Linköping Studies in Science and Technology

Parameterized Automated Generic Model for Aircraft Wing Structural Design and Mesh Generation for Finite Element Analysis

Muhammad Sohaib

LINKÖPINGS UNIVERSITET

Institute of Technology
Department of Management and Engineering (IEI)
SE-581 83 Linköping, Sweden

Linköping 2011

Examiner: Dr. Christopher Jouannet, LiTH
Supervisor: Mr. Kristian Amadori, LiTH

Master Thesis

Muhammad Sohaib
ISRN: LIU-IEI-TEK-A--11/01202--SE
Master's Program in Mechanical Engineering
Linköping 2011

Abstract

This master thesis work presents the development of a parameterized automated generic model for the structural design of an aircraft wing. Furthermore, in order to perform finite element analysis on the aircraft wing geometry, the process of finite element mesh generation is automated.

Aircraft conceptual design is inherently a multi-disciplinary design process which involves a large number of disciplines and expertise. In this thesis work, it is investigated how high-end CAD software's can be used in the early stages of an aircraft design process, especially for the design of an aircraft wing and its structural entities wing spars and wing ribs.

The generic model that is developed in this regard is able to automate the process of creation and modification of the aircraft wing geometry based on a series of parameters which define the geometrical characteristics of wing panels, wing spars and wing ribs. Two different approaches are used for the creation of the generic model of an aircraft wing which are "Knowledge Pattern" and "PowerCopy with Visual Basic Scripting" using the CATIA V5 software. A performance comparison of the generic wing model based on these two approaches is also performed.

In the early stages of the aircraft design process, an estimate of the structural characteristic of the aircraft wing is desirable for which a surface structural analysis (using 2D mesh elements) is more suitable. In this regard, the process of finite element mesh generation for the generic wing model is automated. The finite element mesh is generated for the wing panels, wing spars and wing ribs. Furthermore, the finite element mesh is updated based on any changes in geometry and the shape of the wing panels, wing spars or wing ribs, and ensure that all the mesh elements are always properly connected at the nodes. The automated FE mesh generated can be used for performing the structural analysis on an aircraft wing.

Acknowledgements

The master thesis work presented here is carried out at the Institute of Technology (LiTH), Department of Management and Engineering (IEI), Linköping University in Sweden. The author would like to express his sincere gratitude to the wonderful help, support and guidance given by Dr. Christopher Jouannet and Mr. Kristian Amadori who have made this master thesis work possible. Valuable discussions with my supervisor Mr. Kristian Amadori have enabled a deeper understanding of the problem and helped greatly in completion of this thesis work. Furthermore, the author would like to thank Mr. Guillaume Martinat and Mr. Raghu Chaitanya for helpful comments and views during the thesis work. In addition, I would also like to thank other supporting staff, colleagues and members of the Department of Management and Engineering (IEI) in regards to this thesis work.

Muhammad Sohaib

Linköping, September 2011

Dedicated to my sweet and loving Ammi, Abbu,
Tahir, Athar bhai and Shahida

Nomenclature

Notations

Symbol	*Description*
α	Angle of Attack
b	Semi-wing span
C_R	Root Chord Length
C_T	Tip Chord Length
Λ_{LE}	Wing Panel Leading Edge Sweep Angle
γ	Taper Ratio
L/D	Lift to Drag Ratio

Abbreviations

Abbreviation	*Description*
CAD	Computer Aided Design
CFD	Computational Fluid Dynamics
FEA	Finite Element Analysis
FEM	Finite Element Method
FE	Finite Elements
VB	Visual Basic
VBA	Visual Basic for Applications
KP	Knowledge Pattern
PC	PowerCopy
MAC	Mean Aerodynamic Chord
AR	Aspect Ratio
EKL	Engineering Knowledge Language
IDE	Integrated Development Environment
CATIA	Computer Aided Three-dimensional Interactive Application
UDF	User defined Feature

Table of Contents

List of Figures

List of Tables

1 Introduction

Aircraft design is a complex and multi-disciplinary process that involves a large number of disciplines and expertise in aerodynamics, structures, propulsion, flight controls and systems amongst others. During the initial conceptual phase of an aircraft design process, a large number of alternative aircraft configurations are studied and analyzed. Feasibility studies for different concepts and designs are carried out and the goal is to come up with a design concept that is able to best achieve the design objectives. One of the crucial studies in any aircraft design process is the conceptual design study of an aircraft wing. The aircraft wing is one of the most critical components of an aircraft not only from an aerodynamics point of view but also from a structural point of view. The aircraft wing is designed in such a way that it is able to provide the requisite lift while minimizing the drag. Drag is critical from the aerodynamics point of view because it directly affects the performance of the aircraft like fuel efficiency and range. Not only does the wing provide the necessary lift during flight, the aircraft wing is designed structurally to carry the entire weight of the aircraft. Also, in modern commercial aircrafts and fighter airplanes, the aircraft wing has more than one role. It not only carries the fuel required for the flight but is also used to provide storage bays where, the aircraft landing gears can be mounted and stowed during takeoff (which are normally placed inside the wing root of an aircraft). Furthermore, modern commercial airplanes have podded engines which are placed below the wing. This means that the aircraft wing has to be sufficiently strong from the structural perspective to carry the weight of these engines, fuel inside the wing box and internal components. A variety of components are also placed inside the aircraft wing which includes electro-mechanical actuators, fuel lines, and hydraulic, pneumatic and electrical systems amongst others. All of these components are to be compactly placed inside the wing, thus, the aircraft wing has to perform structurally and aerodynamically well to deliver the desired performance. Weight is one of the fundamental critical factors in any aircraft design process and aircraft designers are always on the lookout for ways to minimize the weight of the aircraft. This means that a light weight aircraft should have a light weight wing. A light weight aircraft is thus beneficial for increasing the design performance.

In the conceptual phase of an aircraft design process, different design studies are carried out for different components of the aircraft. One of the major portions of these studies is dedicated towards the design of the aircraft wing both from a structural and aerodynamics point of view. However, in this stage High-end CAD software's are not employed as they are thought to be too complex or demanding to be used during this stage. Therefore, the promising design configurations have to be remodeled again later in the detail design process which increases cost and the time to production. It can be very beneficial from a design perspective, if these CAD software's are employed from the start of the aircraft design process. This would enable less remodeling of the design in the detail design process and would also enable increased capability to do modeling and simulation during the conceptual phase. A generic model is thus required in this regard that would speed up the design process of analyzing different aircraft wing configurations especially from a structural design perspective. An aircraft designer would thus be able to focus on different design criteria's and

configurations rather than worrying about CAD geometries. The CAD based generic model can eventually also be used for performing aerodynamic and CFD studies on the aircraft wing as well. Furthermore, this model can also help other engineers from other departments like systems and flight controls to check the feasibility of placing internal components at different positions inside the aircraft wing. However, it is required that the CAD model should be sufficiently robust and reliable to be able to cater for different design configurations and geometries. From the structural perspective, aircraft wing should be designed to provide the necessary strength and stiffness at minimum weight. The generic model should enable different structural configurations to be sized so as to provide the required strength and stiffness. In order to do this, the generic model should enable easy to do finite-element-analysis of the entire aircraft wing configuration. This would enable visualization and calculation of the stiffness and strength of the structure but also help in minimizing weight and cost. The finite element mesh should automatically be generated and a finite element analysis (FEA) could easily be performed. In the conceptual design phase, an estimate for the structural characteristics of different aircraft wing configurations is desirable which can be performed swiftly by using a surface structural analysis (2D mesh elements) as compared to a solid structural analysis (3D mesh elements) which would have an increase penalty in the amount of time, resources and cost to perform structural simulation when compared with a 2D analysis.

The generic model presented in this master thesis work can be used for the aircraft wing design. The generic aircraft wing model that is developed includes the aircraft wing panels and the structural components of the wing including both wing spars and wing ribs. The generic model enables automated mesh generation of the aircraft wing geometry along with its structural components. Furthermore, both the solid and the surface model of the aircraft wing are integrated together in the generic model. The surface model is used to perform the finite element analysis. The aircraft wing model is parameterized and these parameters are used to change the size, shape and geometry of the aircraft wing and its structural components including spars and ribs. Two alternative approaches are used for designing this generic model. These approaches are “Knowledge Pattern” and “Power Copy with Visual Basic Scripting (VB)”. A comparison between the two approaches is presented. The generic wing model that is developed during this thesis work can not only be used for designing the aircraft wing but also can be used for designing and modeling propellers, turbine blades, ship propellers and rudders amongst others.

1.1 Aim

The main aim of this thesis work is the development of a generic parameterized aircraft wing model that can employed from the early stages of an aircraft design process. The main tasks of this thesis work includes the following,

1. Develop a Parameterized Automated Generic Model for an Aircraft Wing
2. Implementation using PowerCopy with Visual Basic Scripting Approach
3. Implementation using Knowledge Pattern Approach
4. Comparison between PowerCopy with VB Approach and Knowledge Pattern Approach
5. Structural Mesh Generation of Complete Aircraft Wing Model

This thesis work presents the development of the generic parameterized aircraft wing model by using CATIA V5 CAD software which provides tools and features for automated geometry generation and modification. In using this CAD software, two different approaches are used for the implementation of the generic model which are knowledge pattern and PowerCopy with VB scripting. A comparison between both of these approaches is performed. A structural mesh generation of the generic aircraft wing model is also created. It is ensured that the mesh elements are properly connected at the nodes and the mesh elements are of good quality.

1.2 Motivation

The generic aircraft wing model offers a series of advantages and thus provides a motivation for its development. Some of the advantages provided by a generic aircraft wing model are listed below,

1. Single model is able to represent different aircraft wing planform and configurations
2. Automated CAD geometry generation
3. Automated finite element mesh generation
4. Less file management
5. Faster start-up time for modelling & analysis
6. Lower costs

Firstly, by using a generic model structure, a single model of an aircraft wing is able to cover the different aircraft wing planform and configurations. So, for example, the generic model can be used for designing both straight wing and swept wing etc. Second advantage of using a parametric automated generic model is the automatic generation of the CAD geometry. This dramatically speeds up the process of geometry generation of different wing planform or configurations so that the designer focuses more on the design of the aircraft wing rather than worrying about the creation of the geometry. In order to speed up the process of analysis and simulation, the generic wing model offers the advantage of automated mesh generation. This means that a finite element mesh can be made automatically for different aircraft wing planform and shapes, thus enabling the designer to simulate and analyze many different wing planform from the structural point of view. By using a generic model structure, there is a small number of files to manage and provides a big advantage over traditional single models, where, each model has its own files associated with it. By enabling automated geometry and

mesh generation, the generic model structure provides faster start-up time for modelling and simulation. This lowers the time for designing and analyzing aircraft wings and thus lowers the cost. The generic wing model can help in lowering costs associated with design process. The generic wing model offers cost reduction in that specific CAD designers are not needed for building the geometry, instead, the wing model CAD geometry is automatically created and thus the model can directly be used by the engineer and designers building the wing. Secondly, less man-hours on development means more resource savings and profits for the company. Since, the generic model structure offers automated geometry and mesh generation, and enables faster startup time for modelling and simulation, it can help in lowering man-hours and costs associated with development.

1.3 Applications

The generic wing model that is developed in this thesis work can not only be used for designing of the aircraft wing which is its primary application, but, also can be used for designing other types of wing shapes used in other applications. The structure of the model is made as general and generic as possible for enabling its use in different applications. For example, the generic wing model can be used for designing aircraft propeller blades, gas turbine blades, wind turbine blades, car spoilers and ship propeller blade etc.

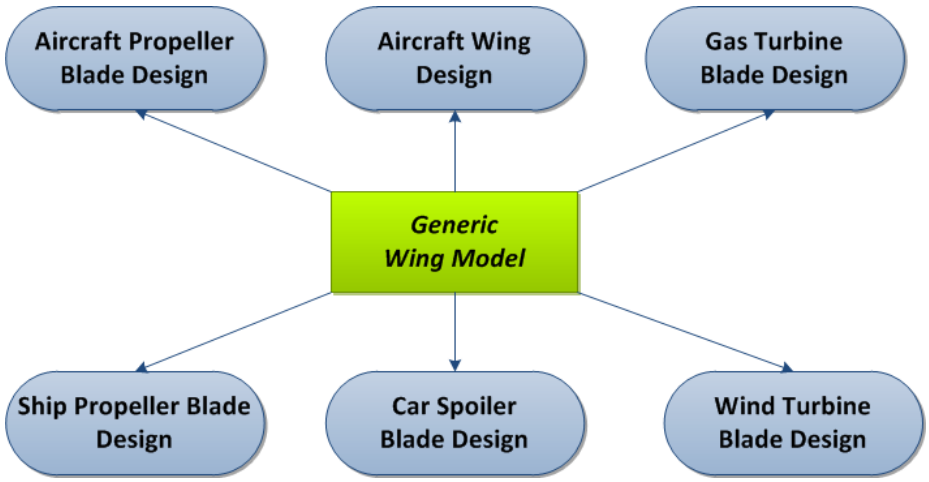


Figure 1: Applications of generic wing model

1.4 Methodology

The methodology adopted in this master thesis work is shown in the figure below,

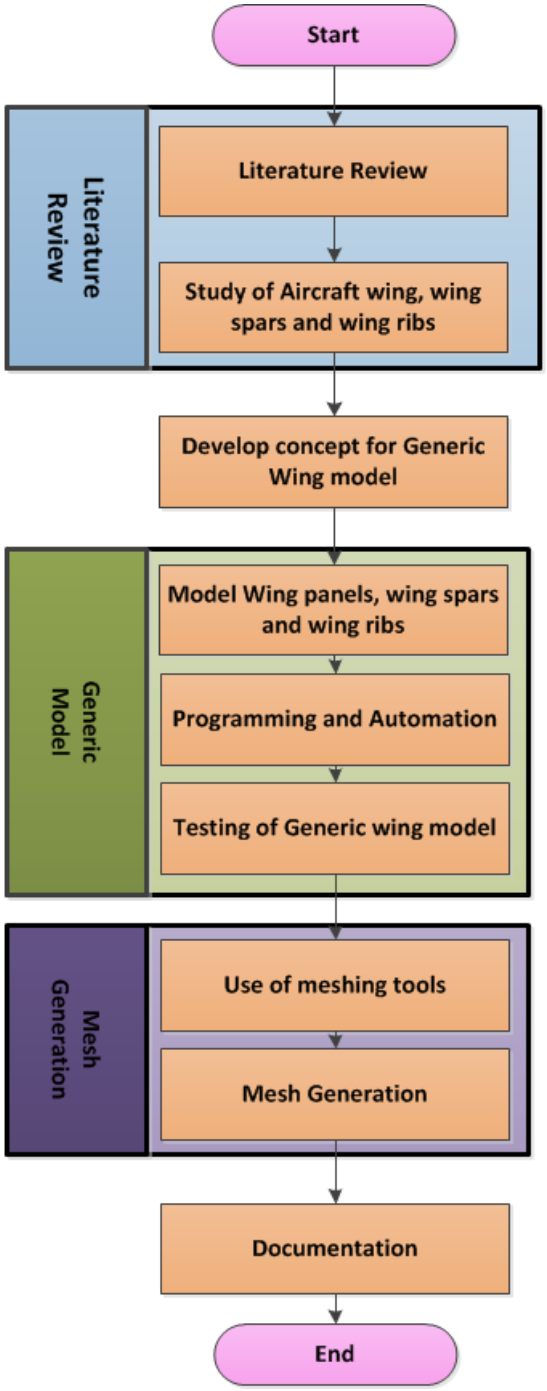


Figure 2 Methodology (Breakdown of master thesis work)

The breakdown of the master thesis work is divided into,

- Literature review
- Generic wing model development
- Automated mesh generation
- Documentation

The first part of the thesis work started with a comprehensive literature review on the aircraft wing including study of wing spars and wing ribs. Furthermore, it also includes study of

research papers on this subject as well as development methods for generic wing model. After the literature review, a concept for the generic model structure was developed. The concept was then used to model the generic wing model in CATIA. The generic model development including programming and writing automation codes that controls all aspects of the generic model. Then, the testing of the generic model was done to insure that the generic aircraft wing model is able to cater for different types of aircraft wing configurations and planform shapes. The next step included the use of meshing tools for automating the structural mesh generation of the generic aircraft wing model. Documentation of the thesis work was written continuously with the progress in the thesis work.

2 Theory: Aircraft Wing

The aircraft wings are the primary lift producing device for an aircraft. The aircraft wings are designed aerodynamically to generate lift force which is required in order for an aircraft to fly. Besides generating the necessary lift force, the aircraft wings are used to carry the fuel required for the mission by the aircraft, can have mounted engines or can carry extra fuel tanks or other armaments.

The basic goal of the wing is to generate lift and minimize drag as far as possible. When the airflow passes the wing at any suitable angle of attack, a pressure differential is created. A region of lower pressure is created over the top surface of the wing while, a region of higher pressure is created below the surface of the wing. This difference in pressure creates a differential force which acts upward which is called lift. For most aircrafts, where, the wings are the primary structures to generate lift, the aircrafts wings must generate sufficient lift to carry the entire weight of an aircraft.

In modern commercial, fighter and jet aircrafts, the aircraft wings are not only designed to provide the necessary lift during the different phases of flight, but also have a variety of other roles and functions. In commercial jet aircrafts, the aircrafts wings are used as the primary storage system for the jet fuel required for the flight. The jet fuel is normally carried in a structure placed inside the outer surface of the wing called a wing box. The fuel carried inside the wing box directly delivers fuel to the jet engines. Modern commercial airplanes like the Boeing 747 and the Airbus A380 amongst many other aircrafts also have podded engines which are placed on the wing. The fuel inside the wing box feeds these jet engines. The mounting of these engines on the wing produces structural loads as well. In fighter aircrafts, weapon systems, missiles and extra fuel tanks or other armament is normally mounted below the wing surface using weapon-pods. These pods are normally attached to the wing spars running through the wing span. During the flight, the aircraft wing has to deal with aerodynamic, gust, wind and turbulence loads. Also, the aircraft wings have to deal with aero-elastic and structural loads as well. Therefore, the aircraft wings must be designed structurally and aerodynamically well for providing good overall performance in all phases of flight.

2.1 Aircraft Wing Configuration

Based on the type of mission requirements and the different flight regimes the aircraft will encounter be it subsonic, transonic, supersonic or even hypersonic, there are different wing configurations [4] and planform shapes that are available to the aircraft designer for the wings. Some of them are shown in the figure below.

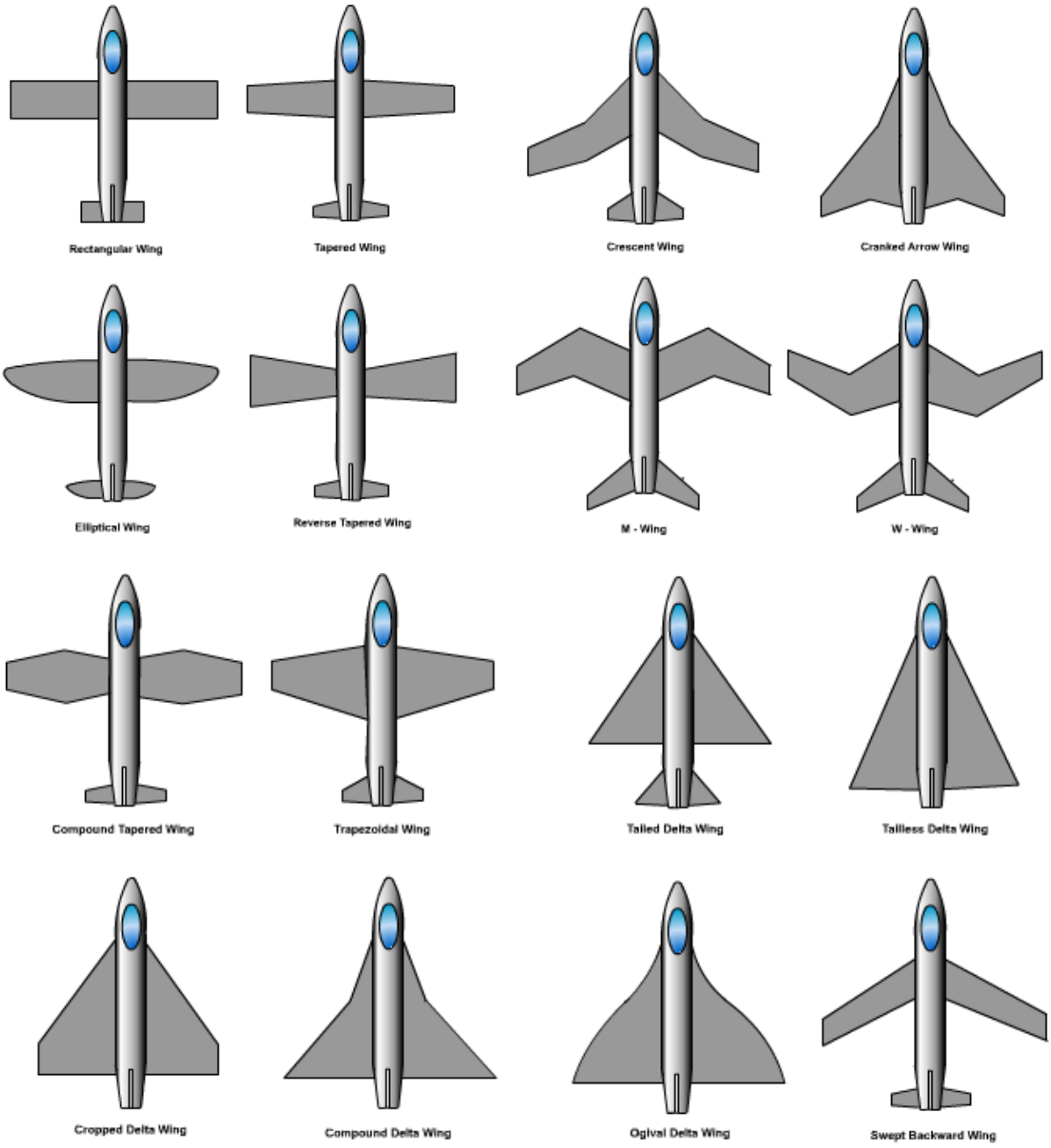


Figure 3: Aircraft wing planform and configurations

A detail description of different types of aircraft wing configurations is presented in Appendix A.

