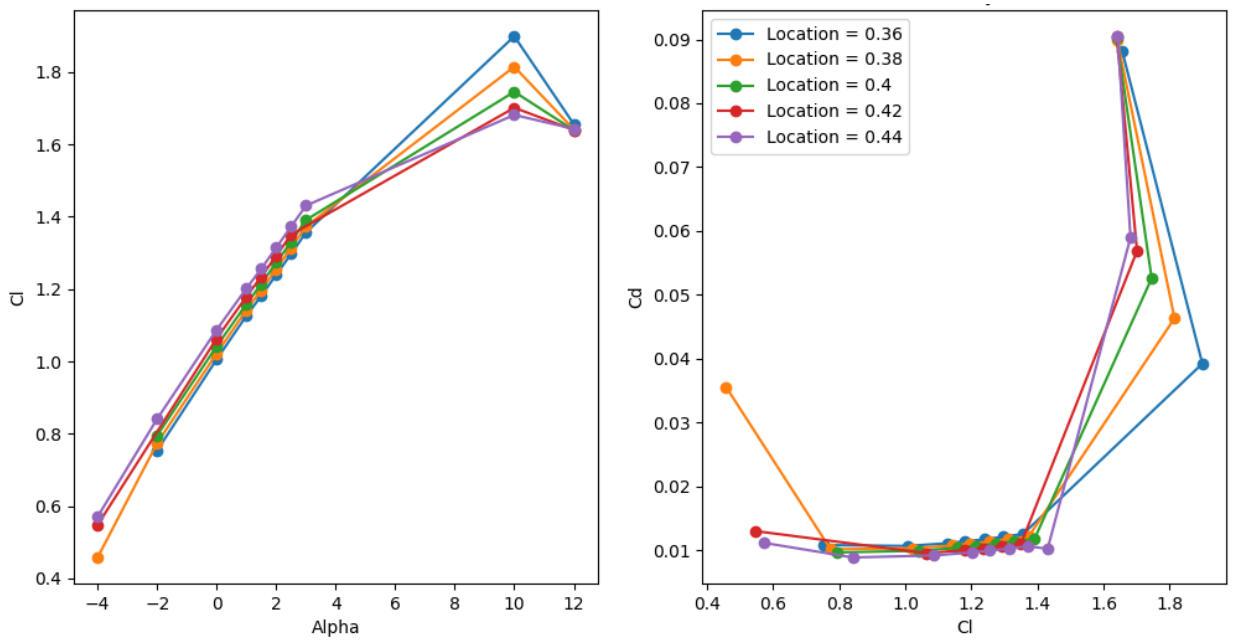

Figure 16: Location of Maximum Camber Analysis 2

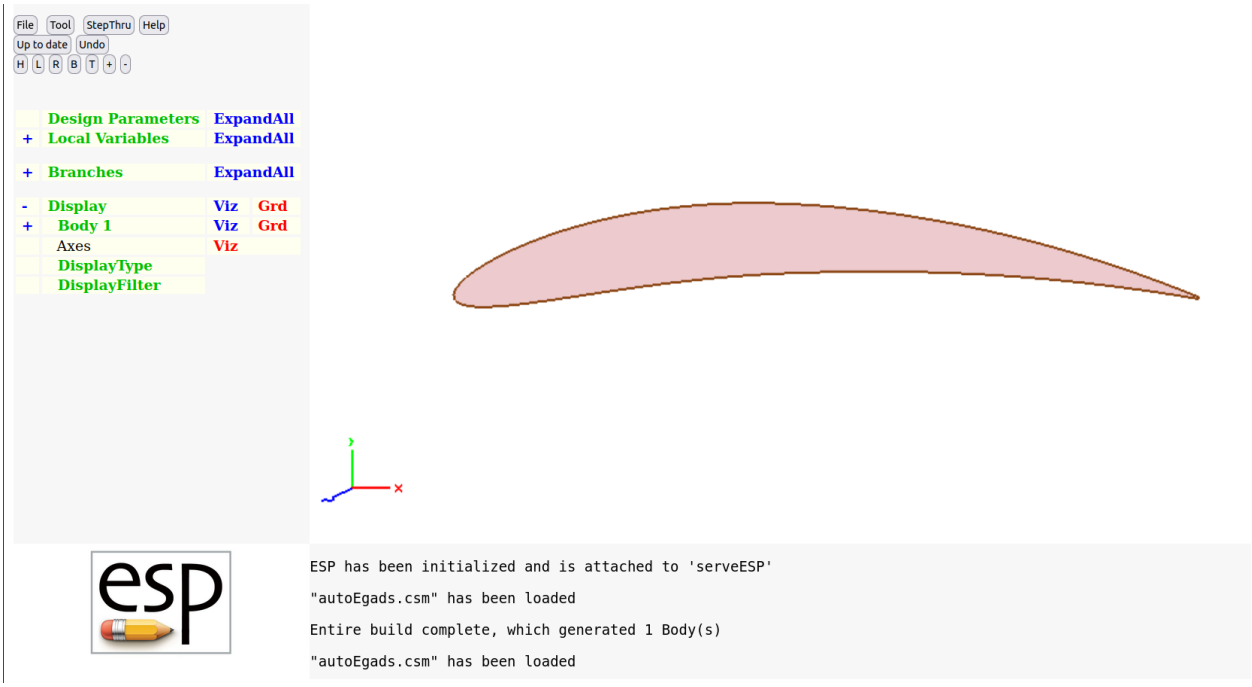

Figure 17: Geometry after Location of Maximum Camber Analysis 2

MSES

MSES performs multi-element airfoil performance analysis using integral Boundary Layer Theory and Euler theory for transonic flow, using e^N and provides analysis and geometric sensitivities. It can be used to carry out optimisation of this airfoil structure.

7.1.2 Wing Optimization using AVL

The geometric description required for optimisation by AVL requires airfoil section data specified at the appropriate locations that describe the skeleton of the aircraft. These sections when lofted as groups and finally unioned together builds the OML.

AVL is suited for aerodynamic configurations which consist mainly of thin lifting surfaces at small angles of attack and sideslip. These surfaces and their trailing wakes are represented as single-layer vortex sheets, discretized into horseshoe vortex filaments, whose trailing legs are assumed to be parallel to the x-axis. AVL provides the capability to also model slender bodies such as fuselages and nacelles via source+doublet filaments. Assumptions during calculations include quasi-steady flow and small relative flow angles, implying the roll, pitch, and yaw rates used in the computations must be slow enough. The traditional Prandtl-Glauert (PG) transformation is used to transform the PG equation into the Laplace equation, which can then be solved using the fundamental incompressible approach. With the conditions of irrotationality and linearization about the freestream, this is equal to the compressible continuity equation. The Kutta-Joukowsky relation is applied to each vortex to compute the forces, and it is still valid for compressible flow.
The aerodynamic outputs include direct forces and moments and derivatives of forces and moments w.r.t freestream, rotation and controls. It can also perform multiple trim calculations using the operating variables like alpha and control deflections along with any direct or indirect constraints if necessary.
The resulting force and moment predictions are consistent with slender-body theory, but the experience with this model is relatively limited, and hence modeling of bodies should be done with caution.

This test case using AVL is optimizing the wing structure based on the design parameter, taper ratio. First, the wing structure depicting the airfoil sections at wing root and both the wing tips are shown.