2.1.1 Positioning and Shape of Wing

When the positioning of the wing on the fuselage and the shape of the wing is changed, different types of wing configurations can be achieved, some of which are shown in the figure below,

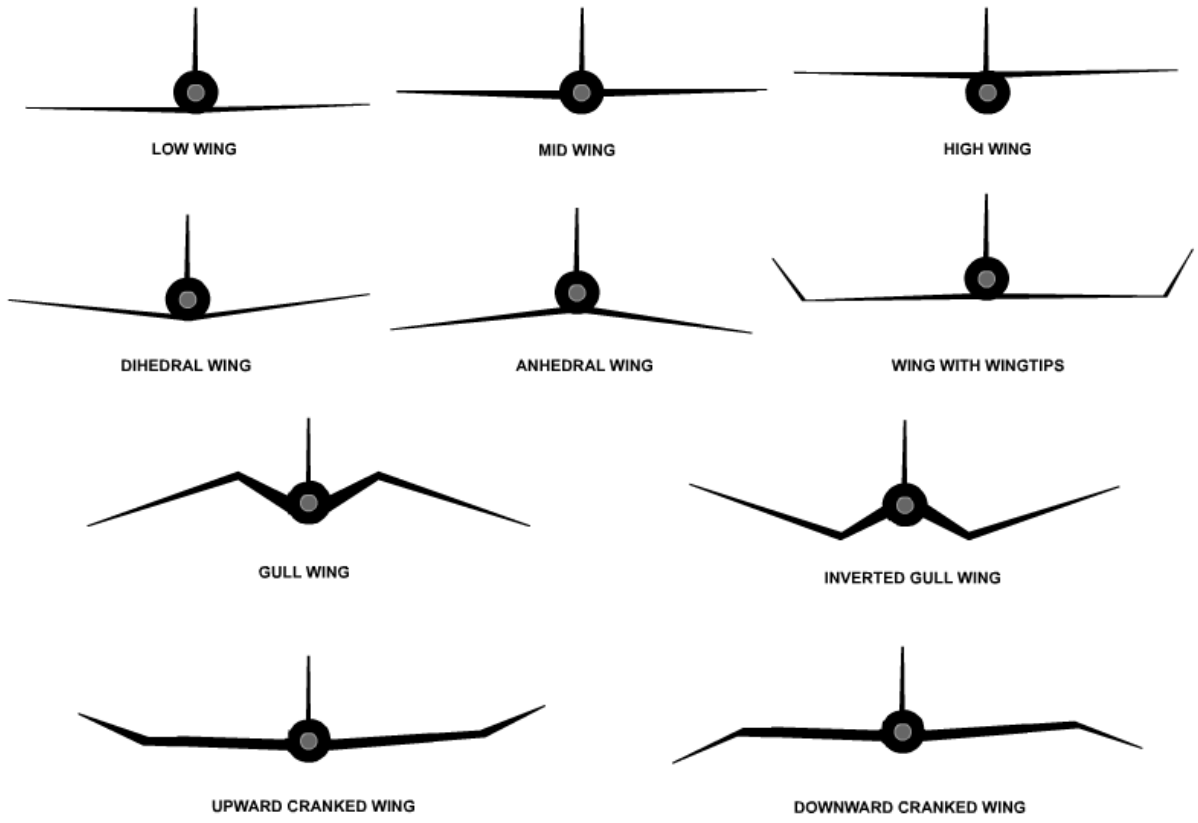


Figure 4: Positioning and Shape of the Wing

A detail description of different types of positioning and shapes of the aircraft wing as well as different factors that affect the aircraft wing design are presented in Appendix A.

2.2 Wing Spars

The wing spars are the main load carrying structural member of the aircraft wing. The wing spars are used to carry the loads that occur during the flight (flight loads) as well as carry the weight of the aircraft wing while on the ground (ground loads). The wing spars run throughout the span of the wing from the root to the tip and can be placed perpendicularly or at an angle. Commercial aircrafts sometimes have less number of wing spars than fighter aircrafts, this is due to the fact that, the fighter aircrafts have to deal with much higher flight loads. The structural and forming members of the aircraft wing known as "wing ribs" are also attached to the wing spars. The wing ribs are aerodynamically shaped and thus provide the aircraft wing with a characteristic airfoil shape. The number of wing spars in a wing varies with values between one and more. Other load carrying structural members like the stressed skin construction also helps in carrying the flight loads. When the aircraft is on the ground, the weight of the gravity pulls the wings downward. This gravitational load is also carried by the wing spars running through the wing span. If the majority of the load and forces is carried by a single spar in the aircraft wing, it is called as the 'main spar'. Main spars are common in smaller lightweight aircrafts, where, the wing spar runs from the wing root to the wing tip.

A single aircraft wing (or a monoplane wing) basically acts like a cantilever beam. The wing spars are then used to carry the loads and forces acting on the monoplane wing structure. Wing box which is another important structural member that is placed inside the aircraft wing

is attached to the wing spars and is used to provide the requisite stiffness and rigidity to the structure enabling it to carry different loads and forces in flight or in ground.

2.2.1 Forces and Loads

The wing spars are subjected to a wide variety of aerodynamic, structural, turbulence, gust, wind, flight and ground loads [5]. Some of the forces and loads that the wing spars carry are mentioned below,

1. As the aircraft wing rests on the ground, gravity is acting on the wing. The wing weight is been pulled down due to the gravitational forces acting on the structure and thus a bending moment is produced since, the wing roots are attached to the fuselage while, the wing tips are free. The wing spars running through the wing of the aircraft act as cantilever beams and take these bending loads. Furthermore, not only does the wing spars carry the weight of the wing while on the ground, modern commercial airplanes carry the fuel inside the wing in the wing box. Moreover, they also have mounted engines which are attached below the wing known as 'podded engines'. In fighter aircrafts, the bottom surface of the wing is used for attaching weapons, armaments or extra fuel tanks using rails or guides. So, while the aircraft rests on the ground, the wing spars are also used to carry the weight of these components hanging from the wing.

2. The primary function of an aircraft wing is to generate lift. As, at a suitable angle of attack, higher pressure exists on the bottom surface of the wing while, a lower pressure exists on the top surface of the wing, a pressure differential is created which results in generation of a differential force which is known as the lift. The lift force generated by the wings of an aircraft creates an upward bending moment. As the wing roots of an aircraft are attached to the fuselage, while, the wing tips are free, they rise upward. The wing spars are then used to resist this upward bending moment. As soon as the wing starts to generate lift, this flight load occurs that has to be carried by the wing spars. A beneficial effect of placing fuel inside the wing structure, using podded engines or extra fuel tanks on the wing tips helps in lowering the upward bending moment on the structure of the wing due to their own weight acting downwards under the action of gravity.

3. As the aircraft wing flies through the air, a drag force is generated. Drag is a necessary consequence of flight in a medium such as air since, air or any other fluid has density. The drag increases with speed and at higher Mach numbers, the drag is considerably higher. These drag loads must also be resisted by the wing spars.

4. The inertial loads must also be taken by the wing spars which act on the aircraft wing such as the rolling inertial loads, which is generated as the aircraft rolls.

5. The wing spars are under the effect of both bending and twisting moment. Due to introduction of wash-out or wash-in and aerodynamic or geometric twist, the wing spars have to carry the twisting loads. Furthermore, due to deflection of the control surfaces such as the aileron, the twisting loads are felt by the wing spars which must be resisted. Moreover, twisting loads and moments are also introduced in the structure by the introduction of podded

engines hanging below the wing. The thrust changes in these engines produce the twisting loads and moments.

2.2.2 Shapes

The wing spars can have a wide variety of shapes such as rectangular, circular, L-shaped, T-shaped etc. The wing spars are bolted, riveted or joined to the top and bottom surface of the wing.

2.2.3 Materials

A wide variety of materials can be used as the material for wing spars. Most old aircrafts typically used wooden construction. The wooden construction used for the wing spars was mostly of spruce or ash, a laminated sheet of wood. However, since, these spars are made of wood, there are under deter oratory effects over time by different environment and biological conditions such as wet and dry conditions. Furthermore, insect infestation can seriously reduce the strength of these wooden spars. Metallic materials are also used as a common material for the wing spars with aluminium sheet metal one of the common choices as the material for wing spars. Metallic fatigue over time is one of the main causes of concern for the metallic spars. However, the metallic spars have overcome some of the problems associated with the wooden spars. Modern composite materials have also been used as the material for wing spars with “fiber-glass” and “carbon-fiber sandwiched” composite structures been one of the common choices as material for wing spars. The composite material based spars have the advantage of providing good strength and leads to a reduction in weight.

2.3 Wing Ribs

The wing ribs are the forming and shaping structural member of an aircraft wing. The wing ribs provide the necessary aerodynamic shape which is required for generation of lift by the aircraft. The wing ribs are designed in the shape of an airfoil and when the wing panels or sheet are attached to the ribs gives the wing its characteristic shape. The wing ribs are attached to the wing spars and thus also provide structural stiffness as well. The wing ribs are normally placed perpendicularly in the wing but can also be placed at different angles. Normally, in modern commercial jet airplanes, the wing ribs are placed at different angles running from the wing root to the wing tip.

The wing ribs are usually made by using a truss structure, or have circular holes in placed in the sheet of the wing ribs. This is done so to lower the weight of the ribs, which in turn is helpful in lowering the weight of the wing as well. A wide variety of different manufacturing techniques are used for making the ribs of an aircraft wing.

There are different types of wing ribs characterized by the way they are manufactured for example, forged ribs, milled ribs, truss ribs [6] etc. The truss ribs are common rib structures which are manufactured by using truss like structure throughout the profile of the rib. This type of wing ribs is most commonly used for the light-weight and other smaller aircrafts. Forged ribs are manufactured by the use of heavy-press machinery to get the rib shape, however, significant after treatment is required in order to smooth out the edges and the

curves. Depending on the position and the loading condition on the wing of an aircraft, different types of wing ribs are used in different sections of the aircraft. For example, due to heavy loading condition during takeoff and landing prevalent in the section where the landing gear is mounted on the wing, forged ribs are used to provide the necessary strength and rigidity to the structure. Milled ribs are also used in the similar type of loading condition especially at the landing gear region on the wing. The milled ribs are manufactured using a single piece of metal where the material is removed by use of milling techniques. Lighter weight ribs are typically use outboard of the wing structure towards the wing tips. The wing ribs are attached to the wing panel sheet by using riveting, bolting or other joining techniques using adhesives and glues. Since, the wing ribs are critical in given the wing structure its characteristic airfoil shape that is necessary for the generation of the lift by the aircraft wing, great care is taken in the manufacturing of the wing ribs and the goal is to match the profile of the wing as accurately as possible.

3 Generic Aircraft Structural Wing Design Concept

3.1 Aircraft Design Process

Aircraft design process is a complex undertaking, however, the design process can generally be divided into three phases which are outlined in the figure below. There is a certain amount of overlapping between these three phases and the number of people, resources and cost associated with the design gradually increases between these phases [9]. The different stages of the aircraft design process are,

1. Conceptual Design
2. Preliminary Design
3. Detail Design

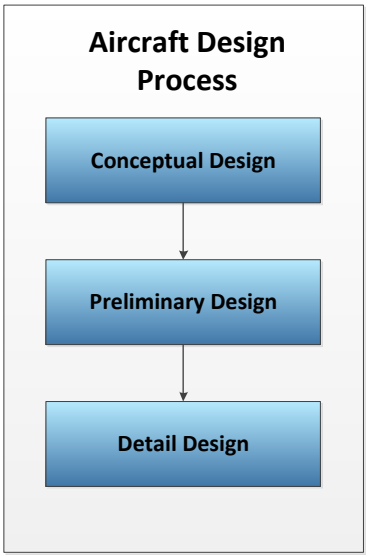


Figure 5: General Overview of Aircraft Design Process

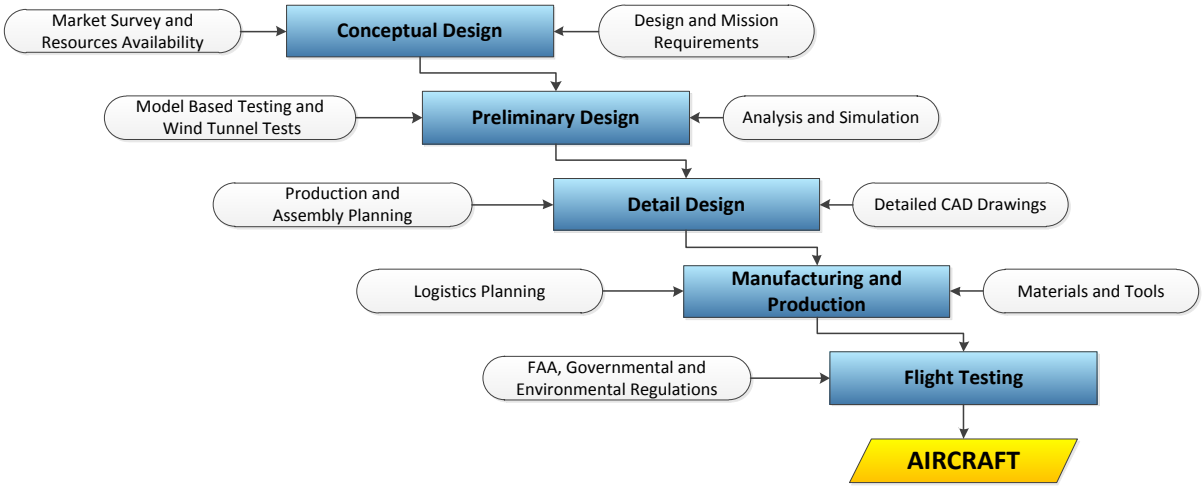


Figure 6: Detail Overview of Aircraft Design Process

The conceptual design phase is the most fundamental stage of the design process where a small number of people are involved. The primary aim of the conceptual design phase is to come up with a design that is able to meet or exceed the design requirements. The conceptual

phase of design is driven by mission and design requirements; furthermore, it is also influenced by resource availability and market factors. Here, in this phase, the designers have a wide variety of options available to them for choosing the external shape of the aircraft. Thus, in this phase, a large number of alternative aircraft configurations are studied and feasibility studies are done. The designers normally look at both conventional and unconventional designs at this stage, and also compare their designs with existing aircrafts. The freedom for design modification especially to the external shape decreases as the design moves from the conceptual to the detailed phase. The conceptual phase of design is important because it lays the foundation on which the entire aircraft would be built. It is of utmost importance that the design that is come up in the conceptual phase is feasible and is a design that is able to be manufactured and developed. Normally, at this stage of the design process, statistical methods are common that drive the design. No sophisticated and detailed CAD models are developed at this stage for the design. The amount of resources and cost is also quite less as compared to a detailed design phase where, the manufacturing of the prototype and product takes place and a large number of people are involved.

After the conceptual design phase, a single or a few suitable candidates are chosen for the preliminary design. The preliminary design is an advanced stage of the conceptual design, where, there is a certain overlap between the two phases. The goal of the preliminary design is to completely fine tune the external geometry of the aircraft. In order to do this, detailed modelling and simulation studies are carried out which look at each specific characteristic of the design in detail. The details of all sub-systems and components are worked out. In this stage, many experimental and computational or numerical methods are used for simulation and analysis of the design. Flow analysis using wind tunnels and structural analysis using FEM (finite element methods) are also performed. A detailed comparison between the different alternative configurations are performed based on a number of parameters including cost, performance, fuel efficiency, range, speed, payload, marketability, lifetime and manufacturability amongst many others. A detailed cost and market analysis is also carried out to fine tune the region and the type of role that the aircraft would be used for. The outcome of the preliminary design process is a final aircraft configuration which is said to be "frozen" which means that no further design modifications will take place relating to the geometry and shape of the aircraft. Only small minor changes to the design can be made. The design is frozen at this stage so as to avoid significant penalty in terms of cost, time and resources for any design modification in a detailed design phase which would eventually influence a large number of sub-systems. The conceptual and the preliminary design are critical in what eventually becomes an aircraft. A large number of specialists are involved in this design phase and every aspect of the design is worked out.

The aircraft design eventually runs into the detailed design after passing the phases of conceptual and the preliminary design. The detail design is the final phase of an aircraft design process, where, each and every component of the aircraft is designed in detail and manufactured. All manufacturing drawings and methods are developed and used for the manufacturing and assembly of the aircraft. A very large number of people are involved in this stage not only specialists but other technicians as well. The amount of cost and resources

are also quite significant. It is of utmost importance that all flaws in the design are ironed out in the preliminary or conceptual design phase, because, any fault found in the detailed design phase can put the design in jeopardy which can lead to the aircraft design project been cancelled or impacting this process with significant cost and time penalties. All manufacturing details, CAD geometry and models for all including the smallest of components are worked out for the design that is frozen at the end of the preliminary stage. In the detailed design process, precision and accuracy is very important and a range of tools and methods are used at this stage.

After the detailed design phase, the manufacturing and production of the prototype takes place. This involves lots of logistical planning and a large number of materials and tools for the actual manufacturing and assembly. Once, the aircraft prototype is manufactured, the prototype runs into the flight testing phase, where, the aircraft goes through a series of flight and ground test to gauge the performance of the aircraft. This phase is influenced by aviation authorities like FAA, Governmental and environmental agencies, and all the requirements set out by these agencies must be met.

3.2 Generic Aircraft Wing Concept

A generic aircraft wing model should constitute an external surface of the wing, and should also be comprised of structural elements of the wing which are spars and ribs. In order to define accurately the external shape of the wing, a number of parameters are required to define the geometry of the wing in detail. The generic aircraft wing model should comprise of both a surface model and a solid model of the geometry of the wing. Both of these models should be integrated together, which means that the parameters should change both the surface and the solid model of the aircraft wing simultaneously. The generic wing model should be able to transform into any wing planform and shape based on the changes in the wing geometry parameters.

3.2.1 Surface and Solid Model Integration

In the surface model, all the features of the wing (wing panels, spars and ribs) will be represented by a series of surfaces. As, the number of wing panels, spars or ribs are changed, the surfaces for each wing panels, spars and ribs will be connected together. The surface model is made because of its use in carrying out the structural analysis of the wing where, the surface model will be utilized to do the mesh generation of the entire aircraft wing geometry.

The solid model will have thicknesses attach to each wing panel, spars and ribs. The solid model can be used for the design purposes, but it is important that both the surface and the solid geometries are properly integrated inside the same wing model.

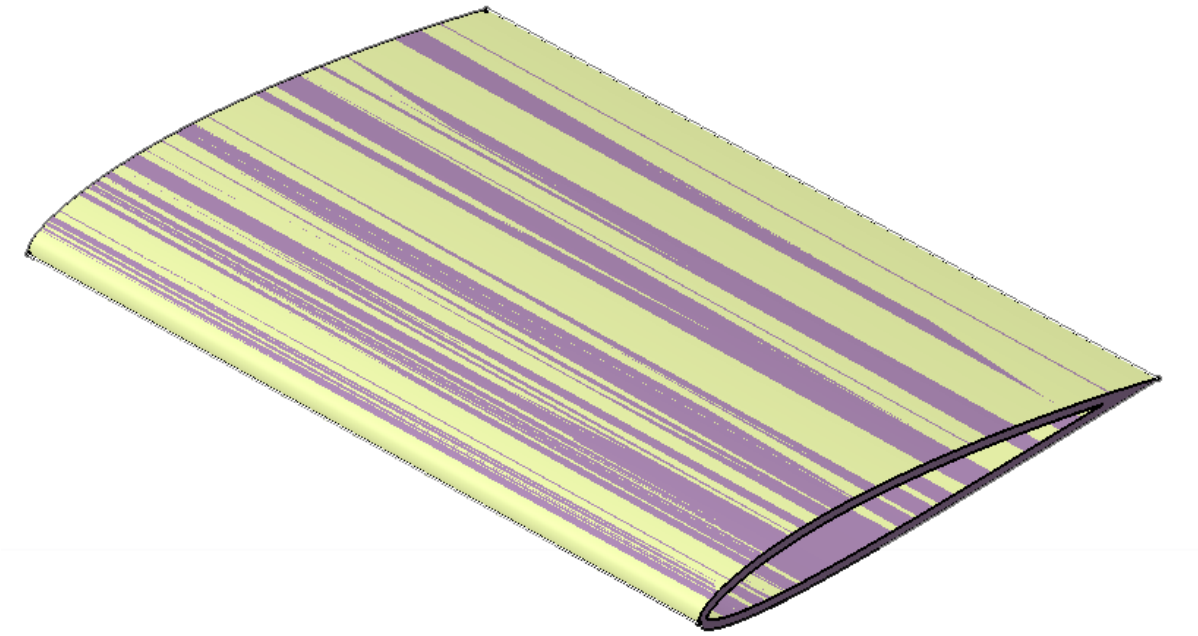


Figure 7: Surface Model and Solid Model Integration in Generic Wing Model (Yellow is Surface, Purple is Solid)

3.2.2 Concept for Wing Panels

The entire wing panels of the aircraft wing should be properly connected, which means that all the wing panels should be connected in such a way so as to yield a continuous surface of the wing. Furthermore, the geometric features associated with the tip of the wing panel should be the same as the root of the next wing panel, so as the geometric features e.g. tip chord length or tip airfoil associated with the tip of the wing panel are changed, the geometric features of the root of the next wing panel should updated accordingly.

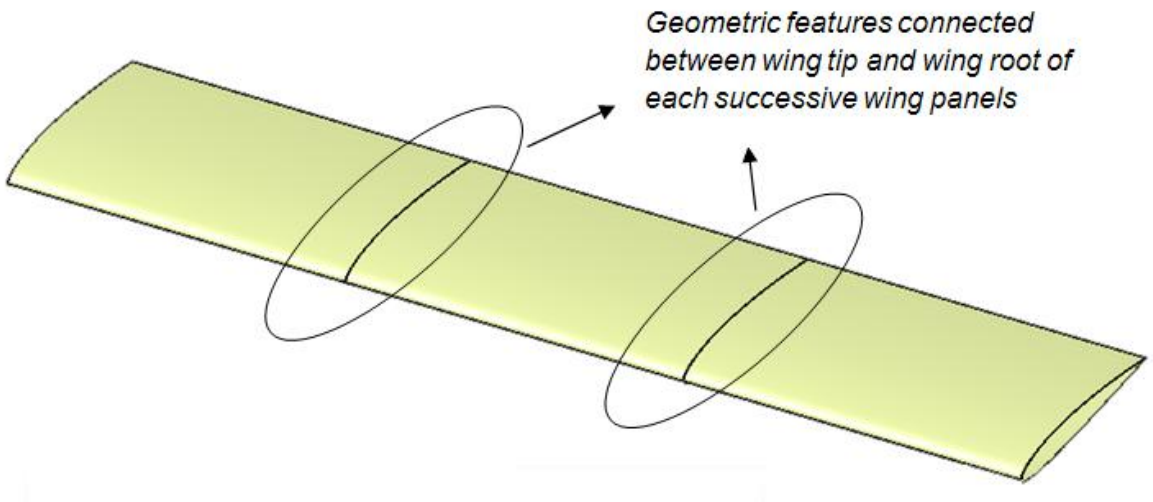


Figure 8: Geometric connection between different wing panels

3.2.3 Concept for Wing Spars

As far as the wing spars are concerned, the wing spar position will be defined by "point values on curve" along the root and tip of the tip of the wing. The point values on curve will range between 0 and 1. 0 means that the spar position will start at the leading edge while, 1

means that the spar position will start at the trailing edge. There are two approaches for the construction of wing spars inside the wing model. One is that the spars run continuously throughout the wing across all the wing panels while the other is that the wing spars are placed along each wing panel separately and then they are joined together to each other. In either approach, it is important that it is not possible for two spars to intersect each other in any way. Furthermore, it is important that, all new spar positions along the root and the tip of the wing should be modified based on the position of the old spars. The wing spars will have a thickness associated with them, however, this thickness should not protrude inside the thickness of the wing panel skin.

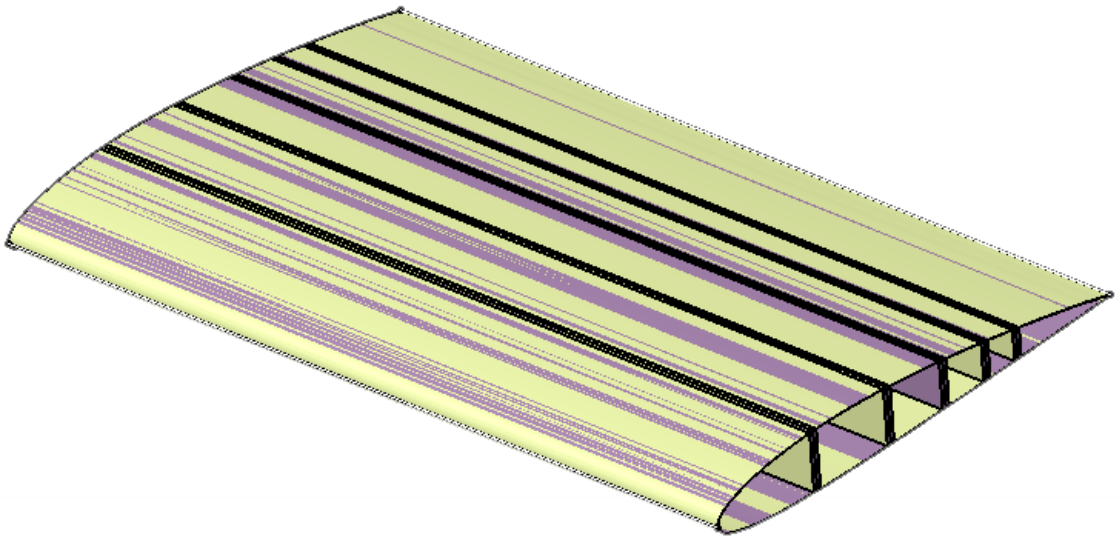


Figure 9: Wing spars inside the generic wing model

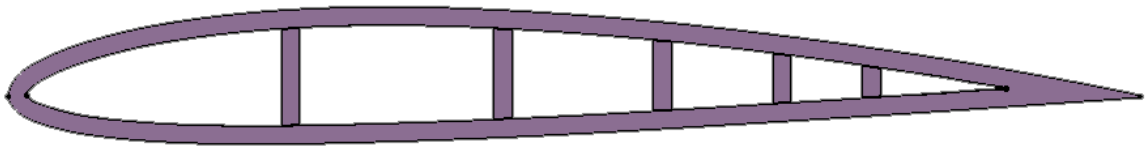


Figure 10: Wing Spars Thickness should not protrude inside wing panel skin thickness

3.2.4 Concept for Wing Ribs

Similar to the case of the spars, the wing rib positions will be defined by "point values on curve" along the span of the wing (front and aft curve of the wing). For the surface model, the rib will be placed perpendicularly or an angle and will just be composed of surfaces, while, for the case of solid model, it is important that the ribs are divided based on the number of spars present in the generic wing model. For example, if there is one spar present in the wing, the wing rib should be divided into two. If there are two spars present in the geometry, the rib is divided into three and so on. It is also important that no ribs should intersect each other. Similar to the case of spars, the ribs will have a thickness associated with them. It is

important that the rib thickness doesn't protrude the spar thickness neither should it protrude the wing thickness associated with the skin of the wing panels.

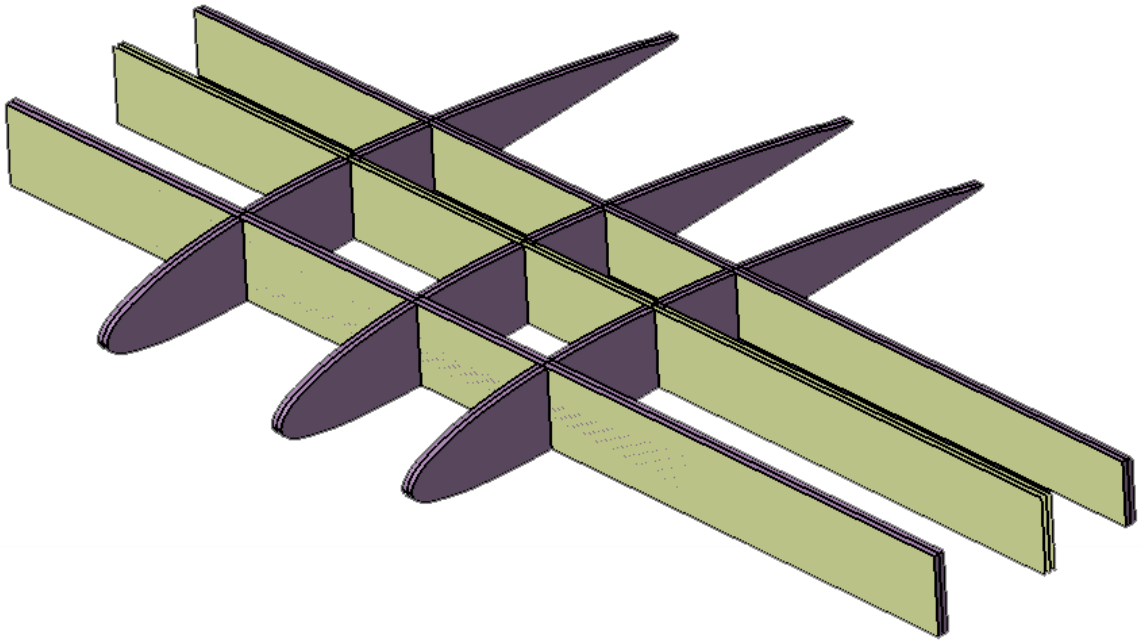


Figure 11: Division of wing ribs based on the number of spars in generic wing model

It is also to be ensured that a wing rib can be placed across multiple wing panels, as an example, in the figure below; a wing rib is placed across two wing panels. The wing ribs can also be placed at any angle inside the generic wing model and no wing ribs should intersect each other.

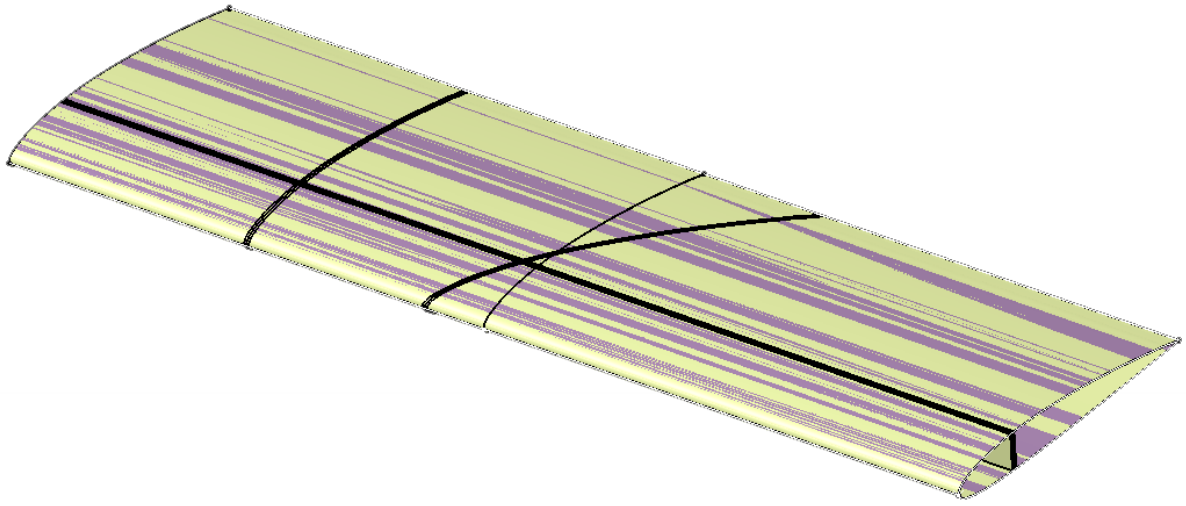


Figure 12: Wing Rib placed across multiple wing panels

3.2.5 Wing Design in Generic Model

The generic wing model will be composed of two airfoil sections for each wing panel, one for the root and the other for the tip. The airfoil of the wing will be defined by using a series of

points or coordinates connected to each other with a spline running through them as shown in the figure below.

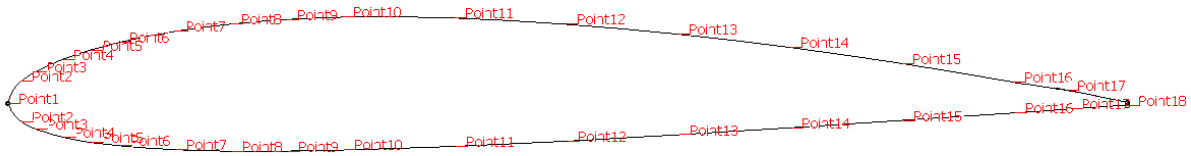


Figure 13: Point coordinates defining the shape of the airfoil

These airfoil coordinates can easily be obtained by using an airfoil coordinate database [3]. It is important that a sufficient number of points are defined for both the top and the bottom surface of the airfoil to correctly define its shape, which will in turn correctly define the shape of the wing as well. A very large number of point coordinates can be used to define the shape of the airfoil, however, in order to make it practical for geometrical purposes, the number of point coordinates for the airfoils of the wing is set to 35 (17 points on the top surface of the airfoil and 17 points on the bottom surface of the airfoil and one point for the origin). The chosen number '35' accounts for the number of point coordinates that are used to define the shape of old NACA airfoils like NACA 2412 etc, which is taken as a reference. Since, a spline will pass through these points, it is noted that these 35 points are sufficient to accurately define the shape of most types of airfoils up to a level of detail that is fit for geometrical purposes.

The generic aircraft wing model will have a variety of parameters that define the geometrical characteristic, planform and shape of the wing including parameters for geometrical and aerodynamic twists. A general wing shape e.g. a shape of a propeller or a turbine blade can have twists and rotations in all the three axis (x, y and z). In order to make the aircraft wing model as general as possible, it should be possible to introduce rotations to the root and tip sections of the wing in all the three axis (x, y and z). This means that the root and the tip of the wing can be independently rotated or twisted in the x, y and z direction. Furthermore, in the aircraft wing, the point of rotation is also important. Sometimes, the aircraft wing is rotated along the quarter-chord point, sometimes along the tip and sometimes along the root of the wing. A parameter must be defined whose value will dictate the point of rotation along the root chord or the tip chord of the wing. As with the other rotations, the point of rotation can be independently set for both the root chord and the tip chord of the wing.

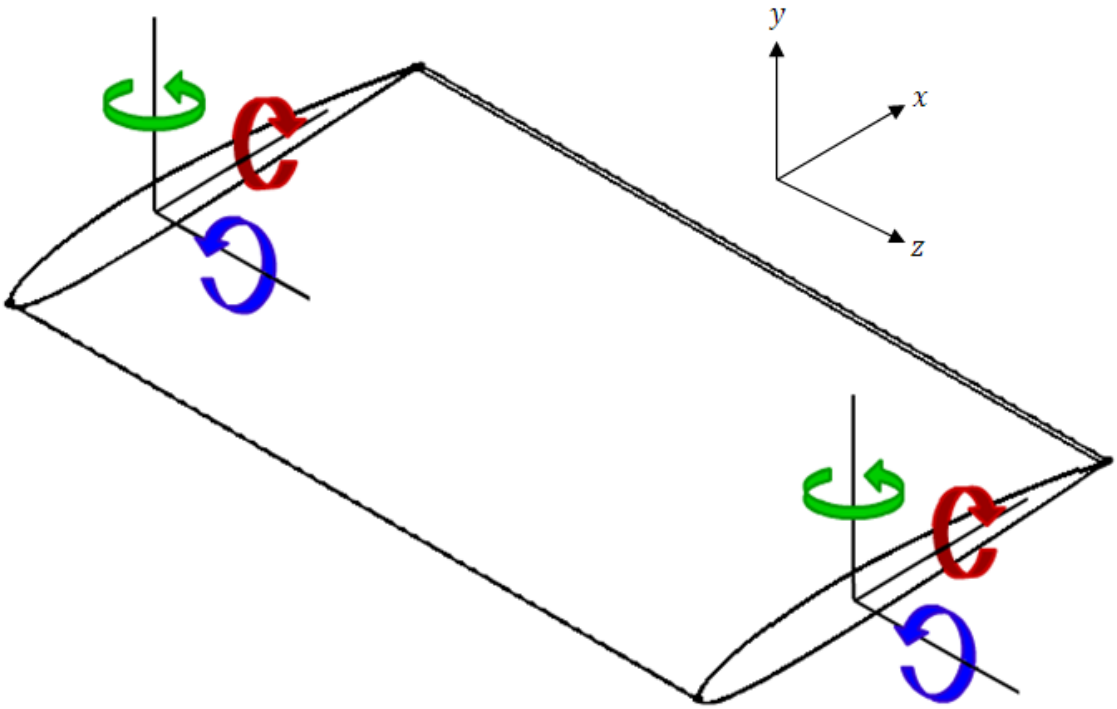


Figure 14: Root and Tip chord of the wing can be rotated along the x, y and the z-axis

4 Approach and Implementation

4.1 CAD Software

The CAD (Computer Aided Design) software that is used during this master thesis work is the Dassault Systems CATIA V5. CATIA is high-end CAD software which provides a variety of advance tools and options for geometry generation and CAD automation. CATIA has seen wide spread use in the aerospace industry and it is been used by major manufactures like, Boeing, Airbus and SAAB amongst others. One other advantage that CATIA provides is ability to automate the CAD design by programming in Visual Basic or by introducing knowledge in the model through built-in tools. Thus, sophisticated macros, code and rules can be written defining the different aspects of the CAD geometry. This is perfect for the present case, where, these features will be extensively use to create the aircraft wing generic model and all codes, rules and macros will be defined within it. Furthermore, CATIA also offers structural analysis capabilities within the same product. This is required as both the geometry creation and mesh generation for FE analysis should be integrated together. This can be achieved well by choosing a software system which offers both capabilities which is the reason for choosing the CATIA V5 software.

4.2 Parametric CAD Modelling

Modern CAD software's allow for different levels of parameterization. Furthermore, they also offer a wide variety of automation capabilities. A schematic representation of the different levels of parameterization is shown in figure which is adapted from [1].

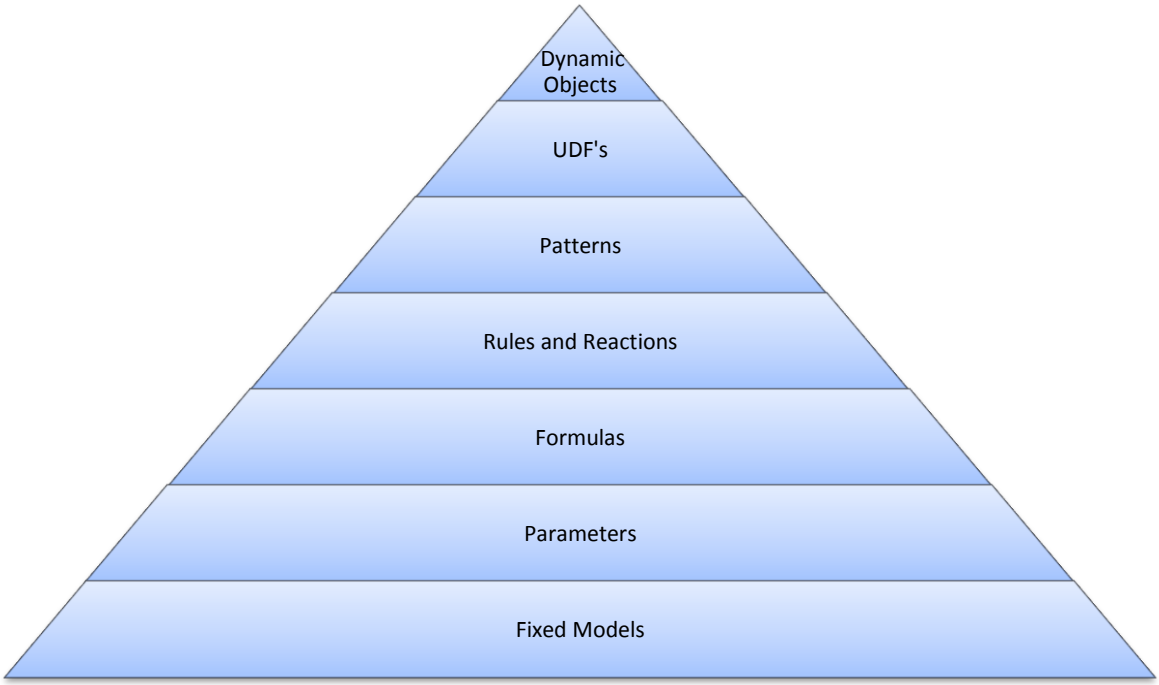


Figure 15: Parametric CAD modelling breakdown (figure adapted from [1])

4.2.1 Fixed Models

Fixed models represent the lowest level of parameterization. In order to modify a geometrical object in fixed models, the object needs to be opened so that the geometrical property is located and is modified. A simple example of this level of parameterization can be a length of some fixed model or geometry. In order to modify this length, the object is required to be opened and its property can then be modified.

4.2.2 Parameters

Parameters represent a higher level of parameterization than fixed models. In this case, the parameters can be used to define the geometrical property outside the object. When the parameter value is changed the geometry associated with this parameter will change also. For example, if the length of a certain geometrical object is define as a parameter then when the parameter is changed, the geometry will reflect the change that has occurred in the parameter.

4.2.3 Formulas

Formulas are used to define mathematical relationship between the parameters or geometrical properties. For example, if the length of certain geometry is changed, then its area will change also. The information regarding the area of the geometry can be defined as a formula in this case.

4.2.4 Rules and Reactions

Rules and Reactions are higher level of parameterization which can be used to introduce the design knowledge into the geometry [2]. Rules are always active in the geometry and can be used to describe the relationships between the geometrical objects, parameters and properties etc. Reactions on the other hand are triggered once they are activated by a given input. Reactions can be used to define the response of the geometry based on a certain trigger or event. Rules and reactions are both handy tools in regard to the automation and the modification of the geometry.

4.2.5 Patterns

Patterns are geometrical copies of a certain geometry that is instantiated. A pattern repeats the same geometry over and over again. The elements of the pattern can't be modified individually. Patterns allow for dynamic instantiation of the geometry as many times as required. One another disadvantage is all elements of the pattern exist as a single entity however, the advantage is that they can be dynamically instantiated by using parameters.

4.2.6 PowerCopy and UDF

PowerCopies and UDF's are ways to introduce knowledge in the geometry. One differing aspect is that unlike the patterns, powercopies and UDF's are context depending, which means that the geometry of the object changes in the context in which it is instantiated. This means that each instantiation is a separate entity on its own which can be modified. However, unlike patterns, the powercopies and UDF's can't be instantiated dynamically.

4.2.7 Dynamic Objects

General dynamic objects are achieved by adding scripts to UDF's [2]. Thus, in this way, dynamic objects are able to combine the benefits of dynamic patterns, context dependency and object instantiation of power copies or UDFs [1]. General dynamic objects represent the

highest level of parameterization which can derive complex geometry which can be automatically and dynamically instantiated in the correct context. Knowledge pattern is an example of general dynamic objects in CATIA V5 which can be used to drive geometries which are automatically and dynamically instantiated.

4.3 Tools and Methods

A breakdown of the tools and methods used in this thesis work is given in the figure below,

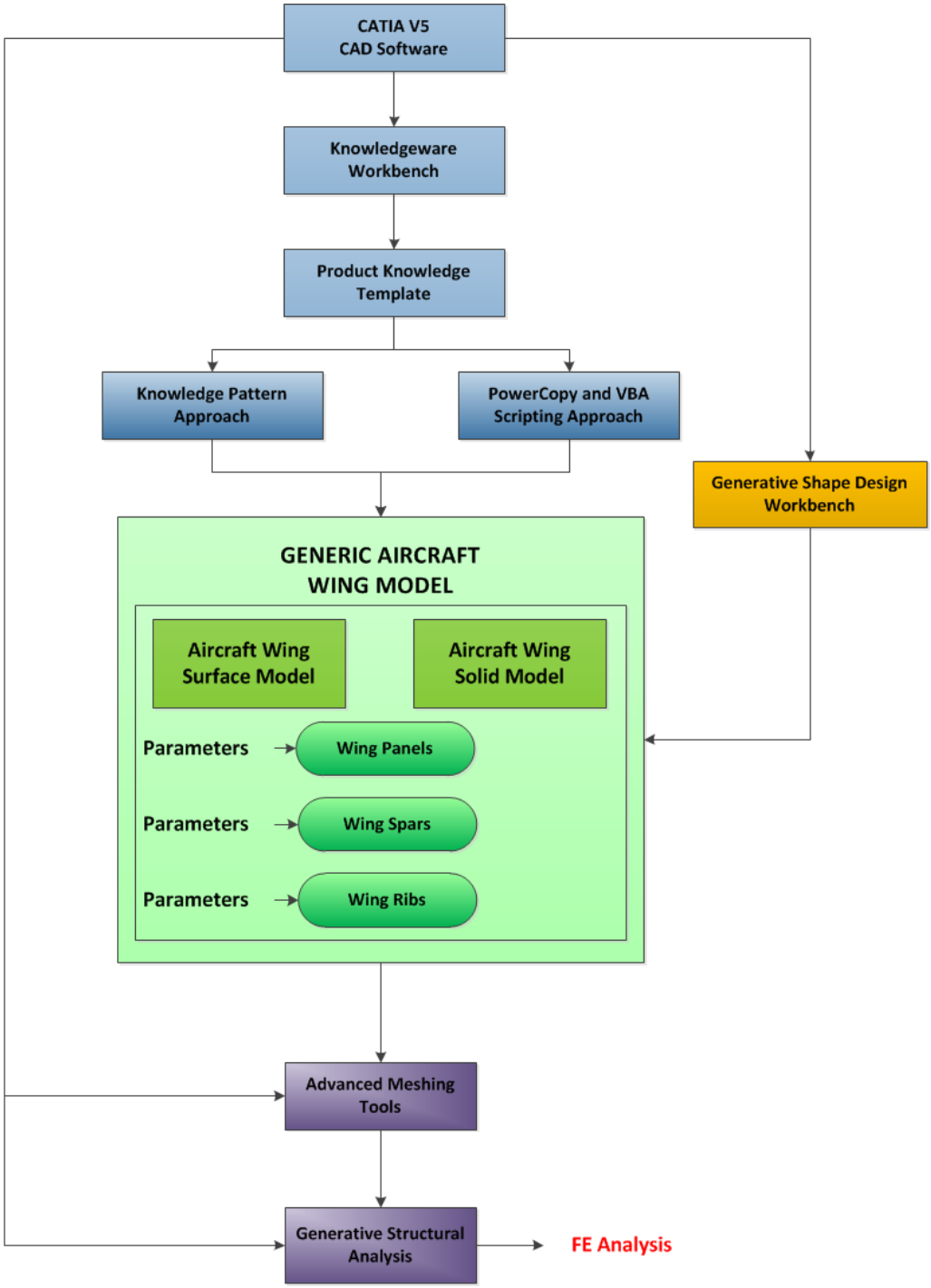


Figure 16: Tools and Methods used for creation of generic aircraft wing model

In the CATIA V5 software, there are a number of workbenches available that can be used for geometry generation and automation. In order to introduce knowledge in the generic model, and for the automation and management of the geometry, the tools available in the "Knowledgeware Workbench" and "product knowledge template" are used. There are two approaches that are adopted in this thesis work, which is knowledge pattern and powercopy with VB scripting. Both of these are implemented by using this workbench and also by using Microsoft Visual Basic IDE environment for debugging Visual Basic code in VB scripting.

Furthermore, since, the model requires sufficient surface modelling; a specialist workbench for surface modelling, "Generative Shape Design" is used. The generative shape design offers a wide variety of features and tools for advance surface modelling.

Thus, the generic model of the wing is completely developed in CATIA V5 including wing spars and wing ribs. Both the surface model and the solid model, which are part of the generic wing model, are integrated together. A number of parameters define the shape of the wing panels, wing spars and wing ribs.

Since, the automated mesh generation of the generic wing model geometry is also required; the tools available in the "Advanced Meshing Tools" workbench are used to integrate the wing model with the meshing program. As the generic wing model changes, the mesh attached with the generic wing model will change also. The tools available in the "Generative Structural Analysis" workbench can then be used to perform the structural analysis on the aircraft wing model.

4.4 Advantages and Disadvantages of Different Methods

The methods used in this master thesis work are,

1. Knowledge Pattern

2. PowerCopy with VB Scripting

A discussion about the advantages and the disadvantages offered by these methods are presented below.

4.4.1 Knowledge Pattern

Knowledge Pattern is part of the "Knowledgeware" and "Product Knowledge Template" workbenches in CATIA. Knowledge Pattern Language (KPE) is somewhat similar to the KWA language used in rules in CATIA.