Figure 18: Initial Wing Structure

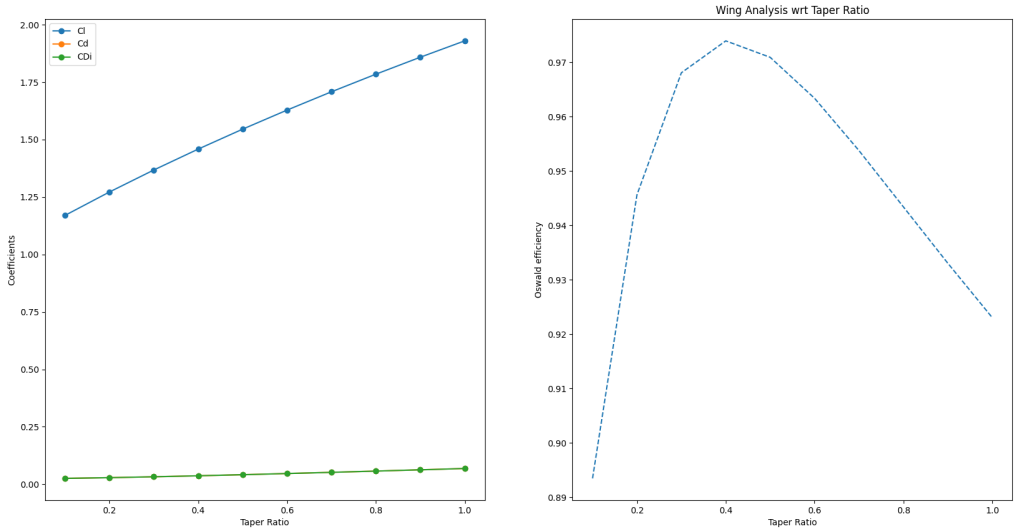

Figure 19: Analysis Values wrt Taper Ratio

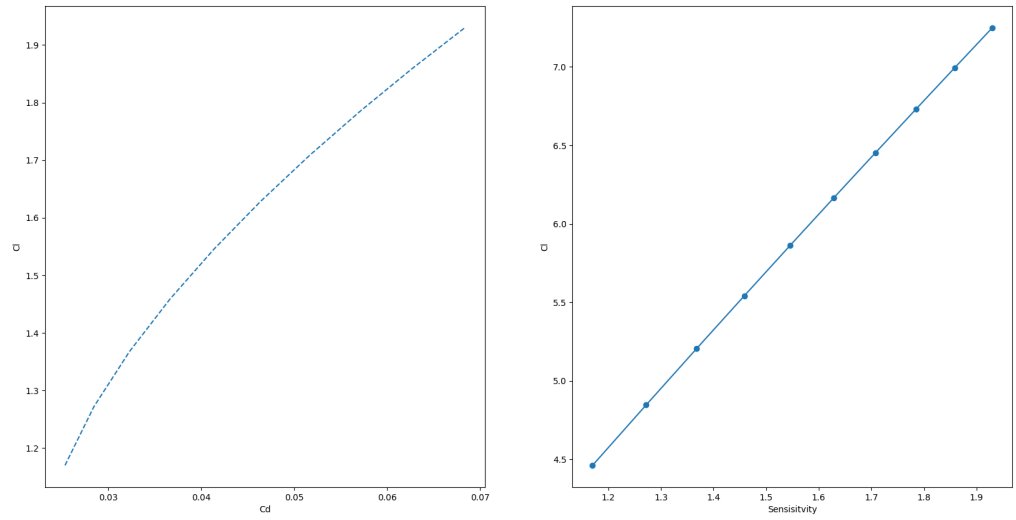

Figure 20:

As seen from the plot, the taper ratio for the best Oswald efficiency factor is chosen and the geometry is modified accordingly.
Cl and Cd data as well as sensitivity of Cl with respect to angle of attack for different taper ratios is also obtained.

Figure 21: Optimised Wing Structure

Wing CFD Analysis

The geometry created with all its attributes to define the topology well is then used to generate the appropriate meshes and the analysis is done using a flow solver. Below, a prebuilt wing with precise attributes is used to depict the usage of mesh generation suites and 3D CFD AIMs.

7.1.3 Meshing using AFLR

Advancing-Front/Local-Reconnection (AFLR) procedure is used in a suite of AFLR mesh generators. AFLR4 is fully automatic unstructured 3D surface mesh generation code that produces a 3D surface mesh on multiple surfaces, each with some sort of geometry component definition. AFLR3 is an unstructured tetrahedral element mesh generation code that produces a tetrahedral volume mesh from an existing surface triangulation.
The geometry/OML resulting from the data produced in the previous steps is inputted. All the meshing parameters are mentioned in the python script and well commented.

AFLR4 is used to generate the surface mesh.