4.4.1.1 Advantages

1. Less amount of scripting is required for automation of geometry as compared with other approaches like VB scripting.
2. The concept of lists and management is built in the knowledge pattern.
3. Deletion of the geometry is taking care automatically, as compared to the VB scripting where, all the individual elements of the geometry, rules and formulas are needed to be selected before deletion.

4.4.1.2 Disadvantages

1. There are no options available in the Knowledge Pattern environment to place stops in the code for debugging. This makes debugging a knowledge pattern code time consuming. Furthermore, there is no option available for undoing a previous edition.
2. Pasting code from a text file into Knowledge Pattern environment often leads to syntax errors generated, so some parts of the code has to be rewritten again in the knowledge pattern environment.
3. Knowledge Pattern programming cannot be done on standard programming editors likes Visual Basic.
4. Knowledge Pattern is unable to access other programs like Microsoft Excel and Notepad, and thus it is difficult to integrate Microsoft Excel and knowledge pattern through macros or coding.
5. Error checking functionality is not available in the Knowledge Pattern (KPE) language.
6. The functionality and functions are limited only to geometry generation and its maintenance as compared to Visual Basic.

4.4.2 PowerCopy

Power Copy is a geometric creation feature available in CATIA V5 software that is used for the instantiation of the geometry. PowerCopy is also part of the "Knowledgeware" and "Product Knowledge Template" workbenches in CATIA.

4.4.2.1 Advantages

1. To create a power copy doesn't require any scripting, (however, the automatic instantiation is required than it requires VB scripting).
2. The power copy contains all the geometric features contain within it, and once it is instantiated all the features are visible and can be modified.

4.4.2.2 Disadvantages

1. Once the power copy is instantiated, there is then no link between the original geometry from which the power copy was created and the instantiated geometry. This means that any changes made to the original will not affect the instantiated geometry in any way. The instantiated geometry is totally unaware of any changes to the original. So, in order to reflect the change, the instantiated geometry needs to be re-instantiated.

4.4.3 Visual Basic (VB) Scripting

The Visual Basic scripting for geometry automation or modification can be done within CATIA as well as by using Microsoft Visual Basic Editor or IDE (Integrated development environment).

4.4.3.1 Advantages

The visual basic is a comprehensive programming language. The Microsoft Visual Basic Editor can provide help on the syntax. Furthermore, the API/Syntax is well known and documented.

The concept of references libraries are built in the VBA-IDE which means that a large number of applications API's can be used in connecting Visual Basic with another application or software. For example, by choosing the references library of CATIA in VBA-IDE, the Visual Basic development environment is connected with the CATIA software and code can be rewritten for automation.

The code that is required to be run in a reaction can be written inside the VBA-IDE, where, the visual basic code can be debugged. It can then be copied back to the reaction where, it will be eventually use. However, since, there is a capability of debugging available, it makes the entire process of coding much easier, and also the code develop can be made much more robust and reliable, by performing a wide variety of tests and scenario's.

One of the good advantages is that the Visual Basic code can be run across all the workbenches available in CATIA software which is not the case for the Knowledge Pattern

Using the concept of macro recording built inside CATIA, Visual Basic scripting code can be extracted for almost any type of operation e.g. split, join, trim etc. This helps tremendously in speeding up the code generation process.

4.4.3.2 Disadvantages

One of the disadvantages of the VB scripting is that, it sometimes required to write syntax overhead for variables and objects, e.g. when defining a new integer in the code, it is always required to enter "Dim int1 As Integer" and so on.

4.5 Implementation

4.5.1 Methodology

The methodology adopted for the creation of the generic wing model for both the "knowledge pattern" and "powercopy with VB scripting" approaches is shown in the figure below,

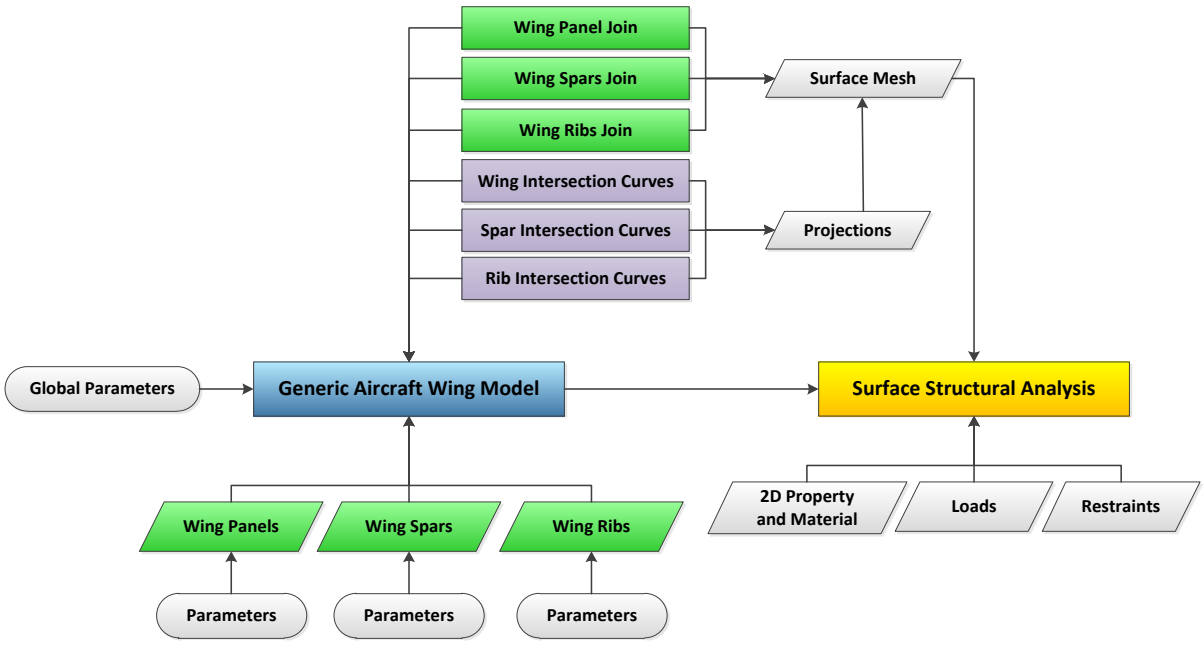


Figure 17: Methodology for generic aircraft wing model

The generic aircraft wing model is composed of both the surface and the solid model for wing panels, wing spars and wing ribs. Each wing panel, wing spar and wing ribs also have individual parameters that define the geometry and shape of each element, furthermore, there are also global parameters which control the number of wing panels, spars and ribs as well as the mesh characteristics. Whenever, a new wing panel, a wing spar or a wing rib is added into the model, a join which already exists in the model is updated with the new geometry. These joins are connected to each individual surface mesh for wing panels, wing spars and wing ribs. In order to ensure, that all the mesh elements are properly connecting at the nodes of the aircraft model, projections are required for generating the mesh in the correct fashion. This is done by defining three intersections between the join surfaces. These are Intersection between Wing and Spars, intersection between wing and ribs and intersection between spars and ribs. These projections are then used to correctly define the surface mesh for wing panels, spars and ribs. In order to perform the structural analysis, 2D properties, materials, loads and restraints are required. A 2D property adds thickness to the surface. Different types of loads and materials can be applied to the wing structure. The 2D property for each surface is linked with a material as well.

5 Generic Aircraft Structural Wing Model

The generic aircraft structural model is developed by knowledge pattern using EKL (Engineering Knowledge Language), Visual Basic scripting and geometry automation tools and features that are available in CATIA V5 CAD software. The generative structural analysis and Advanced meshing tools workbenches available in CATIA V5 software are used to perform the structural analysis and automated mesh generation of complete aircraft wing geometry. The aircraft structural wing model is made up of different elements whose number, shape and geometry can be changed by a range of different parameters. The aircraft wing model includes,

1. Wing Panels
2. Wing Spars
3. Wing Ribs

5.1 PowerCopy with Visual Basic Scripting Approach

In the powercopy approach, in order for the management of wing panels, wing spars and wing ribs the following parameters are introduced. These parameters are,

1. New Number of Wing Panels
2. Old Number of Wing Panels
3. New Number of Spars
4. Old Number of Spars
5. New Number of Ribs
6. Old Number of Ribs

These parameters receive integer inputs. When the “New number of Wing Panels” parameter value is changed, it triggers a reaction which results in either more wing panels been added to the model or been deleted from the model. When the new wing panel is introduced in the model, it automatically attaches with the old wing panel through a “join” surface. Furthermore, some formulas are also created as result of new wing panel addition which creates relation between the parameters from the old and the new wing panels. When the “Old number of Wing Panels” is larger than the “New Number of Wing Panels”, the wing panels along with relations, formulas and joins are deleted.

When the “New number of Spars” parameter value is changed, a reaction is triggered which causes either new wing spars to be added or deleted from the model. As, with the case of wing panels, all new spars are connected with the previous spars by a “join” surface. However, in this case, they need not be connex. All new spars position is dictated by the position of the previous spars. The new spars are added in such a way that it is not possible to have an intersection between any two spars. The spars run throughout the wing panels across the entire wing span of the wing. Similar, to the case of wing panels, when the “New number of Spars” is smaller than the “Old number of Spars” the spars are deleted.

When the “New number of Ribs” parameter value is changed, a reaction is triggered which causes either new wing ribs to be added or deleted from the model. As, with the case of wing panels, all new wing ribs are connected with the previous ribs by a “join” surface. This is done so, because the surface mesh will be attached to these “join” surfaces which when the geometry of the wing is modified. Similar, to the case of the ribs, the new ribs are added in such a way, that there is never a possibility for two ribs to intersect with each other. The primary reason for doing this is to avoid problems with mesh generation. When the “Old number of Wing Ribs” is larger than “New number of Wing Ribs” the wing ribs are deleted.

The minimum number of Wing Panels, wing spars and wing ribs in the aircraft structural wing model can be one. This is done so, because the surface meshes that are used for automated generation of the mesh for structural analysis are attached to these surfaces. Deletion of any wing panel, spar or rib to zero will cause the surface mesh to be lost and mesh generation will then not be possible. One other supporting point for this, comes from the observation that any wing model that should have at least one wing panel, one spar and one rib, which is the case at present.

There is no upper limit to the number of Wing panels, wing spars and wing ribs. This ensures that a wide variety of wing configurations can be developed e.g. an elliptical wing or a wing with wing tips.

Some other dedicated parameters are defined which act as a trigger for different reactions for wing panels, spars and ribs. However, the user is not required to change their values as these are set automatically by the code.

5.2 Knowledge Pattern Approach

Since, in the knowledge pattern, the concepts of list and management are built in the model, less number of parameters is required to control the number of wing panels, spars and ribs. The parameters that control the number of wing panels, spars and ribs are,

1. New Number of Wing Panels
2. Old Number of Wing Panels
3. New Number of Spars
4. New Number of Ribs

When any of the above parameter value is changed, the knowledge pattern code is triggered which in turn adds or deletes wing panels, spars and ribs depending on as the case may be. There is only one knowledge pattern relation that manages the automation of the aircraft wing structural model.

When the “New Number of Wing Panels”, “New Number of Spars” or the “New Number of Ribs” parameter values are changed, the knowledge pattern code runs and creates a “join” surfaces for wing panels, spars and ribs. Thus, all wing panel surfaces are joined with each other in one join, similarly, all spars are joined in one surface, and all ribs are joined with

each other in one surface but they need not be connex. These surfaces are connected with the surface mesh for wing panels, spars and ribs, which ensures that as the geometry of the wing is changed, the automated mesh should reflect the new geometry that is been created.

As, with the Power Copy Approach, the minimum number of wing panels, wing spars and wing ribs can be one while, there is no upper limit on the number of wing panels, wing spars and wing ribs, thus, ensuring that a wide variety of wing configurations can be designed and studied.

5.3 Wing Panels

The wing panels define the external shape of an aircraft wing. They are basically the skin of the aircraft wing with a specified thickness attached to them. In order to accurately define the external shape of the aircraft wing, a large number of parameters are required. These large numbers of parameters are required so because an actual wing can have a wide variety of twist, sweep, dihedral, thickness and chords etc. Depending on the shape of the aircraft wing to be designed, an aircraft wing can have a large number of wing panels. A single wing panel is of trapezoidal shape, however, with a large number of wing panels any shape of the wing can be achieved e.g. an elliptical wing or a double tapered wing etc. If the aircraft wing has more than one wing panel, they are joined together at the tip and root of each successive wing panels. This ensures that two wing panels are connected to each other geometrically and are connex.

Each wing panel is defined by 24 parameters which are defined below with the type and unit of the parameter is shown in the parenthesis.

1. Root Airfoil (string)
2. Tip Airfoil (string)
3. Wing Panel Span (m)
4. Root Airfoil Chord (m)
5. Tip Airfoil Chord (m)
6. Wing Panel Leading Edge Sweep Angle (deg)
7. Dihedral Angle (deg)
8. Root Airfoil Rotation Point along chord w.r.t. z-axis (null)
9. Tip Airfoil Rotation Point along chord w.r.t. z-axis (null)
10. Root Airfoil Rotation Point along chord w.r.t. y-axis (null)
11. Tip Airfoil Rotation Point along chord w.r.t y-axis (null)
12. Root Airfoil Rotation w.r.t. z-axis (deg)
13. Tip Airfoil Rotation w.r.t z-axis (deg)
14. Root Airfoil Rotation w.r.t x-axis (deg)
15. Tip Airfoil Rotation w.r.t x-axis (deg)
16. Root Airfoil Rotation w.r.t y-axis (deg)
17. Tip Airfoil Rotation w.r.t y-axis (deg)
18. Wing Panel Area (m^2)
19. Taper Ratio (null)
20. Mean Aerodynamic Chord (m)

21. X position of the Mean Aerodynamic Chord (x_MAC) (m)
22. Y position of the Mean Aerodynamic Chord (y_MAC) (m)
23. Aspect Ratio (null)
24. Wing Panel Skin Thickness (m)

For any aircraft designer, an estimate of the wing area is essential for e.g. computing the lift and drag of the wing or for calculating moments along the quarter-chord point of the wing. This is catered for in the aircraft wing model, where, a projected surface is shown directly below the wing panels which show the wing area for each panel. The position of the Mean aerodynamic chord of each wing panel is also marked on the projected surface of the wing panel as well as the 25% and 50% chord lines. The sums of the projected areas of all the wing panels are added in a single parameter which gives the total area of the wing. Furthermore, the Mean Aerodynamic Chord of each wing panel is calculated individually, and then in a global Mean Aerodynamic Chord parameter, the Mean Aerodynamic Chord (MAC) of the entire wing is calculated which is quite useful to calculate the MAC for complex geometries of the wing.

A detail description of aircraft wing panel parameters in the generic model is presented in Appendix B.

5.4 Wing Spars

The wing spars are the main load carrying member of the aircraft wing. Not, only do they provide rigidity and support to the wing structure but also are used to mount podded engines in commercial jet aircrafts and weapon systems and fuel tanks in a wide variety of fighter aircraft. The wing box which normally carries most of the fuel required for the aircraft to perform its mission is also attached to the spars. The wing spars also act as supported structures where, the landing gears in e.g. commercial jet aircrafts are mounted. Since, the wing spars are the main structural load carrying element of the aircraft wing; they are subjected to a wide variety of loads. A wide variety of structural, aerodynamic, gust, landing, takeoff, turbulence and gravity loads amongst others affect the wing spars. Furthermore, the wing spars should have sufficient rigidity, flexibility and elasticity to be able to withstand the loads during the flight. Fatigue and cracks in the wing spars structure is also a major cause of concern which has to be dealt with detailed analysis, simulation and testing. The spars are placed along the wing span of the aircraft wing. The spars that are designed in this framework are continuous, which means that the entire spar is a single piece and runs continuously between the wing root and wing tip of each wing panel. Each wing spar has a thickness parameter associated with them which controls the thickness of the spar.

Each wing spar is defined by the following parameters. These includes,
1. Spar Position P1
2. Spar Position P2
3. Spar Thickness

On each wing panel, a spar has two positions one on the root chord and the other on the tip chord. The root chord position is given by “Spar Position P1” and the tip chord position is given by “Spar Position P2”. The spar positions can be set between 0 and 1 for each case. Each subsequent spar will be placed between the previous spar and the end point of the chord or the tip.

Each spar position is defined by using a dedicated naming sequence. As an example, let’s consider the name of parameter “Spar Position P1.23”. The P1 in the name represents that the point is in the root chord. The second digit “2” represents that the spar position point is of the second wing spar and the third digit “3” in the name represents that the point is in the third wing panel. So, to conclude, the second digit represents the number of spar and the third number represents the number of wing panel where the spar position point is located.
More spar position points will automatically be created based on the number of spars or the number of wing panels of the aircraft wing. Each spar position can be changed individual.

5.5 Wing Ribs

The wing ribs are the structural elements that are used for the shaping of the wing surface so that the wing is able to generate wing during flight. The wing ribs are normally of an airfoil shape and are placed perpendicularly to an aircraft wing inside the wing panels. The wing ribs are used to provide the aerodynamic (airfoil) shape to a finite 3d wing. Besides, there use for providing the external shape to the wing they are also used to provide structural stiffness and rigidity to the wing structure as well. The wing ribs also have a certain thickness attached to them.

The wing ribs are defined by the following parameters.
1. Rib Position P1
2. Rib Position P2

The Rib position P1 point represents the forward position (along the span) of aircraft wing where the rib is located while; the Rib position P2 represents the rearward aft position (along the span) of the aircraft wing where, the rib is located.

5.6 Surface Model

The surface model of the aircraft wing is specifically designed to be used for the finite element analysis of an aircraft wing. The surface model includes only surfaces which represent the wing shape including spars and ribs. The surface model is chosen for the finite element analysis because it is faster to do a finite element analysis on a 2D surface rather than a 3D one. It is also easier to mesh a 2D surface then a 3D one and also it is computationally beneficial. In a conceptual phase, it is also not necessary to carry out a very precise finite element analysis for the aircraft wing. So, a 2D analysis will give sufficient information to be useful in the conceptual phase while saving time and resources compared to a full 3D FE analysis.

5.7 Solid Model

The solid model contains the solid geometry and thickness for the aircraft wing panels, spars and ribs. The solid model can be used for the detailed design for an aircraft wing which shows the actual thickness and geometries of the structural elements of the wing. As the surface and the solid model of the aircraft wing are integrated together, any change in the parameters of the aircraft wing will be reflected in both the surface and the solid model of the aircraft wing.

5.8 Generic Model

The following picture shows the generic wing model developed using the PowerCopy with VB scripting approach.

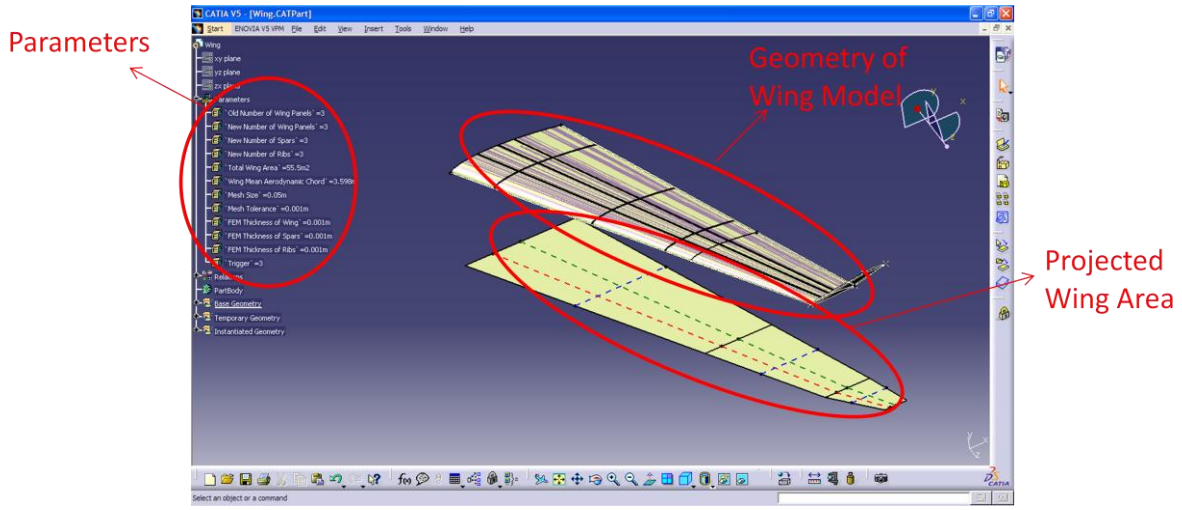


Figure 18: Generic wing model

6 Automated Finite Element Mesh Generation

An automated finite element mesh generation is required for the generic aircraft wing model which would enable fast structural analysis of different wing configurations. The automated mesh generation is done in such a way, that the aircraft wing model can be modified and changed and the mesh will be updated accordingly. The surface model is used for creating the finite element mesh of the entire aircraft wing. This is done so because in the conceptual design phase, a detailed structural analysis is not required. At this stage, an estimate of the structural characteristic of an aircraft wing is beneficial for which a 2D structural analysis is more suitable in terms of computational time and resources. Furthermore, it enables fast analysis of different configurations of the aircraft wing as compared to a full 3D structural analysis. A more detailed structural analysis can off course always be performed in the preliminary or detail design phase.

6.1 Mesh Criteria

The finite element mesh of the aircraft wing should be of good quality. This means that there are not any uneven or large angles in the mesh nodes. Furthermore, it is essential that the all the mesh elements of the wings, spars and ribs are properly connected at the nodes. This means that the growth points of the mesh be properly controlled so that the mesh elements take into account the position and orientation of the ribs, spars and the wing panels.

A global mesh size and tolerance parameter is defined which controls the size and characteristics of the mesh.

6.2 Analysis and Simulation Workbench

The "Advanced Meshing tools" available in the generative shape structural analysis workbench in CATIA V5 software is used for creating the surface mesh of the entire aircraft wing and performing the structural analysis.

In order to perform a structural analysis on a surface, there are four types of elements are available in CATIA V5 for surfaces [8]. They are mentioned in the Table 1 below,

Name of Finite Element	*Type of Finite Element*	*Mesh Connectivity*
Linear Triangle	Surface Element	3 Nodes
Parabolic Triangle	Surface Element	6 Nodes
Linear Quadrangle	Surface Element	4 Nodes
Parabolic Quadrangle	Surface Element	8 Nodes

Table 1: Name and Type of Finite Elements

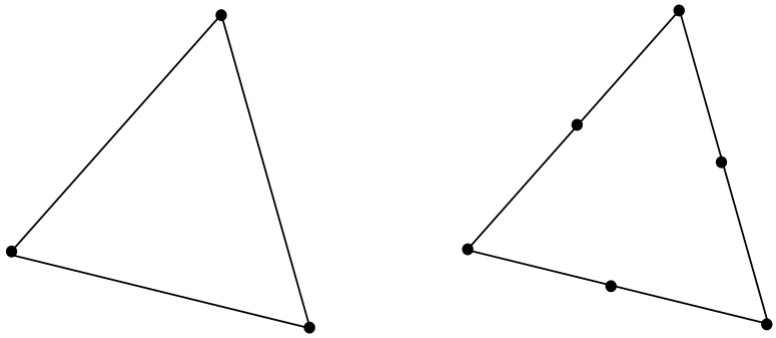


Figure 19: Linear Triangle (3 nodes) and Parabolic Triangle (6 nodes) finite elements

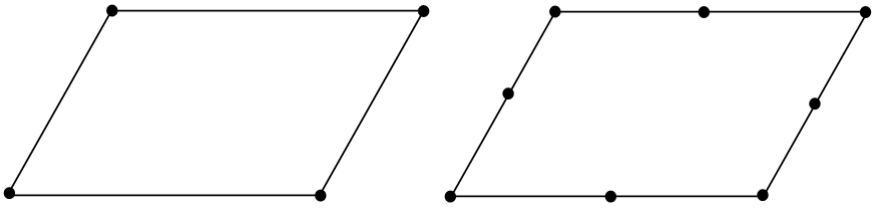


Figure 20: Linear Quadrangle (4 nodes) and Parabolic Quadrangle (8 nodes) finite elements

Each node of the surface elements has 6 degree of freedoms (3 translations and 3 rotations) associated with them.

6.2.1 Methodology for Mesh Generation

The methodology adopted for the mesh generation of the generic wing model is shown in the figure below,

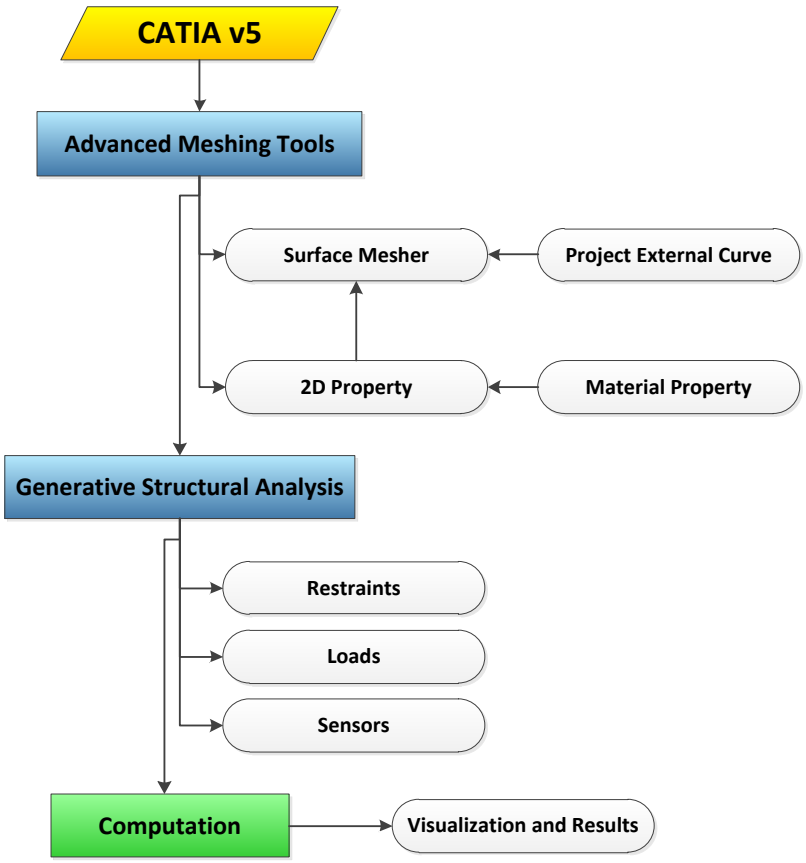


Figure 21: Methodology for mesh generation of generic wing model

The "Advanced Meshing Tools" workbench is used for,

- Linking geometry of generic model to surface mesher
- Define FE mesh elements
- Surface mesh generation

The "Generative Structural Analysis" workbench is used for,

- Defining loads and restraints
- Perform structural analysis
- Visualization of results

6.3 Wing Panels Mesh

A Surface mesh is generated for the wing panels which are linked to a "join wing" surface in the Wing Panel model geometrical set. As, the number or size/shape of the wing panels are changed, the "join wing" surface is updated. As, this surface is connected with the surface mesh for the wing panels, this ensures that when the mesh is updated, it takes into account any and all changes made to wing panel geometry. In order to ensure, that the nodes of the mesh elements between wings and ribs as well as wing and spars are properly connected, an intersection lines are defined between "Wing Join and Ribs Join" and between "Wing Join and Spars Join" surfaces. The growth of the wing panels now take place from these intersection lines, thus ensuring that the nodes between wing panels, ribs and spars are

connected properly to each other. As an example of the wing panel mesh is shown in the figure below,

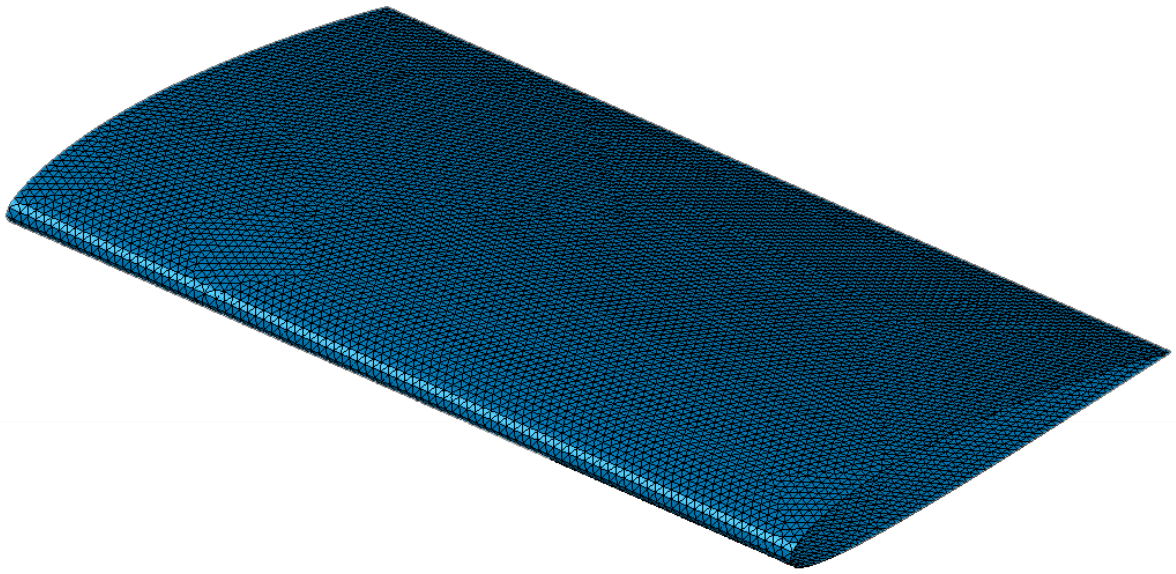


Figure 22: Wing Panel mesh with parabolic triangular finite elements

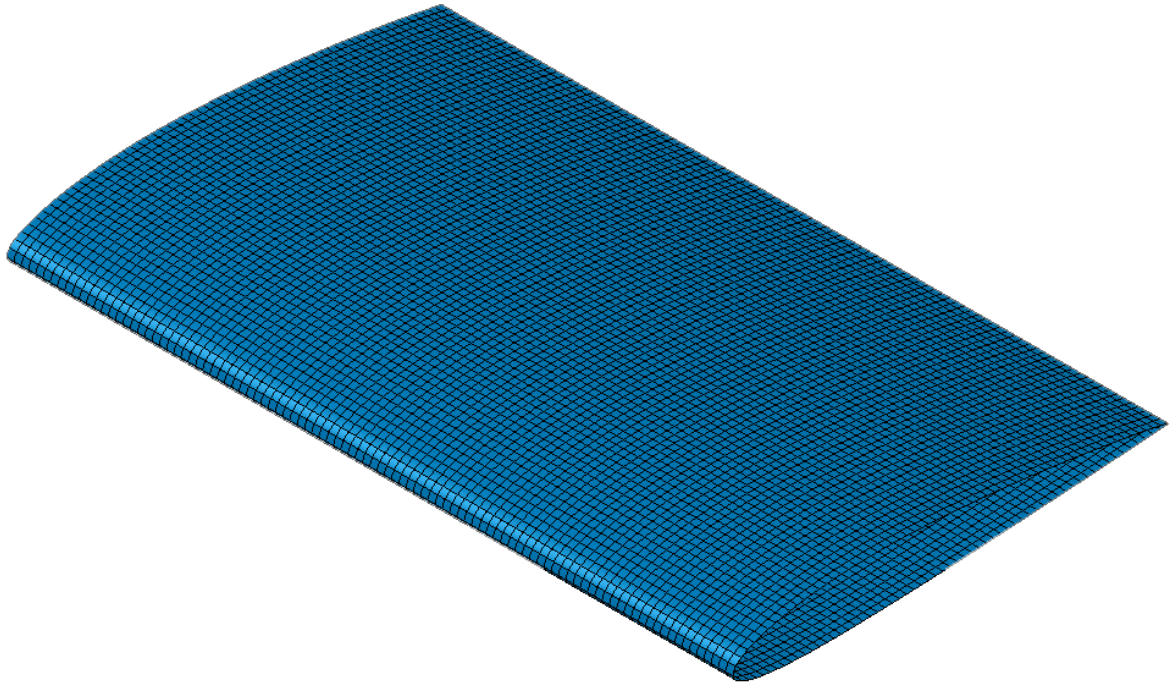


Figure 23: Wing Panel mesh with parabolic quadrangle finite elements

The quality of the mesh can be checked by using a dedicated mesh quality tool available in CATIA "Advanced Structural Analysis" module. The tool shows the quality of mesh in terms of a color palette. A green color for the mesh indicates that a mesh is of good quality while, a red color shows that the mesh is of bad quality. Interim colors between the green and the red shows that the mesh is of intermediate quality. Meshing tools available in CATIA V5 CAD software can be used for altering the mesh nodes if some elements nodes are found to be of

bad quality or at uneven or high angles. As an example, for checking the quality of the mesh, the mesh quality tool is used on the above wing panel as shown in the figure below,

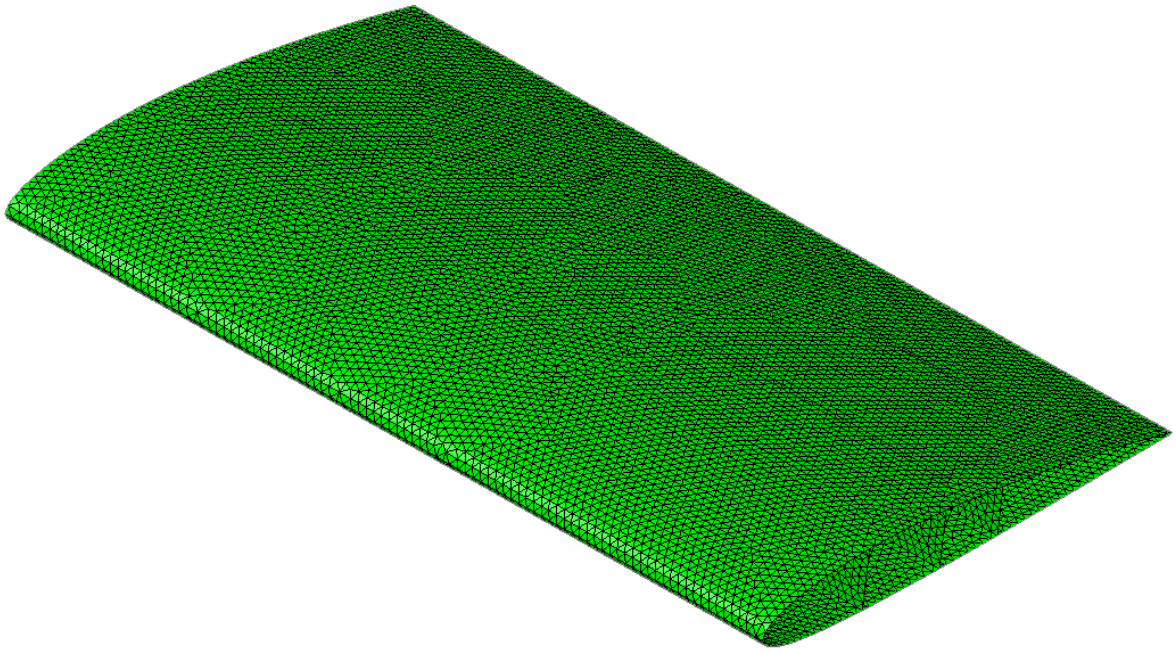


Figure 24: Quality of the mesh on the wing panel (green color shows mesh elements are of good quality)

6.4 Wing Spars Mesh

A surface mesh is generated for the spars of the wing which is linked to a "join spar" surface in the Spars geometrical set in the aircraft model. As the number, position or the thickness of the spars is changed, the "join spar" surface is updated automatically. This ensures that the current geometrical information for the spars is available for the mesh and when the spar mesh is updated, the mesh is generated for the current geometry. Intersection lines between "Spars and Ribs" are also defined. They ensure that the mesh growth for the spars takes place in such a manner that the nodes of the mesh between the Ribs and the spars are properly connected to each other. An intersection line between "Wing Panel and Spars" was already defined in the wing panel mesh so that the mesh elements between wing panels and ribs are already properly connected to each other.

As an example, a surface mesh for the wing spars (which in this case are 4) is shown in the figure below,

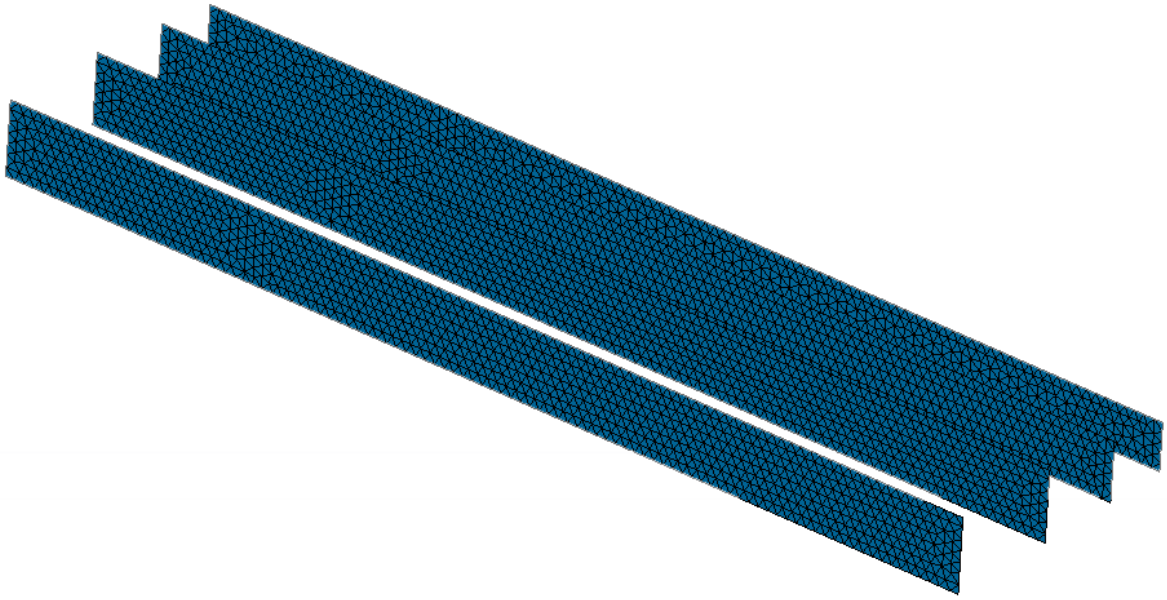


Figure 25: Wing Spars mesh with parabolic triangular finite elements

As an example of the mesh connection between spars and the ribs the following figure is shown which shows that the mesh elements nodes between the wing spars and the wing ribs are properly connected to each other.

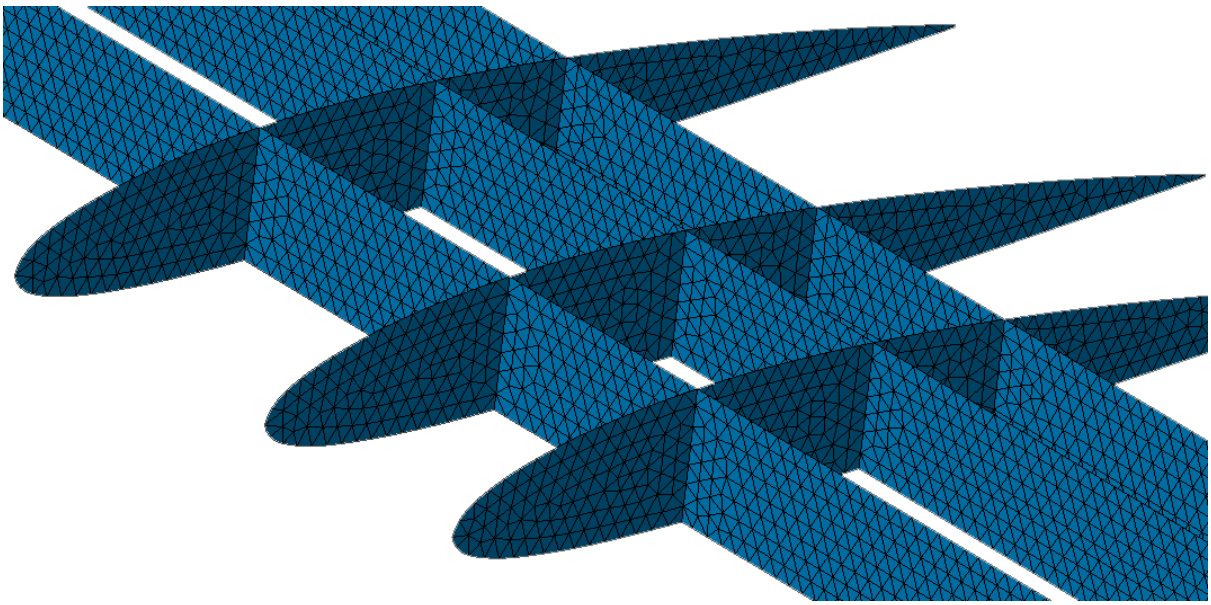


Figure 26: Mesh connectivity between FE mesh of wing spars and FE mesh of wing ribs

The quality of the mesh can be checked by using the dedicated mesh quality tool that is available in CATIA V5 software. Elements in the green color show that the mesh elements are of good quality.

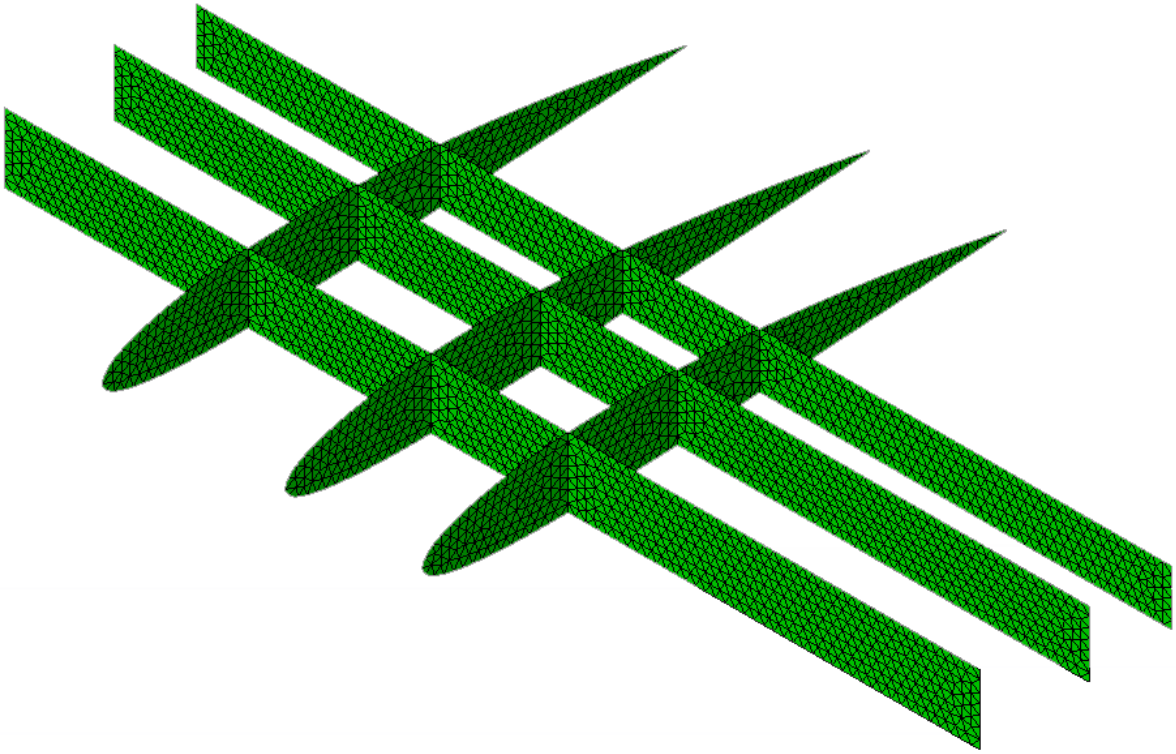


Figure 27: Quality of FE Mesh on wing spars

6.5 Wing Ribs Mesh

A surface mesh is generated for the ribs of the aircraft wing which is linked to a "join ribs" surface in the Ribs geometrical set in the aircraft model. As the number, position or the thickness of the wing ribs is changed, the "join ribs" surface is updated automatically. This ensures that the current geometrical information for the wing ribs is available for the mesh and when the rib mesh is updated, the mesh is generated for the current geometry of the ribs. Intersection lines between "Spars and Ribs" are also defined. They ensure that the mesh growth for the wing ribs takes place in such a manner that the nodes of the mesh between the Ribs and the spars are properly connected to each other. As an example of the surface mesh generated for the wing ribs is shown in the figure below,

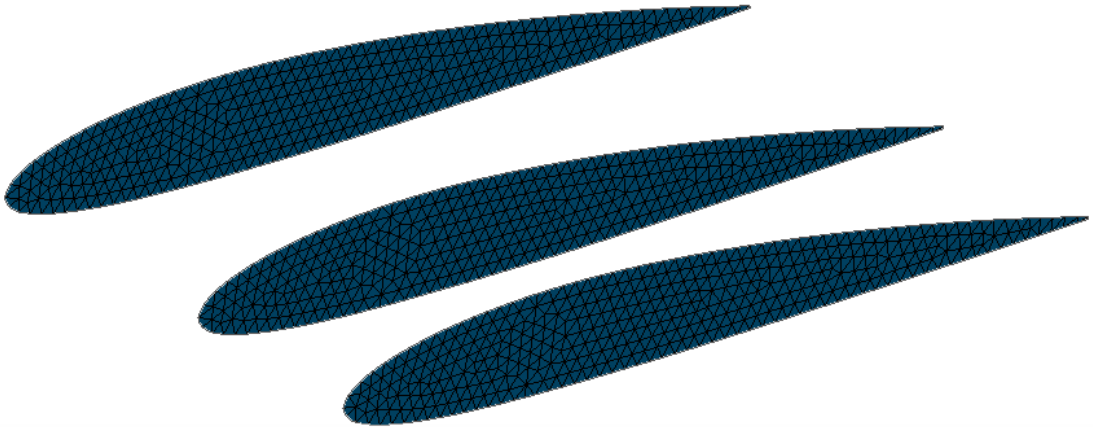


Figure 28: Wing Ribs mesh with parabolic triangular finite elements

7 Results and Discussion

7.1 Comparison between Knowledge Pattern and Power Copy with VB Scripting Approach

A comparison between the knowledge pattern approach and the power copy with VB scripting approach is performed for the generic wing model.

7.1.1 Time for Instantiation and Deletion

The time to instantiation and deletion of the generic wing geometry is measured for the two approaches. In order for the results to be as accurate as possible, the test is performed on the same computer with only CATIA V5 CAD software running in the foreground and all other applications close with minimum number of processes of windows operating system running in the background. The computer that was used to perform the test is,

Computer: Intel Pentium Dual CPU T2390 @ 1.83 GHz with 1.99 GB of RAM
Operating System: Windows XP Professional, Service Pack 3

Two readings of time were taken for each case and the average of these two times is quoted as the official time.