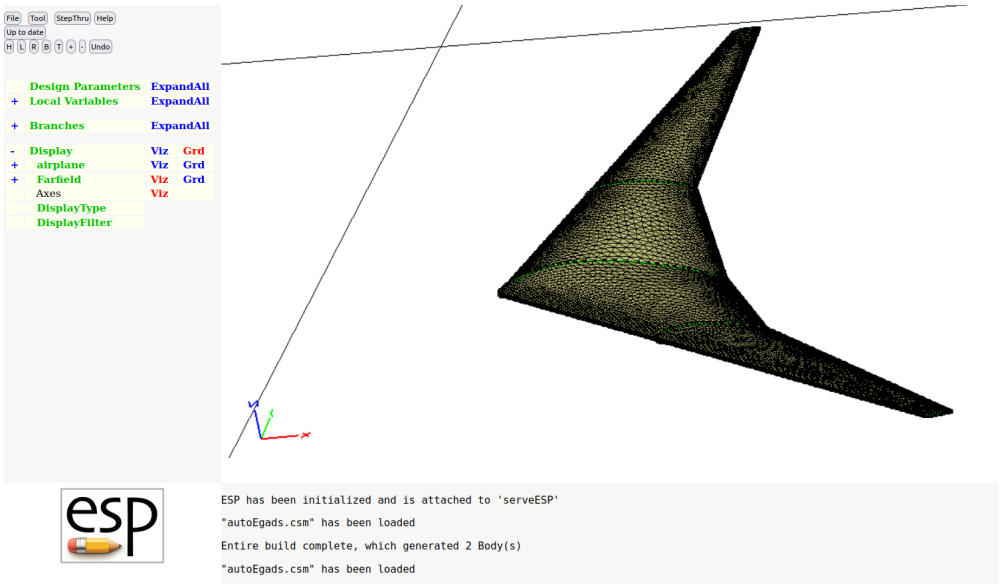

Figure 22: Surface Mesh of Geometry

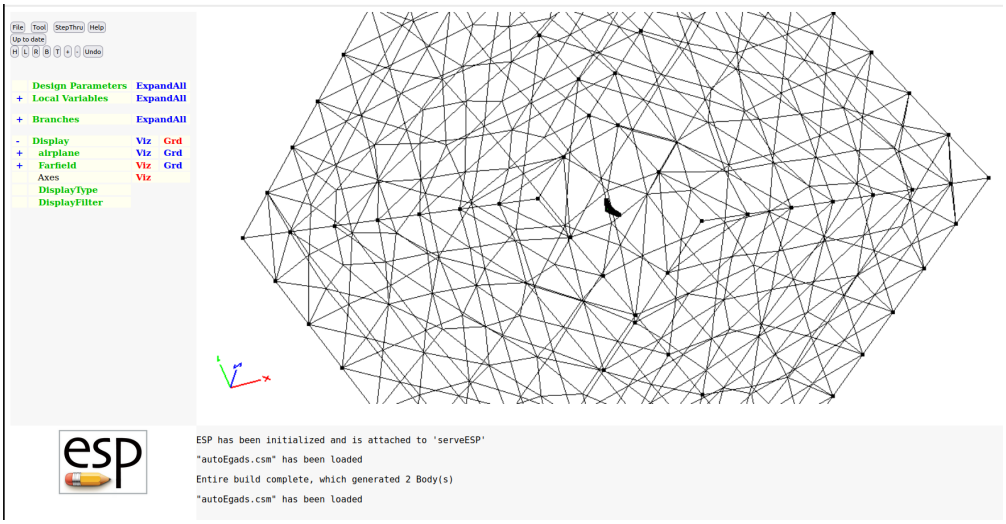

Figure 23: Surface Mesh of Farfield Boundary

AFLR3 is used to generate the volume mesh.

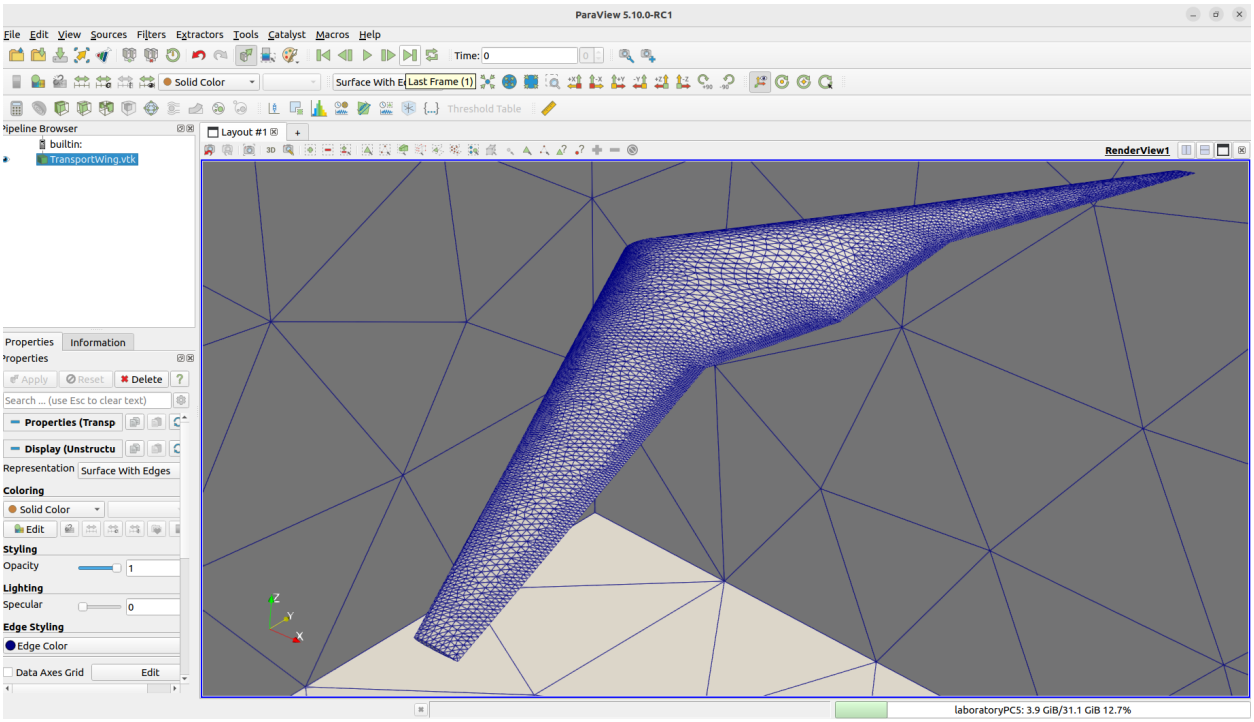

Figure 24: Volume Mesh

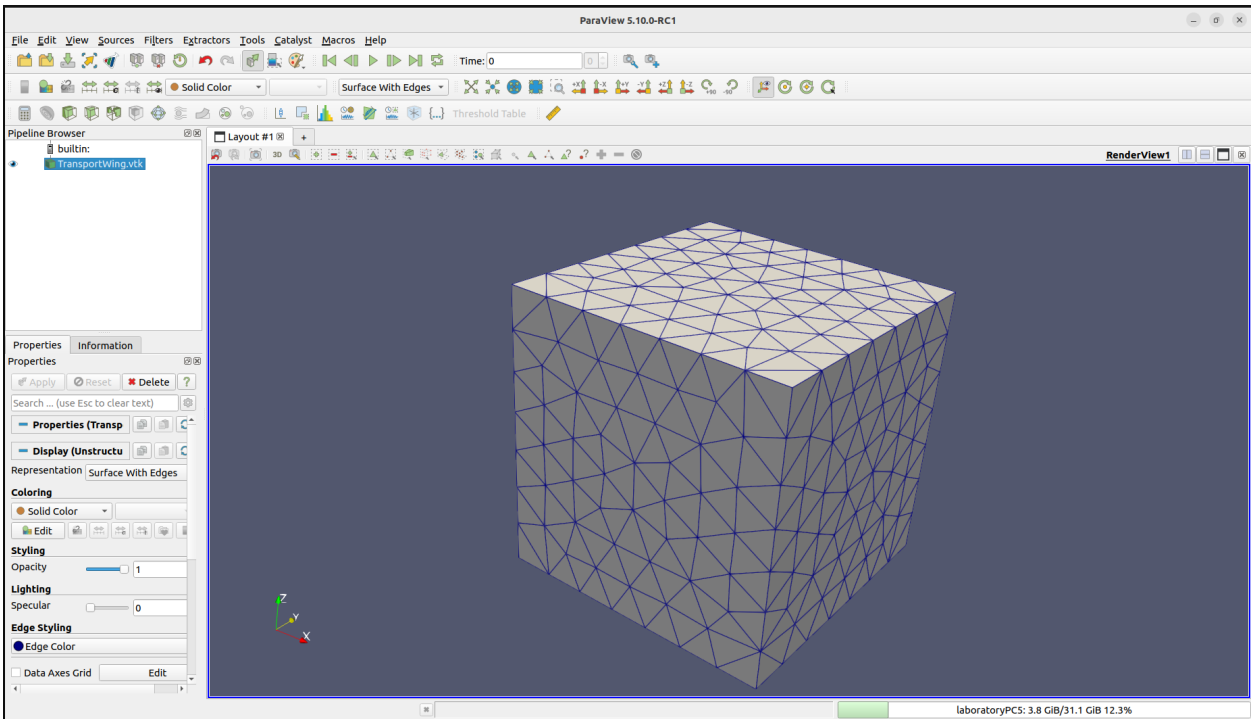

Figure 25: Farfield

7.1.4 Flow Analysis using SU2

SU2 is an open source C++ based computational analysis and design package that has been developed to solve multiphysics analysis and optimization tasks using unstructured mesh topologies. Its unique architecture is well suited for extensibility to treat partial differential equation based and PDE constrain optimisation problems not initially envisioned.
Subsonic inviscid flow analysis is performed on the geometry below using the meshes generated and the results are shown in ParaView along with Cl, Cd and other coefficient values.