7.1.1.1 Test # 1: Time to Instantiate and delete wing panels

A comparison between the times to instantiate wing panels in both approaches is shown in the figure below,

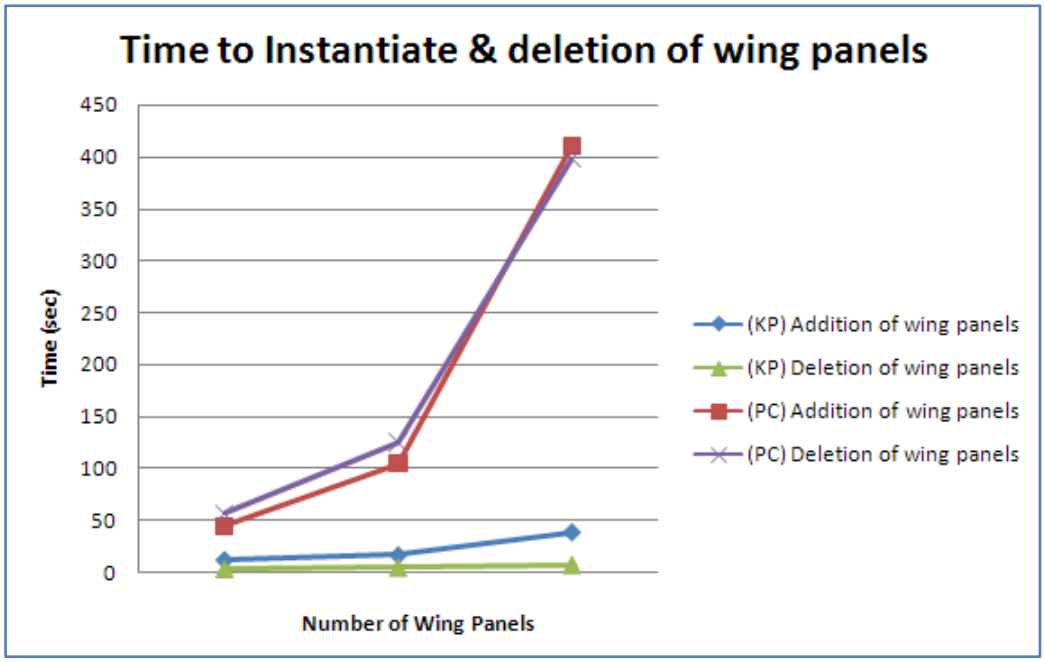


Figure 29: Time to instantiate and deletion of wing panels

7.1.1.2 Test # 2: Time to Instantiate and delete wing spars

A comparison between the times to instantiate wing spars in both approaches is shown in the figure below. In this case, the number of wing panels is set to one for both approaches.

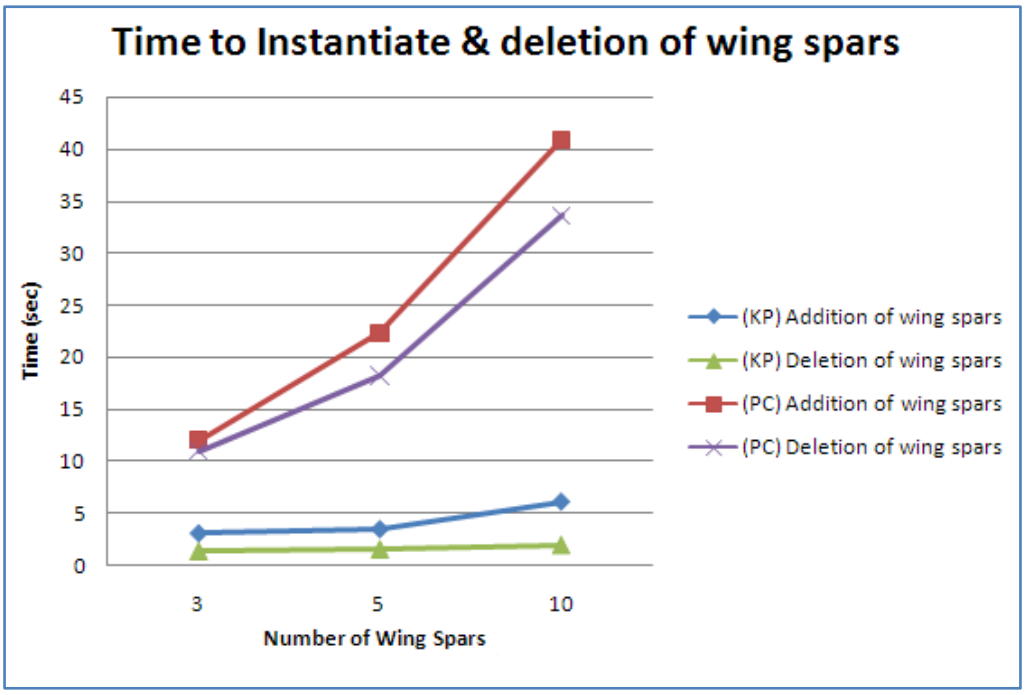


Figure 30: Time to instantiate and deletion of wing spars

7.1.1.3 Test #3: Time to instantiate and delete wing ribs

A comparison between the times to instantiate wing ribs in both approaches is shown in the figure below. In this case, the number of wing panels is set to one for both approaches.

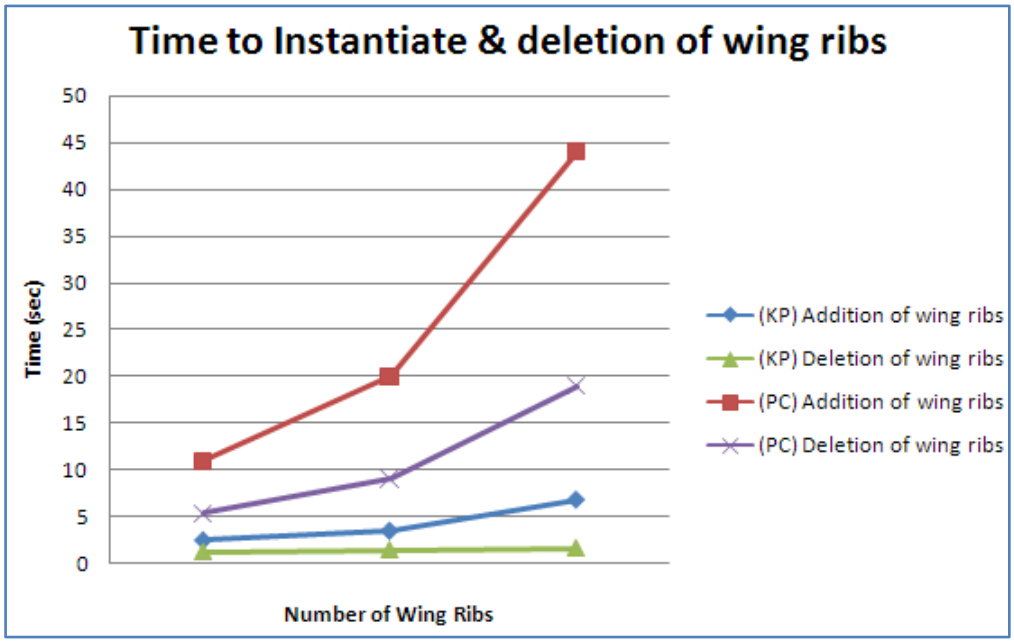


Figure 31: Time to instantiate and deletion of wing ribs

7.1.1.4 Conclusion

It is observed that in both the time to instantiate and the time to deletion for all the six tests on the generic aircraft wing model, the knowledge pattern approach is *significantly faster* than the power copy with visual basic scripting approach. The difference in time is sufficiently clear in both the time to instantiate and time to delete the wing panels, while the difference is

less in the time to instantiate and time to deletion of the wing spars and wing ribs. Since, a lot of geometric data has to be added and deleted from the generic model in powercopy with VB scripting approach, the addition and deletion takes more time than the equivalent knowledge pattern approach.

7.1.2 Difference in programming between the two approaches

Both of these approaches use a different syntax and are based upon different programming languages. The Knowledge Pattern approach is based upon the Engineering Knowledge Language (EKL) while, the PowerCopy with Visual Basic Scripting is based upon the Microsoft Visual Basic language.

7.1.2.1 Amount of Code Lines

The amount of lines of code that needs to be entered between the two approaches is quite different. In order to accomplish a similar task in both the approaches, it is found out that the amount of lines of code that needs to be written in knowledge pattern approach is smaller than the powercopy with VB scripting approach. Furthermore, since, the knowledge pattern has the ability for the management of the geometry and deletion sequence is catered for automatically, the programming lines for deletion code are not necessary. However, in the powercopy with VB scripting approach, the entire deletion sequence has to be written, and thus the amount of code becomes larger.

7.1.2.2 Programming Syntax

As of the year 2011, not much help about the knowledge pattern and its syntax is available. This is because the knowledge pattern feature was introduced a few years back. Because of this, some time needs to be investigated to find out the correct syntax and functionality before the actual code writing for the model takes place. The case is totally different for the PowerCopy with VB scripting approach. Since, it is based on the standard Microsoft Visual Basic language, significant amount of documentation and help is available for the syntax. Coding in Visual Basic environment is simpler, due to the ability to record macros. Furthermore, any IDE (Integrated development environment) for programming in VB can be used to write the code for PowerCopy with VB scripting, not just the one included in CATIA V5.

7.1.2.3 Error Checking and Debugging of Code

There is no way for error checking and debugging of the code in knowledge pattern as of the year 2011 and version R18 of CATIA V5. This causes a lot of problems in finding an error in the code, and debugging of the code. The entire knowledge pattern code runs and the code cannot be run one step at a time. So, the exact understanding of the working of the code is essential for the debugging of the code written for the knowledge pattern. The case is quite different in PowerCopy with VB scripting approach, where, a standard IDE for VB programming can be used. Here they are available many ways for syntax error checking and debugging of code. “Watches” can be set up to record the values of parameters for example or the entire code can be run one step at a time. In this way, the entire process to error checking and debugging of the code can be made logical and methodological. This also offers

the possibility for making the code more robust to handle a wide variety of scenarios by debugging and error checking.

7.1.2.4 Accessibility of Features and Tools in CATIA

It is noted that not all the features and tools that are available in CATIA V5 can be accessed by the knowledge pattern. Only the features that are available in the knowledge pattern syntax can be used to access the tools. This means that the knowledge pattern is somewhat limited in the number of features in CATIA that it can access, and thus the code that is written in knowledge pattern can only take into account these features. On the other hand, in the powercopy with VBA scripting approach, a very large number of features and tools can be accessed. This offers possibility to use a range of tools and features in programming the model that can't be accessed by the knowledge pattern.

8 Conclusions

The following conclusions are made regarding the master thesis work,

1. Both the knowledge pattern approach and the powercopy with VBA scripting approach can be used successfully in creating a generic wing model as shown in this thesis work.

2. If the time for modification, update and deletion of the geometry of the generic model is critical, then the knowledge pattern model is superior to the powercopy with VBA scripting model. The knowledge pattern based generic model was faster in all the tests compared to the powercopy with VBA scripting based generic model.

3. Not all features and tools in CATIA can be accessed by the knowledge pattern, whereas, a large number of features and tools can be accessed using VBA.

4. Not all geometrical entities of the generic model are accessible in the knowledge pattern based model while, all the geometrical entities of the generic model are accessible in the powercopy with VBA scripting based generic model.

5. The amount of code required to be written for the knowledge pattern based generic model is significantly smaller than the powercopy with VBA scripting based model.

6. Debugging and error checking is much easier in powercopy with VBA scripting than in the knowledge pattern due to the availability of the debugging tools in VBA environment.

7. The generic aircraft wing model that is developed can be very effective and useful in the aircraft conceptual design process and can result in cost savings associated with the design process of aircrafts.

8. Automatic finite element mesh generation and modification is possible as shown in this thesis work.

9 Recommendations

9.1 Future Work

As future work to this master thesis, the following recommendations are presented.

1. The methodology presented in this thesis work for the creation of the generic model for the aircraft wing can also be used to create a generic model for the aircraft fuselage.
2. Wing Flaps, ailerons and slats can be incorporated in the generic model for the wing.
3. Wing stringers can be incorporated into the generic model, in addition to the wing panel skin, wing spars and wing ribs already included in the generic model.
4. The wing spars can be designed in such a way to have different sections e.g. an I-section, a T-section etc. Wing Ribs can also be designed to have holes and cutouts in them.

10 References

[1] Ledermann, C., Hanske, C., Wenzel, J., Ermanni, P., Kelm, R., ”Associative parametric CAE methods in the aircraft pre-design”, Journal of Aerospace Science and Technology, Vol. 9, Issue 7, October 2005, pp. 641-651, Elsevier

[2] Amadori, K., “On Aircraft Conceptual Design, A framework for knowledge based engineering and design optimization”, Thesis No. 1366, Linköping University, Linköping, Sweden, 2008

[3] UIUC Airfoil Coordinates Database, http://www.ae.illinois.edu/m-selig/ads/coord_database.html

[4] Wing Configuration, http://en.wikipedia.org/wiki/Wing_configuration

[5] Wing Spars, http://en.wikipedia.org/wiki/Spar_(aviation)

[6] Wing Ribs, http://en.wikipedia.org/wiki/Rib_(aircraft)

[7] Knowledge Pattern or VBA, http://www.gtwiki.org/mwiki/index.php?title=Knowledge_Pattern_or_VBA%3F

[8] CATIA V5 Release 20 Help, Finite Element Reference, http://catiadoc.free.fr/online/CATIAfr_C2/femugCATIAfrs.htm

[9] Raymer, Daniel P. Aircraft Design: A Conceptual Approach, 2nd Edition. American Institute of Aeronautics and Astronautics, Inc. 1989.

[10] Sohaib, M., Razzaq, H., “Conceptual Design of a Business Jet Aircraft”, TMAL02 Project Report, Linköping University, Sweden, 2010

11 Appendix A

11.1 Aircraft Wing Configuration

11.1.1 Variation of Wing Planform along the Wing Span

When the wing chord is varied along the wing span different types of wing planform can be achieved some of which are presented in the figure below,

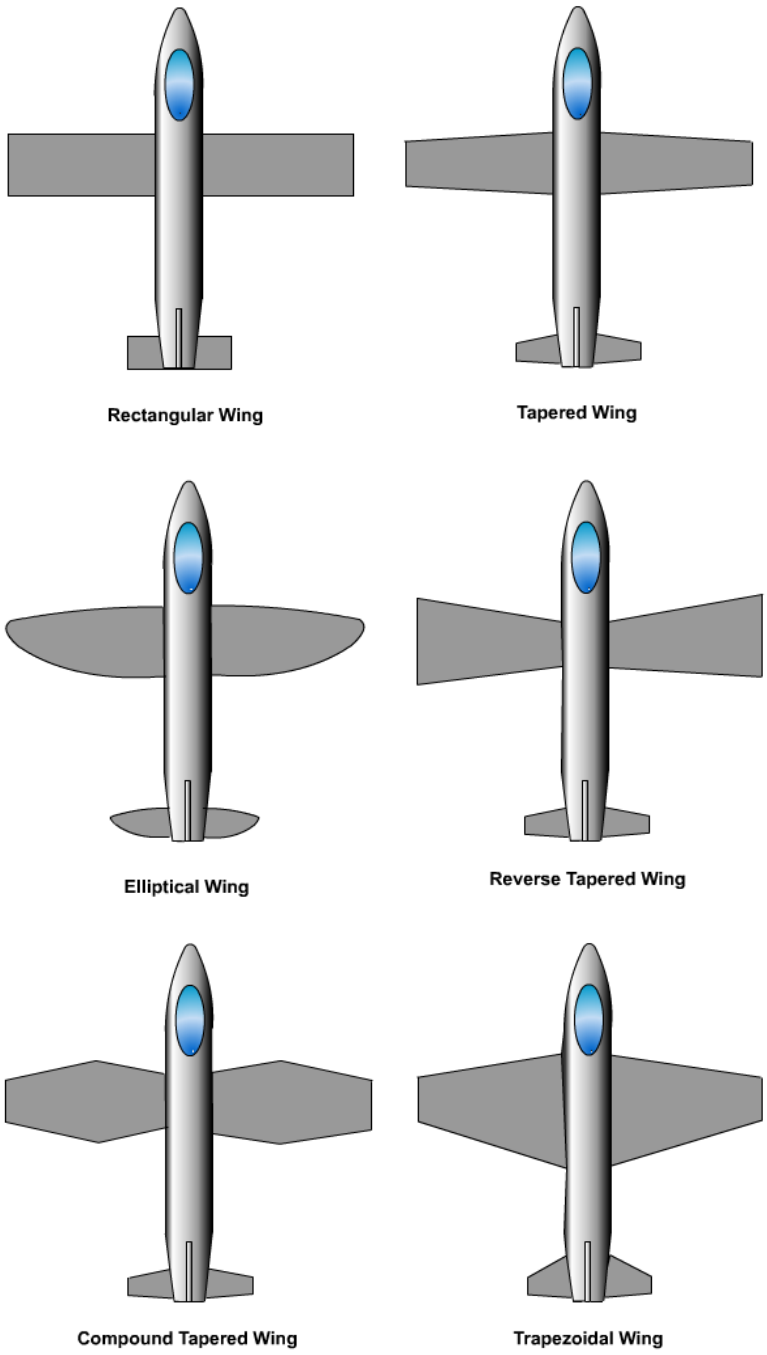


Figure 32: Variation of Wing Planform along the wing span

11.1.1.1 Rectangular Wing (Constant Chord)

The rectangular wing is the type of wing planform where, the root chord and the tip chord of the wing are of equal length. The rectangular or constant chord wing is normally found on low-subsonic aircrafts. These aircrafts fly in a low-mach regime typically between Mach 0.2 to Mach 0.4. A straight rectangular constant chord wing is structurally easier to built and cost less than other types of wing planform.

11.1.1.2 Tapered Wing

A tapered wing is the type of wing planform, where, the root and the tip chord of the wing are not of equal length. A tapered wing is chosen for aerodynamic reasons. A taper introduced to the wing will change the lift distribution. The ideal form of span-wise lift distribution is elliptical. A tapered wing design is chosen to achieve a span-wise lift distribution that is closer to an elliptical shape. Furthermore, the taper wing design is structurally easier to manufacture than an elliptical wing.

11.1.1.3 Elliptical Wing

An elliptical wing is the type of wing planform, where, the wing planform is in the shape of an ellipse and the edges of the wing turn inward to form a rounded tip. The ideal form of span-wise lift distribution is elliptical, which is one of the prime reasons for choosing an elliptical wing planform. However, the elliptical wing planform is difficult to manufacture than a rectangular or a tapered wing.

11.1.1.4 Reverse Tapered Wing

A reverse tapered wing is a type of wing planform, where, the root chord of the wing is smaller than the tip chord of the wing. The reverse tapered wing is structurally inefficient as larger weight is concentrated on the wing tip rather than the wing root, leading to a higher weight for the wing.

11.1.1.5 Compound Tapered Wing

A compound tapered wing is a type of wing planform, where, the taper ratio changes along the span. A compound tapered wing can be thought of as consisting of two trapezoidal wing panels. In the first wing panel, the wing root is smaller than the wing tip, while, in the other wing panel, the wing tip is smaller than the wing root.

11.1.1.6 Trapezoidal Wing

A trapezoidal wing is a type of wing planform, where, the root and the tip chord of the wing are not equal. Furthermore, the leading edge of the wing sweeps backward, while, the trailing edge of the wing sweeps forward.

11.1.1.7 Delta Wing

A delta wing has a triangular planform with a swept leading edge. A delta wing is typically found on supersonic jet aircrafts, where, it is aerodynamically efficient than a straight wing shape, offers advantages of a swept wing design and structurally sound as well. A delta wing shape is less aerodynamically efficient at low-subsonic speeds. Some of the different types of delta wing planform are shown in the figure below,

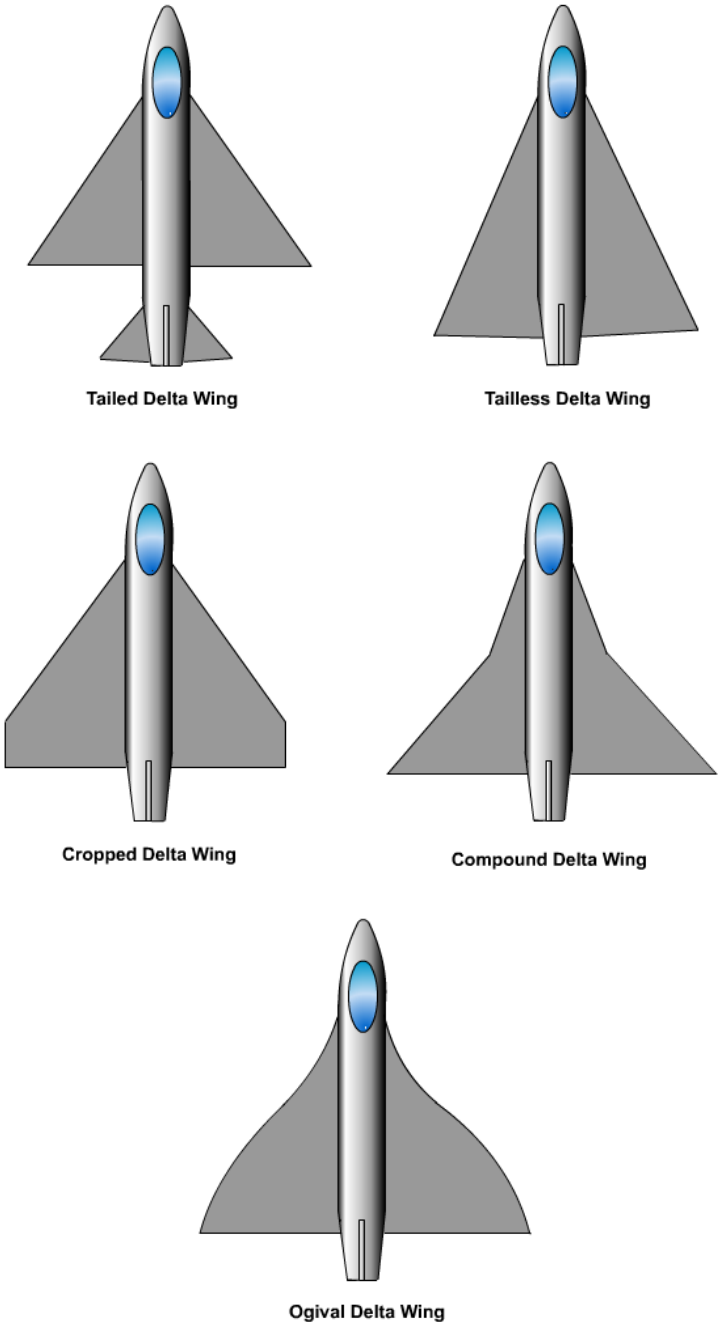


Figure 33: Different types of Delta Wing Planform

11.1.1.7.1 Tailed and Tailless Delta Wing

In a tailed and a tailless delta wing, the taper ratio of the wing is zero. The tailed delta wing is characterized by a standard conventional horizontal tail, while, in the tailless delta wing design there is no horizontal tail.

11.1.1.7.2 Cropped Delta Wing

A cropped delta wing is a type of delta wing planform where, the tip of the delta wing is cut off.

11.1.1.7.3 Compound Delta Wing

A compound delta wing is a type of delta wing planform, which has two sweep angles. In a compound delta wing, the inner sweep angle is larger or steeper, while, in the outer sweep angle is smaller.

11.1.1.7.4 Ogival Delta Wing

An Ogival delta wing is a type of delta wing planform, where the leading edge of the wing blends smoothly from the root to the tip, typically resembling a "wine-glass" shape. This type of delta wing planform is seen on Concorde SST aircraft.