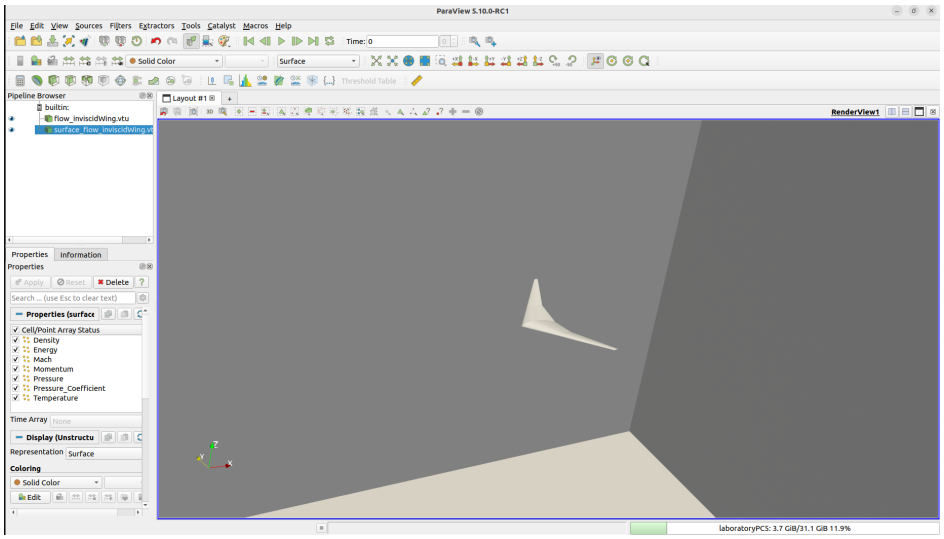

Figure 26: SU2 Geometry

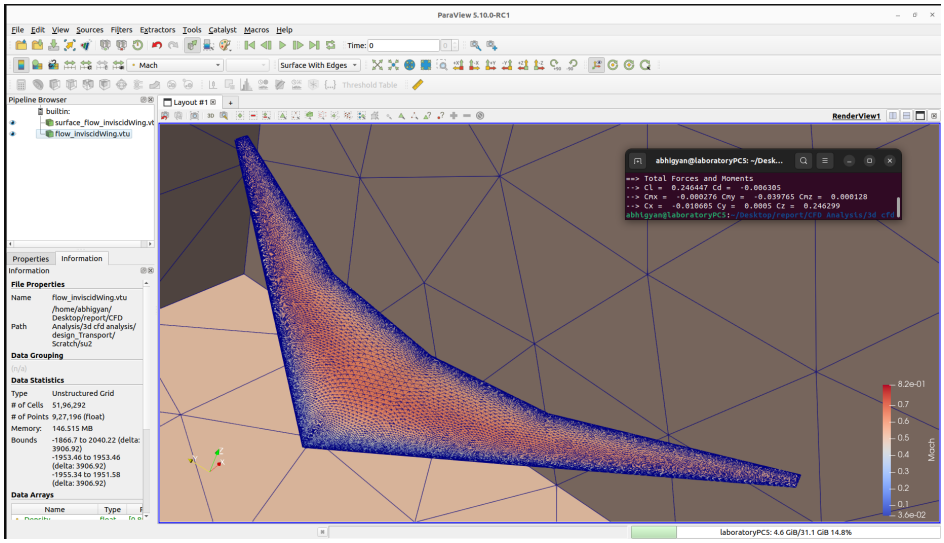

Figure 27: Flow Analysis

All the python scripts, csm scripts and other data files of this example are present https://github.com/abhx7/IISc—Summer-23-/tree/main/ESP

8 Conclusion

The Engineering Sketch Pad is an open source integrated design environment. The major benefits include watertight geometry building, persistent attribution at all levels allowing for both geometric and analytical sensitivity, accessible parameterization which allows for generation of multiple models from a single set of parameters, custom features like UDPs, collaboration as well as easy integration into the entire design and analysis ecosystem such as through import and export of cad files of different formats and direct connection to meshing tools.
The Computational Aircraft Prototype Syntheses (CAPS) environment using ESP and various Analysis Interface Modules (AIMs) seems to be a suitable option to achieve Multidisciplinary Analysis and Optimization (MDAO) goals.

9 Appendix

Some elements of the ESP environment not discussed above

9.0.1 Sensitivity

A sensitivity analysis helps determine the mesh size (e.g. how large the boundary layers are), the timestep size (although adaptive timestepping can also work), convergence criteria and other calculated variables. Adjusting the maximum value of certain criteria, such as the timestep size, and observing the results (e.g. compute time) in the Solver and observing errors based on the above parameters. A design sensitivity analysis calculates the gradients of the objective and constraint functions with respect to design variables. Similarly, by running a mesh sensitivity study, the user can have a better understanding of the spatial discretization errors and how much the mesh is affecting the solution. In an optimal scenario, as the cell size approaches zero, the spatial discretization error tends to zero as well.
In the past decade, numerous approaches for obtaining sensitivities of output functionals to CAD design parameters have been proposed. Among these methods, the Computational Aircraft Prototype Syntheses (CAPS), shipped with Engineering Sketch Pad (ESP), caught the interest of the authors due to its demonstrated capability to construct solid geometry from CAD-like parameters, calculate parametric sensitivities, and handle geometric constraints efficiently. In addition, this framework includes Application Programming Interfaces (APIs) to softwares like SU2 and meshing packages, allowing efficient field data communication among the geometry builder, meshing software, and SU2 solver to calculate parametric sensitivities. There are papers presenting workflows for interfacing SU2 with CAPS to handle geometric constraints and obtain functional sensitivities with respect to geometric parameters for MDO.

9.0.2 Deformation Method in Geometry Creation

The use of CAD for geometry definition in various industries has become prevalent because traditional deformative methods suffer from certain limitations that hinder their practical applications. This section discusses the specific issues and introduces CAD-based parameterization using constructive methods as an alternative.

One major drawback of deformative methods is their reliance on discrete surface definitions, which can be restrictive in real-world applications. The optimized geometry resulting from this approach often requires manual remodelling in CAD software, leading to prolonged design cycles and the potential introduction of manual errors. Obtaining the original sensitivity of CAD geometry design parameters from deformative methods requires additional chain-rule multiplication, which can be cumbersome and time-consuming. Deformative methods may fail to retain the realistic design intents and constraints built into the original CAD model during analysis. Although constraints on FFD control points can partially address this issue, enforcing

complex geometry relationships remains challenging.

CAD-based Parameterization Using Constructive Methods To overcome the limitations of deformative methods, this section introduces CAD-based parameterization using constructive methods as a viable solution. The proposed approach offers the following advantages: By using constructive methods, the proposed approach allows for continuous geometry definition, eliminating the constraints imposed by discrete surface representations. It simplifies sensitivity calculation by directly extracting design variable sensitivities without the need for additional chain-rule multiplication. Through CAD-based parameterization, the method ensures the preservation of design intents and constraints, maintaining the original CAD model's accuracy during analysis.