Some other types of wing planform are,

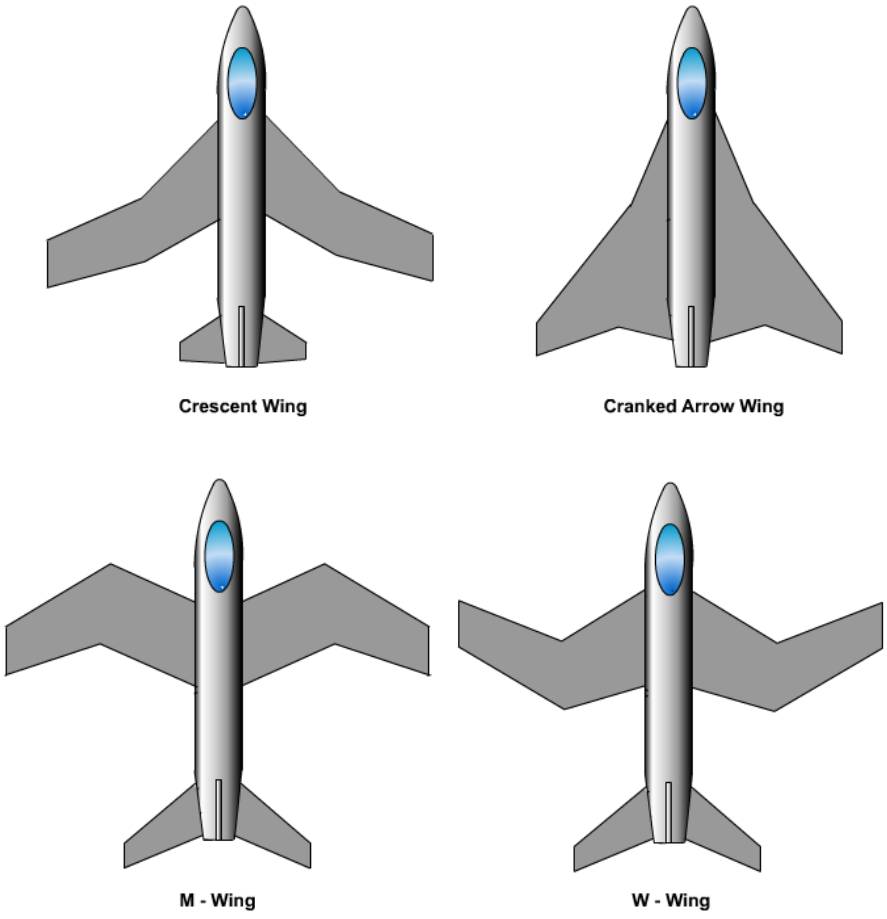


Figure 34: Different Types of Wing Planform

11.1.1.8 Crescent Wing

The crescent wing is a type of aircraft wing planform where, the outer section of the wing has a smaller sweep angle than the inner section of the wing.

11.1.1.9 Cranked Arrow Wing

The cranked arrow wing is a type of aircraft wing planform which has two distinct sweep angles at the leading edge of the wing. This type of planform is similar to the compound delta planform, however, in the cranked arrow wing, the trailing edge of the wing is kinked inwards. This type of configuration can be seen on the General Dynamics F-16XL aircraft.

11.1.1.10 M-Wing

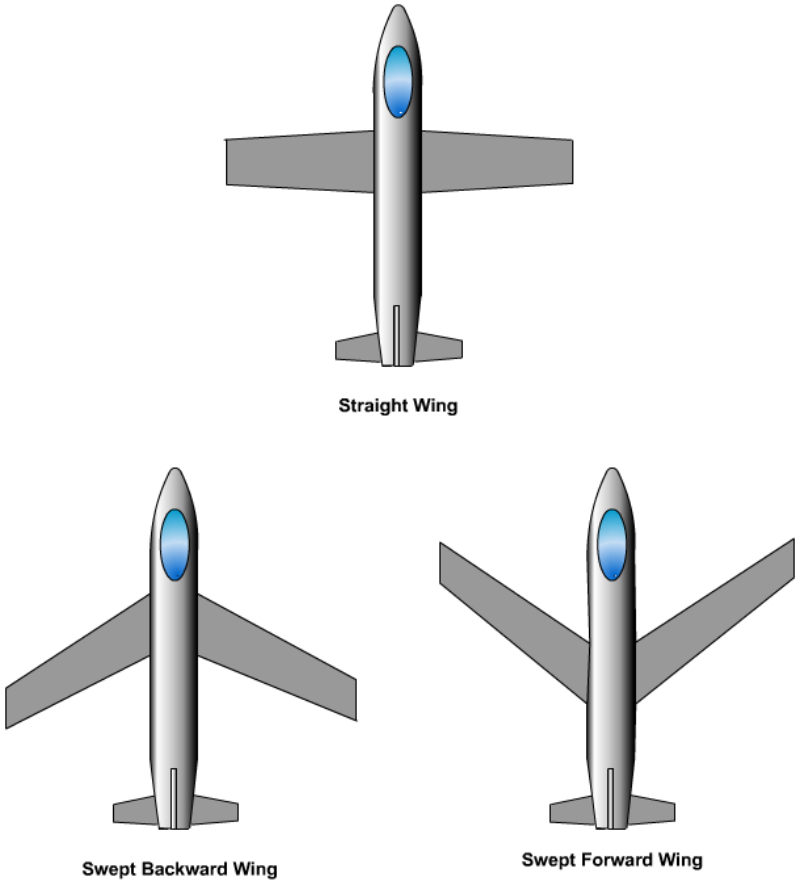
The M-wing is a type of wing planform which is in the shape of the alphabet (M). In this type of wing planform, the inner section of the wing sweeps forward while, the outer section of the wing sweeps backward.

11.1.1.11 W-Wing

The W-wing is a type of wing planform which is in the shape of the alphabet (W). In this type of wing planform, the inner section of the wing sweeps backward while, the outer section of the wing sweeps forward.

11.1.2 Wing Planform based on Wing Sweep

When the sweep of the wing is changed, different types of wing planform can be achieved some of which are shown in the figure below,



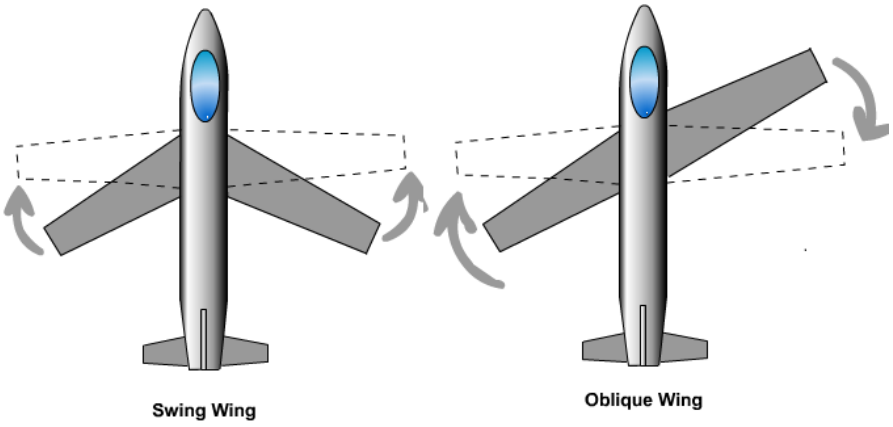


Figure 35: Wing Planform based on Wing Sweep

11.1.2.1 Straight Wing

A straight wing is a type of aircraft wing planform with little or no sweep angle associated with it. A straight wing planform can have a rectangular shape or a tapered shape. The straight wing planform is structurally easier to build then other types of wing planform which have sweeps associated with them.

11.1.2.2 Swept Backward Wing

A swept backward wing is a type of wing planform where, the wing has a backward sweep angle associated with it. The wing angles backward from the root to the tip of the wing. This type of planform is typically used for aircrafts that will fly in the high subsonic or transonic flight regime. The sweep backward introduced to the wing is used to overcome the adverse effects relating to high-subsonic or transonic flight.

11.1.2.3 Swept Forward Wing

A swept forward wing is a type of wing planform where, the wing has a forward sweep angle associated with it. The wing angles forward from the root to the tip of the wing. A forward swept wing produces greater bending moment at the root of the wing compared with a swept backward wing and thus requires higher amount of stiffness to be added. A swept forward wing can be used for high-subsonic flight.

11.1.2.4 Swing-Wing (Variable Sweep) Wing

A swing-wing design is a type of wing planform, where, the sweep of the wing can be varied so this type of wing planform is also called as a ‘variable sweep’ wing. A variable sweep wing is beneficial when the aircraft fly for long periods of time in both low subsonic and high subsonic regimes. When the aircraft is flying at a low-subsonic speed, the swing-wing behaves much as a straight tapered wing design while, flying at a higher subsonic or supersonic speed the sweep of the wing can be changed to offer better aerodynamic performance at higher speeds. This type of wing planform can be seen on the F-14 aircraft.

11.1.2.5 Oblique Wing

An oblique wing design is a type of wing planform, where, the whole wing can be rotated along the central or mid-point of the wing. The wing is pivoted in the center (in the fuselage)

and can be rotated causing one side to have a forward sweep angle while the other side to have a backward sweep angle.

11.1.3 Wing Planform based on Aspect Ratio

When the aspect ratio of the wing is changed, different types of wing planform can be achieved some of which are shown in the figure below,

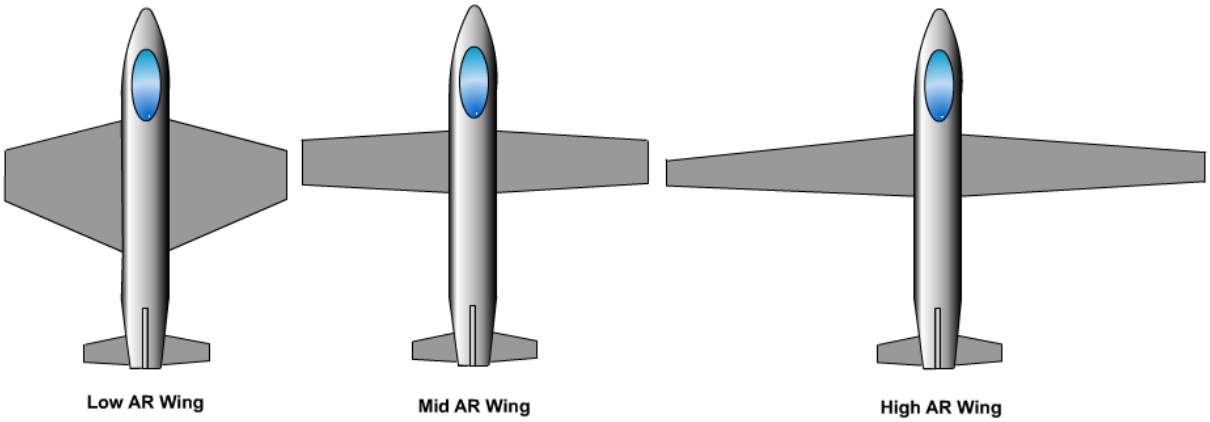


Figure 36: Wing Planform based on Aspect Ratio

11.1.3.1 Low Aspect Ratio Wing

Aspect Ratio is the ratio between the wing span to the wing area. A low aspect ratio wing is a type of wing which has a smaller span. This type of wing planform is used mostly on fighter aircrafts. This type of wing planform has a higher induced drag associated with it.

11.1.3.2 Moderate Aspect Ratio Wing

The moderate aspect ratio wing is a type of wing which has a bigger span than a low-aspect ratio wing, typically, seen on modern commercial aircrafts.

11.1.3.3 High Aspect Ratio Wing

In the high aspect ratio wing, the wing span of the wing is quite large, leading to an efficient aerodynamic configuration which has a lower induced drag. This type of wing planform is typically used on high-altitude subsonic aircrafts.

11.1.3.4 Low-Wing

The low wing configuration is the type of aircraft wing configuration where, the wing is attached to the bottom of the fuselage. The low-wing configuration can be used for mounting and storing the landing gears. The wing fuselage structural design in a low-wing configuration is less complex as compared to a mid or a high wing configuration [10].

11.1.3.5 Mid-Wing

The mid wing configuration is the type of aircraft wing configuration where, the wing is attached to the middle of the fuselage. The mid wing configuration is aerodynamically efficient since, the wing is placed ideally between the upper and lower sections of the fuselage. One of the disadvantages of this type of configuration is the structural complexity that is introduced at the wing-fuselage intersection. Furthermore, a mid-wing configuration

doesn't provide provision for the mounting of the landing gear on the wing, thus, the landing gears have to be mounted on the fuselage which uses valuable space.

11.1.3.6 High-Wing

The high wing configuration is the type of aircraft wing configuration where, the wing is attached to the top of the fuselage. High wing configuration has the advantage that they have very good lift distribution across the span and the negative impact on the lift due to the fuselage is removed. This is one of the reasons why, this type of configuration is seen on heavy transport or cargo aircrafts. Furthermore, wing pylon engines can be easily mounted on the wing with good ground clearance availability. However, the disadvantage of this type of configuration is the high bending loads experienced at the wing fuselage intersection, thus a very strong wing-fuselage structure is needed in order to support the wing.

11.1.3.7 Dihedral Wing

A dihedral wing is a type of aircraft wing configuration where, the root of the wing is lower than the wing tip, when the aircraft is seen from the front along the horizontal axis. Dihedral contributes positively to the stability of the aircraft along the roll-axis (spiral mode). As the aircraft bank, the dihedral wing tends to roll the aircraft and helps to restore indirectly the wings level [9].

11.1.3.8 Anhedral Wing

An anhedral wing is a type of aircraft wing configuration where, the root of the wing is higher than the wing tip, when the aircraft is seen from the front along the horizontal axis. Anhedral contributes negatively to the stability of the aircraft. An anhedral wing is used for configurations to augment the loss of maneuverability as a result of some feature introduced to the aircraft which adds too much stability. An anhedral wing is thus used to increase the maneuverability of the aircraft.

11.1.3.9 Wing with Wing Tips

An aircraft wing can have a wide variety of wing tips. The wing tips are present at the tip of the wing and help to reduce the induced drag. At the wing tips, when the airflow passes from below to the top of the wing, this results in a creation of vortices, which are known as wing tip vortices. The wing tips are designed to reduce these wing tip vortices and contribute positively to the reduction of drag on an aircraft.

11.1.3.10 Gull Wing

A gull wing is a type of aircraft wing configuration which has a sharp dihedral on the inner section of the wing. The outer section of the wing can have a little or no dihedral angle or otherwise an anhedral angle. This type of configuration is typically used to raise the wing above the ground so as to provide better ground clearance.

11.1.3.11 Inverted Gull Wing

An inverted gull wing is the opposite of the gull wing and is the type of aircraft wing configuration which has a sharp anhedral angle on the inner section of the wing while, the outer section of the wing has a dihedral angle. The inverted gull wing can be used on those types of aircrafts, where, the length of the wing mounted landing gears has to be reduced.

11.1.3.12 Upward Cranked Wing

An upward cranked wing is the type of aircraft wing configuration where, the wing tip has greater dihedral angle than the main or inner section of the wing.

11.1.3.13 Downward Cranked Wing

A downward cranked wing is the type of aircraft wing configuration where, the wing tip has an anhedral angle associated with it.

There are also several other types of aircraft wing configurations that can be achieved by changes in the shape, geometry and positioning of the wing along the fuselage which are not mentioned here.

In the next section, some of the different parameters and aspects which influence the design of the aircraft wing are discussed.

11.2 Aspect Ratio

For an aircraft, the aspect ratio is defined as the wing span squared divided by the wing area. A higher aspect ratio wing means that the wing span is large while, a lower aspect ratio means that the wing span is small. Higher aspect ratio wings are most seen on gliders or high-endurance or long-range aircrafts, like the solar impulse or the U-2, while, lower aspect ratio wings are mostly seen on fighter aircrafts, where, the wing span is considerably smaller compared to the jet aircrafts. A higher aspect ratio wing has a lower drag than a smaller aspect ratio wing, which is one of the reasons why, these types of wings are often used on gliders. Modern commercial jet airliners normally have moderate aspect ratio's typically between 7 and 9.

11.3 Wing Sweep

The wing sweep defines how the wing is swept. The wing sweep is normally given along the leading edge or along the quarter-chord line. It is used to counteract the adverse effects of the high subsonic and transonic flight regime. As a straight wing is taken up to higher mach numbers, it experiences a considerable increase in drag. For a straight wing, the drag divergence which is phenomenon where, the drag increases significantly occurs at a lower mach number. This means that the straight wing experiences drag divergence earlier and thus, its aerodynamic efficiency decreases at higher mach numbers. A sweep added to the wing along the leading edge or along the quarter-chord line can be used to overcome the adverse effects of the transonic flow. For a swept wing, the drag divergence Mach number is higher as compared to a straight wing. For modern commercial jet airlines, which fly in the transonic flow regime typically at mach numbers of around 0.8, wing sweep is essential otherwise, there would be a huge penalty in terms of drag increase. A wing sweep introduced in a wing improves the stability and has a natural dihedral effect. However, sweeping the aircraft wing adds weight to the structure. Most high speed fighter jets and supersonic aircrafts have highly swept or delta shaped wings to provide better aerodynamic performance at high subsonic and supersonic mach numbers.

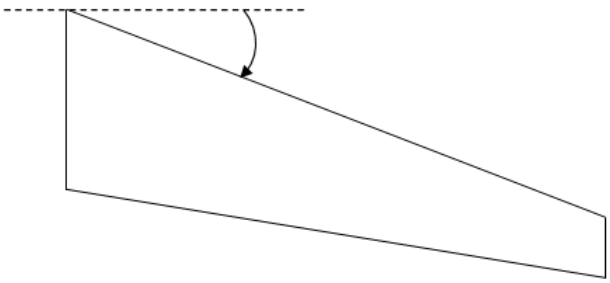


Figure 37: aircraft wing with a sweep introduced at the leading edge

11.4 Taper Ratio

For an aircraft wing, the taper ratio is defined as the ratio between the tip chord of the wing and the root chord of the wing. The symbol used for the taper ratio is the Greek letter lambda (λ). The taper ratio is defined as,

$$\lambda = \frac{C_{tip}}{C_{root}} = \frac{Tip\ Chord}{Root\ Chord}$$

The taper ratio has values between 0 and 1. A taper ratio of 1 means that both the tip chord and the root chord are equal thus, leading to a rectangular planform for the wing. A taper ratio of zero means that the tip chord is zero which leads to a delta wing planform, while, any other value of the taper ratio between 0 and 1 leads to a tapered wing planform. These three planform shapes are shown in the figure below,

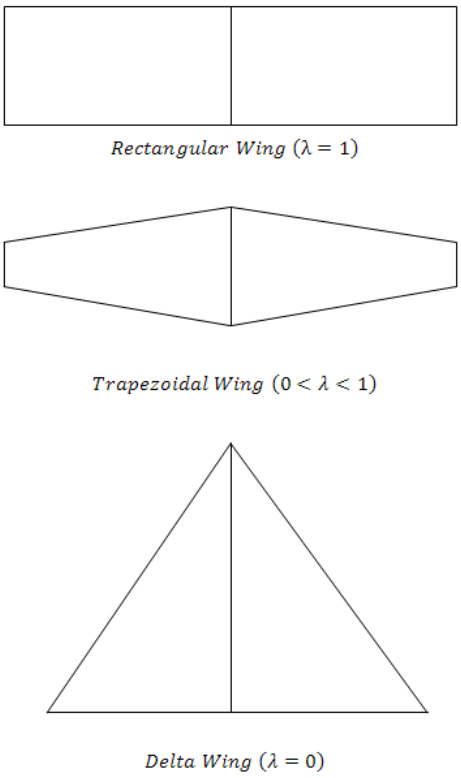


Figure 38: Types of Wing Planform based on Taper Ratio

From the Prandtl wing theory, the minimum drag due to the lift or the induced drag occurs when the wing has an elliptical lift distribution. This means that the wing planform must be in form of an ellipse in order to have the most efficient span-wise lift distribution. However, a wing planform in the shape of an ellipse is structurally difficult to manufacture. Therefore, a taper wing configuration is sometimes chosen for the aircraft wing as the span-wise lift distribution is closer to that of an elliptical lift distribution.

11.5 Wing Twist

If the angle of incidence of the wing tip is different from the angle of incidence of the wing root then the aircraft wing is said to have a twist and vice versa. If the tip of the wing has a higher incidence angle than the wing root, then the wing is said to have a positive twist angle or wash-in. If the tip of the wing has a lower incidence angle than the wing root, the wing is said to have a negative twist angle or wash-out. If different airfoil sections are used along the span of the wing from the wing root to the wing tip, then the wing is said to have an aerodynamic twist. If the tip and the root airfoil sections are the same, but at different incidence angles, then it is known as geometric twist.

In aircrafts, wing twist is added to the tip of the wing. This acts as a safety feature in case of stall. For example, if the wing root stalls at certain angle of attack, then due to the twist of the wing, the angle of attack seen by the tip of the wing will be different from the wing root. Normally, in this scenario, the angle of attack seen by the tip of the wing is smaller than the stall angle, which means, that the ailerons can be used to maneuver the aircraft out of stall. If the tip of the wing stalls, which is normally the case in some forward swept wings, then the ailerons become ineffective and are unable to produce a control moment. In this case, there is a big danger that the entire aircraft wing would stalls and it is very difficult to recover the wing from this stall.

One of the disadvantages of having the wing twist added to the wing is that the lift is reduced, since; the twist added to the tip of the wing is negative, which means that as the angle of attack is lower, which causes the lift coefficient to be lower, which in turns generates less lift.

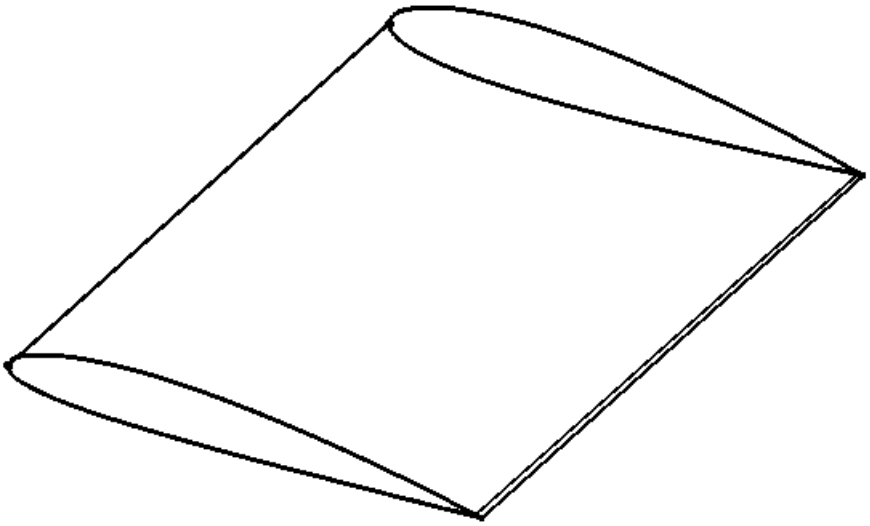


Figure 39: Aerodynamic Twist Different Root and Tip Airfoil Sections

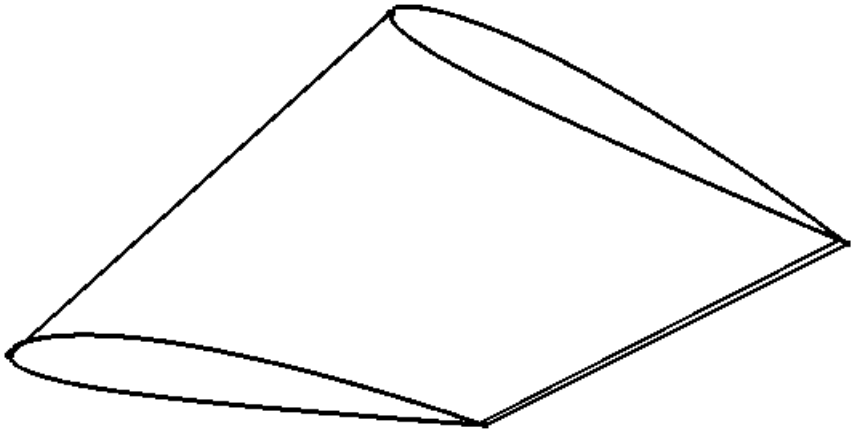


Figure 40: Geometric Twist (Same Root and Tip Airfoil Sections, Different Twist Angles)

11.6 Wing Incidence

The wing incidence is defined as the angle of the wing that is formed between the fuselage and the pitch angle of the aircraft. This value of the wing incidence should be chosen that gives the best Lift to Drag Ratio (L/D) and minimizes the drag during the cruise phase of flight.