In conclusion, CAD-based parameterization using constructive methods presents a promising solution to address the limitations of deformative methods in practical industry applications. The proposed method offers a valuable alternative to traditional deformative methods enhancing geometry definition, sensitivity calculation, and constraint preservation and paves the way for further research in this field.

9.0.3 BSplines

The role of shape design in CAD applications has grown significantly, demanding more efficient and versatile tools. B-spline curves offer a promising solution due to their ability to handle complex geometries and maintain smoothness throughout the design process. B-spline curves can be used to drive analysis based shape design.
B-spline curves have a specifc mathematical representation, and properties which include control points, degree, knot vectors, and the de Boor algorithm for curve generation. It also has various advantages over other curve types, such as Bezier curves and NURBS.
Advantages of B-spline-based Shape Design

- Flexibility and Control
  B-spline curves allow designers to flexibly control the shape of curves by adjusting control points and knots, making them highly adaptable to design changes.
- Smoothness and Continuity
  The inherent smoothness and continuity properties of B-spline curves ensure aesthetically pleasing designs without sharp corners or sudden changes in curvature.
- Local Editing Capability
  B-spline curves offer local editing capabilities, enabling designers to modify specific segments of curves independently, facilitating precise adjustments.

The process of initializing B-spline curves as the foundation for shape design is explained in papers like (https://acdl.mit.edu/ESP/Publications/AIAApaper2022-1736.pdf), along with strategies for selecting the appropriate degree and number of control points. There are techniques for interpolating given data points or approximating existing curves using B-spline curves , ensuring that the designed shape conforms to specific requirements. Methods for manipulating control points to iteratively refine the shape design while maintaining smoothness are being developed. There are real-world examples of B-spline-based shape design in various industries. Examples may include automotive design, aircraft wing profiles, and consumer product aesthetics, demonstrating the versatility and effectiveness of the approach.

9.0.4 EGADSlite

EGADSlite is a lightweight geometry kernel designed to support parallel mesh generation, solver-based grid adaptation, and high(er) order spacial discretizations. Its development was motivated by concerns raised in NASA's "CFD Vision 2030 Study" regarding inadequate linkage to CAD, poor mesh generation performance

and robustness, and a lack of scalable CFD pre- and post-processing methods.
It is designed to address these concerns by providing a scalable process that views the process as an integrated whole and removes any serial portions. This approach allows for the entire process to be taken into account, not just the cost and efficiency of the solver. It is also designed to be thread-safe without any low-level common code that needs protection, which allows for appropriate scalability when threaded.
EGADSlite is useful because it provides a solution to the challenges facing CFD simulations, particularly in the context of design optimization. It allows for parallel mesh generation, solver-based grid adaptation, and high(er) order spacial discretizations, which are all critical components of CFD simulations. Additionally, it is designed to be thread-safe, which allows for appropriate scalability when threaded.
EGADSlite is needed because it addresses the concerns raised in NASA's "CFD Vision 2030 Study" regarding inadequate linkage to CAD, poor mesh generation performance and robustness, and a lack of scalable CFD pre- and post-processing methods. These concerns pose significant challenges to most current Computer-Aided Engineering (CAE) areas, including CFD, and efforts should be made to attempt to achieve a fully automated process. It provides a promising solution to these challenges and has the potential to significantly improve the efficiency and accuracy of CFD simulations.

Sidenote

I believe that adaptive mesh refinement can be integrated into the CAPS framework by building packages for mesh adaptation which could be called fitted with appropriate functionalities. It would require furthering my understanding of ESP attribution and OOP to build the interface.

References

[1] https://acdl.mit.edu/ESP/Publications/AIAApaper2013-3073.pdf

[2] https://acdl.mit.edu/ESP/Publications/

[3] https://web.mit.edu/drela/Public/web/xfoil/xfoildoc.txt

[4] https://web.mit.edu/drela/Public/web/mses/

[5] https://web.mit.edu/drela/Public/web/avl/avldoc.txt

[6] https://www.simcenter.msstate.edu/software/documentation/system/index.html

[7] https://arc.aiaa.org/doi/full/10.2514/1.J053813

[8] Bspline paper

[9] https://acdl.mit.edu/ESP/Publications/AIAApaper2018-1401.pdf

https://acdl.mit.edu/ESP/ has all available resources for ESP, including a training program for ESP and CAPS.