11.7 Dihedral

Dihedral is the angle that the wing of the aircraft makes with the horizontal axis when seeing from the front. If the wing tip is higher than the wing root, then the angle is considered positive and is called a dihedral. When the wing tip is lower than the wing root, then the angle is considered negative and is called an anhedral. The dihedral angle helps in the lateral stability of an aircraft. The dihedral angle tends to roll the aircraft as the aircraft banks and tries to return the aircraft to the original trimmed flight condition where the wings are leveled. The dihedral does this by providing a rolling moment as the aircraft banks. An aircraft wing is subjected to a wide variety of aerodynamic, gust and wind loads during the flight. So, some

dihedral is normally added to the wings of the aircraft in order to provide lateral stability. A negative dihedral or an anhedral has the opposite effect. It has lateral destabilizing effect on the wing. Since, more lateral stability means less rolling controllability, the aircraft designer has to decide between lateral stability, controllability and operational performance in order to determine the amount of dihedral or anhedral given to the aircraft. A positive effect of dihedral is that by adding a dihedral to the wing, the ground clearance is improved. Since, the wing tip is higher than the wing root, there is less chances that the wing touches or skids the ground during takeoff and landing. The opposite is true for anhedral angle. As, the wing tips are lower than the wing root, there are a greater chances that the wing touches the ground.

Figure 41: Dihedral and Anhedral Wing on Aircrafts

11.8 Aerodynamic Center

The aerodynamic center is the point where, the pitching moment doesn't vary with the angle of attack. Normally, for airfoils in incompressible or low-subsonic flow, the point is at 25% chord line, while, for a wing this point lies on the quarter-chord line running from wing root to the wing tip. For higher Mach numbers and in supersonic flow, the aerodynamic center moves aft typically for airfoils at about 50% of the chord line while for the wing the aerodynamic center lies along the 50% chord line running from root to tip.

12 Appendix B

12.1 Generic Aircraft Wing Model Parameters

12.1.1 Root Airfoil

The root airfoil defines the shape of the root of the wing panel. The Root airfoil is a drop-down string parameter which can be selected from a list. As, an example, the root airfoil could be "NACA 2412". When any airfoil is selected from the list, the geometry changes automatically to reflect this change for the new airfoil.

12.1.2 Tip Airfoil

The tip airfoil defines the shape of the tip of the wing panel. The tip airfoil is a drop-down string parameter which can select from a list. As an example, if the tip airfoil selected from the list is the "NACA 2415" airfoil, then the geometry of the tip of the airfoil changes automatically to reflect this change. Furthermore, since the tip and the root chord of each successive wing panel are attached together, so when the tip airfoil of for example, first wing panel is changed, the root airfoil of the second wing panel changes automatically to match the tip airfoil of the first wing panel. This ensures that all the wing panels are geometrically connected with each other.

12.1.3 Wing Panel Span

The wing panel span parameter defines the span of the wing panel from the root to the tip. Changing the wing panel span will either increase or decrease the span of the wing. Furthermore, the projected area of the wing panel are also updated continuously, if any or all changes are done to the wing panel span, root chord or tip chord etc.

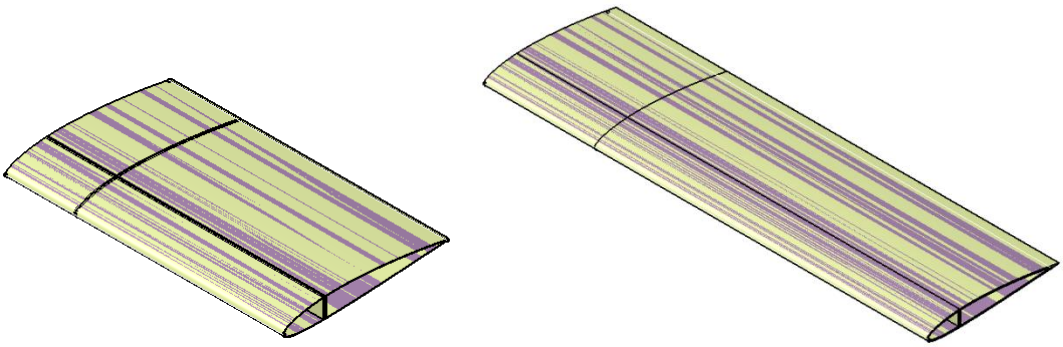


Figure 42: Increasing wing panel span from 5 to 10 meters

12.1.4 Root Airfoil Chord

The root airfoil chord parameter changes the chord length of the root of the wing panel. As, the root airfoil chord is changed, the airfoil of the root of the trapezoidal wing panel is sized accordingly.

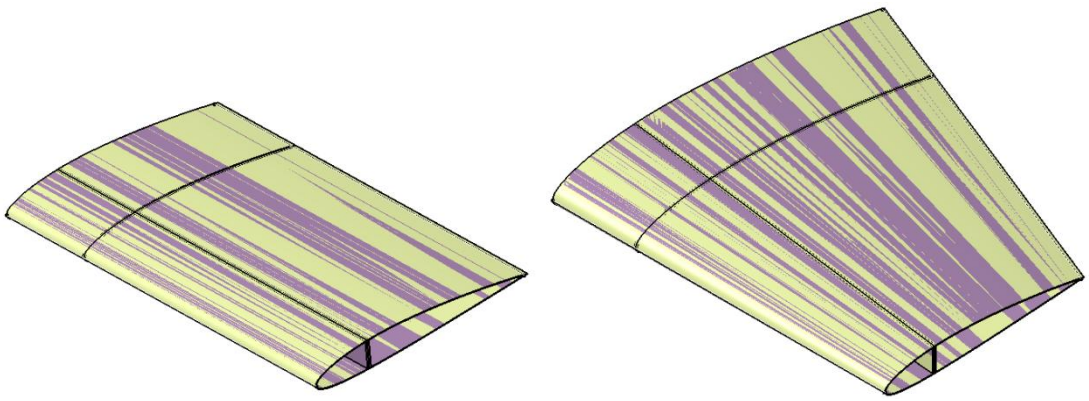


Figure 43: Increasing Root Airfoil Chord from 3 to 5 meters

12.1.5 Tip Airfoil Chord

The tip airfoil chord defines the chord length of the tip of the wing panel. As, the tip airfoil chord length is changed, the airfoil of the trapezoidal wing panel is sized accordingly. Furthermore, the tip and the root chord of each successive wing panel is linked together. So, e.g. when the tip chord of the 1st wing panel is changed from say 3m to 2m, then the root chord of the second wing panel assumes the same value as well. This ensures that the all the wing panels are geometrically connected to each other.

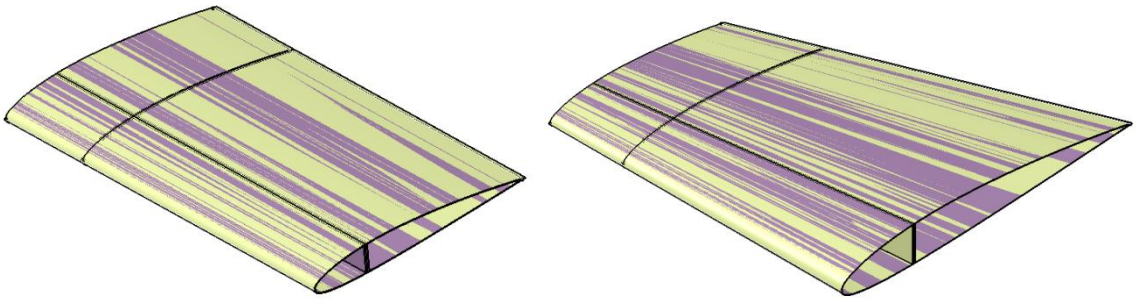


Figure 44: Increasing Tip Airfoil Chord from 3 to 5 meters

12.1.6 Wing Panel Leading Edge Sweep Angle

The wing panel leading edge sweep angle parameter changes the sweep angle of the trapezoidal wing panel. Each individual wing panel can have a different leading edge sweep angle.

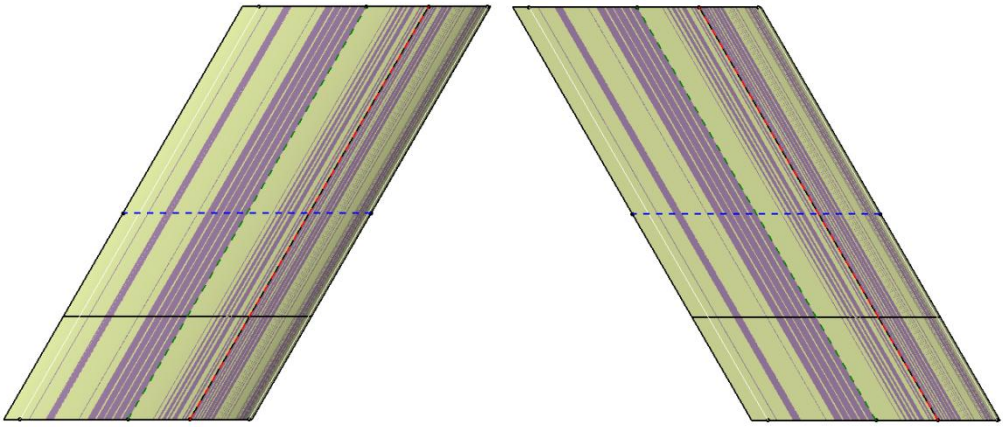


Figure 45: Wing Panel with a forward swept leading edge and a backward swept leading edge (angle: +-30 degree)

12.1.7 Dihedral Angle

A dihedral angle is basically an upward angle that is measured from the horizontal of the wings. It shows that how much the tip of the wing is raised above the root of the wing. A dihedral angle of zero means that the tip and the root of the wing are at the same level, while, a negative dihedral angle means that the tip of the wing is below the root of the wing. A negative dihedral angle is called an anhedral angle. Both positive and negative values of the dihedral angle can be placed put in the dihedral angle parameter and the geometry will reflect the change based on this parameter.

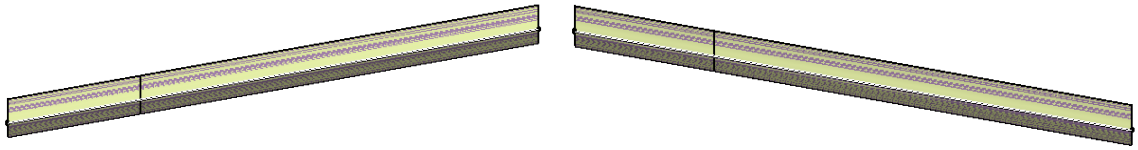


Figure 46: Wing Panel with a dihedral and an anhedral angle of 10 degrees

12.1.8 Root Airfoil Rotation Point along chord w.r.t. z-axis

An aircraft wing or more generally any wing shape can have a variety of twist and angles associated with it. In order to cater for this, rotation point and angles are defined along the y and z-axis of the wing panel. A "Rotation point" parameter can have a value between zero and one and defines the location along which the rotation will take place. The "Root Airfoil Rotation Point along chord w.r.t. z-axis" defines the rotation point along the root chord of the wing with respect to the z-axis of the wing.

12.1.9 Tip Airfoil Rotation Point along chord w.r.t. z-axis

Similar to the case of the root airfoil rotation point, the tip airfoil rotation point, defines the location of the tip airfoil rotation point along chord with respect to the z-axis of the wing panel.

12.1.10 Root Airfoil Rotation Point along chord w.r.t. y-axis

An aircraft wing panel can also have rotation along the y-axis of the wing. A rotation point is defined which can have a value between zero and one, and defines the point along which the rotation of the root of the wing panel will take place with respect to the y-axis of the wing panel.

12.1.11 Tip Airfoil Rotation Point along chord w.r.t y-axis

Similar to the case of the tip airfoil rotation point, the tip airfoil rotation point, defines the location of the tip airfoil rotation point along chord with respect to the y-axis of the wing panel.

12.1.12 Root Airfoil Rotation w.r.t. z-axis

After the rotation points with respect to the y and z axis are defined, the rotation or twist angle for the root and the tip of the wing panel can be defined. An aircraft wing or more generally any wing shape can have rotation in x, y and z-axis. The "root airfoil rotation w.r.t z-axis" parameter defines the angle of twist or rotation of the root of the wing panel with respect to the z-axis.

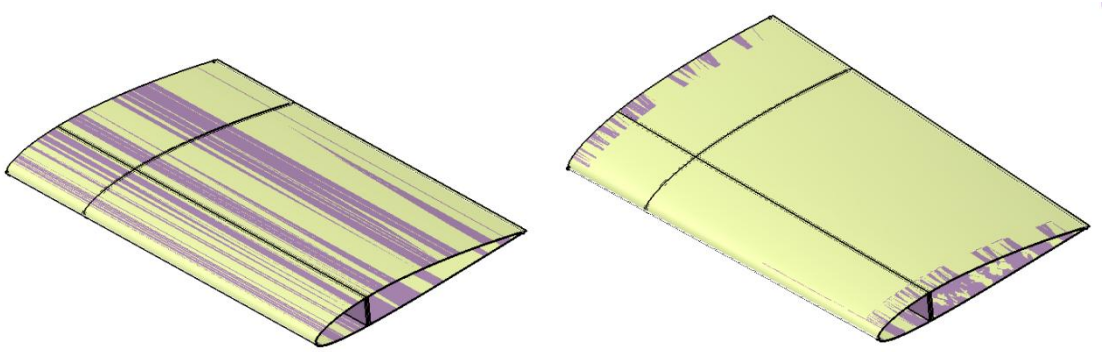


Figure 47: Root Airfoil Rotation w.r.t z-axis by 10 degrees

12.1.13 Tip Airfoil Rotation w.r.t z-axis

As for the root of the wing, the tip of the wing panel can also have a twist or rotation attach to it. A tip airfoil rotation defines the angle of rotation or twist of the tip of the wing with respect to the z-axis of the wing panel.

12.1.14 Root Airfoil Rotation w.r.t x-axis

"The root airfoil rotation w.r.t x-axis" parameter defines the rotation or twist of the wing panel along the x-axis.

12.1.15 Tip Airfoil Rotation w.r.t x-axis

Similar to the case of the root airfoil rotation parameter, the "tip airfoil rotation" parameter defines the rotation or twist of the tip of the wing panel with respect to the x-axis of the wing panel.

12.1.16 Root Airfoil Rotation w.r.t y-axis

An aircraft wing panel can also have rotation along the y-axis of the wing. The "root airfoil rotation w.r.t y-axis" parameter defines the rotation or twist angle of the root of the wing with respect to the y-axis of the wing panel.

12.1.17 Tip Airfoil Rotation w.r.t y-axis

Similar to the case of the root airfoil rotation angle, the "tip airfoil rotation w.r.t. y-axis" parameter changes the tip airfoil rotation with respect to the y-axis of the wing.

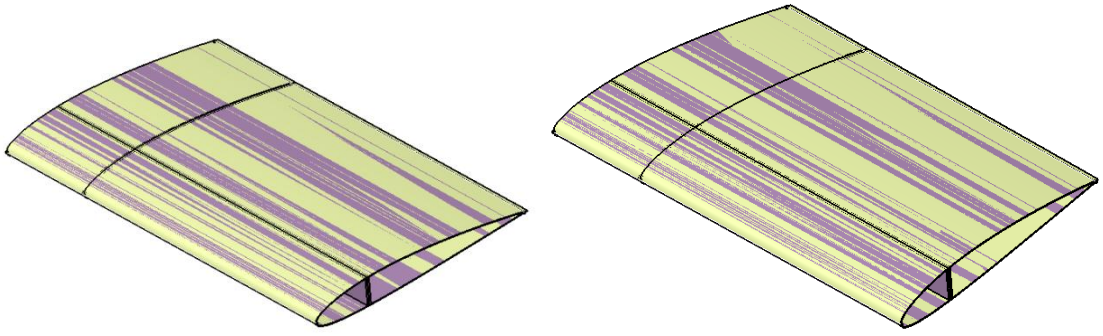


Figure 48: Tip Airfoil Rotation w.r.t y-axis by 10 degrees

12.1.18 Wing Panel Area

A surface is shown below each wing panel which shows the projected surface area of the wing panel. As, the geometry of the wing panel is changed, for example, due to a change in

the wing panel span, the dihedral angle, twist, rotation or sweep angle etc, the projected surface changes accordingly. The area of this projected surface is measured. This gives the wing panel area. When all the wing panel areas are added together, it gives the global wing area of the entire wing.

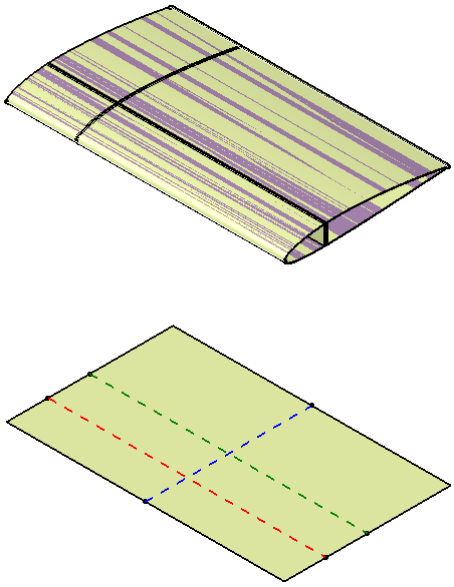


Figure 49: Surface showing wing panel area below the wing

12.1.19 Taper Ratio

The taper ratio is the ratio between the tip chord and the root chord. It is dimensionless quantity and is defined as,

$$Taper\ Ratio = \gamma = \frac{Tip\ Chord}{Root\ Chord}$$

The taper ratio defines the shape of the wing panel. A taper ratio of "1" means that the wing panel is of a rectangular shape with the root and tip chord are equal. Any other value of taper ratio means that the wing panel has a tapered shape. A taper ratio of 0 means that the aircraft wing planform has a delta shape.

12.1.20 Mean Aerodynamic Chord

For a trapezoidal wing panel, the Mean Aerodynamic Chord (MAC) is given by,

$$MAC = r_{chord} * \left(\frac{2}{3}\right) * \left(\frac{1+\gamma+\gamma^2}{1+\gamma}\right)$$

Where,

$$r_{chord} = Root\ Chord\ of\ the\ Wing\ Panel$$

$$\gamma = Taper\ Ratio = \frac{Root\ Chord}{Tip\ Chord}$$

The mean aerodynamic chord has units of meters (m).

The mean aerodynamic chord for the entire wing can be calculated by,

$$MAC\ of\ the\ entire\ wing = \frac{\sum_{i=1}^{n}(MAC_i)(Area_i)}{\sum_{i=1}^{n}(Area_i)}$$

$$MAC = \frac{(MAC_1)(Area_1) + (MAC_2)(Area_2) + (MAC_3)(Area_3) + \cdots + (MAC_n)(Area_n)}{Total\ Area}$$

12.1.21 X position of the Mean Aerodynamic Chord (x_MAC)

For a trapezoidal wing panel, the "x" position of the mean aerodynamic chord is given by,

$$x_{MAC} = \frac{b_{span} * \tan(LE_{sweepangle}) * r_{chord} + 2 * r_{tip}}{3 * r_{chord} + r_{tip}}$$

Where,

$$b_{span} = Wing\ Panel\ Span$$
$$LE_{sweepangle} = Leading\ Edge\ Sweep\ Angle$$
$$r_{chord} = Wing\ Panel\ Root\ Chord$$
$$r_{tip} = Wing\ Panel\ Tip\ Chord$$

12.1.22 Y position of the Mean Aerodynamic Chord (y_MAC)

For a trapezoidal wing panel, the "y" position of the mean aerodynamic chord is given by,

$$y_{MAC} = \frac{2 * b_{span} * \left(\frac{1}{2}\right) * r_{chord} + r_{tip}}{3 * r_{chord} + r_{tip}}$$

Where,

$$b_{span} = Wing\ Panel\ Span$$
$$r_{chord} = Wing\ Panel\ Root\ Chord$$
$$r_{tip} = Wing\ Panel\ Tip\ Chord$$

12.1.23 Aspect Ratio

For a trapezoidal wing panel, the Aspect ratio is defined as,

$$AR = \frac{b^2}{S}$$

Where,

$$S = Total\ Area\ of\ the\ Wing\ Panel\ (symmetric)$$
$$b = Total\ span\ of\ the\ Wing\ Panel\ (symmetric)$$

12.1.24 Wing Panel Skin Thickness

The wing panel skin thickness defines the thickness of the wing panel. The thickness is added equally to all sides of the wing surface.

13 Appendix C

13.1 Tables of comparison between PC with VB Scripting and KP Approach

13.1.1 Test # 1: Time to Instantiate wing panels

A comparison between the times to instantiate wing panels in both approaches is shown in the table below,

Number of Wing Panels	*Knowledge Pattern Approach*	*Power Copy with VB Scripting Approach*
3	11.9 sec	45.2 sec
5	16.4 sec	1m 45 sec
10	37.8 sec	6m 50 sec

Table 2: Time to Instantiate wing panels between the two approaches

13.1.2 Test # 2: Time to delete wing panels

A comparison between the times to delete the wing panels in both approaches is shown in the table below,

Number of Wing Panels	*Knowledge Pattern Approach*	*Power Copy with VB Scripting Approach*
3 to 1	2.8 sec	57.2 sec
5 to 1	4.1 sec	2m 5 sec
10 to 1	6.9 sec	6m 37 sec

Table 3: Time to delete wing panels between the two approaches

13.1.3 Test # 3: Time to Instantiate wing spars

A comparison between the times to instantiate wing spars in both approaches is shown in the table below. In this case, the number of wing panels is set to one for both approaches.

Number of Wing Spars	*Knowledge Pattern Approach*	*Power Copy with VB Scripting Approach*
3	3.1 sec	12 sec
5	3.5 sec	22.3 sec
10	6.1 sec	40.9 sec

Table 4: Time to instantiate wing spars between the two approaches

13.1.4 Test # 4: Time to delete wing spars

A comparison between the times to delete wing spars in both approaches is shown in the table below. In this case, the number of wing panels is set to one for both approaches.

Number of Wing Spars	*Knowledge Pattern Approach*	*Power Copy with VB Scripting Approach*
3 to 1	1.3 sec	11 sec
5 to 1	1.5 sec	18.2 sec
10 to 1	1.9 sec	33.6 sec

Table 5: Time to delete wing spars between the two approaches

13.1.5 Test # 5: Time to instantiate wing ribs

A comparison between the times to instantiate wing ribs in both approaches is shown in the table below. In this case, the number of wing panels is set to one for both approaches.

Number of Wing Ribs	*Knowledge Pattern Approach*	*Power Copy with VB Scripting Approach*
3	2.5 sec	11 sec
5	3.5 sec	20 sec
10	6.8 sec	44 sec

Table 6: Time to instantiate wing ribs between the two approaches

13.1.6 Test # 6: Time to delete wing ribs

A comparison between the times to delete wing ribs in both approaches is shown in the table below. In this case, the number of wing panels is set to one for both approaches.

Number of Wing Ribs	*Knowledge Pattern Approach*	*Power Copy with VB Scripting Approach*
3 to 1	1.2 sec	5.3 sec
5 to 1	1.4 sec	9 sec
10 to 1	1.6 sec	18.9 sec

Table 7: Time to delete wing ribs between the two approaches

14 Appendix D

14.1 Aircraft Wing Design Examples built by using generic model

The generic aircraft wing model can be used for designing any kind of wing planform shape or configuration.

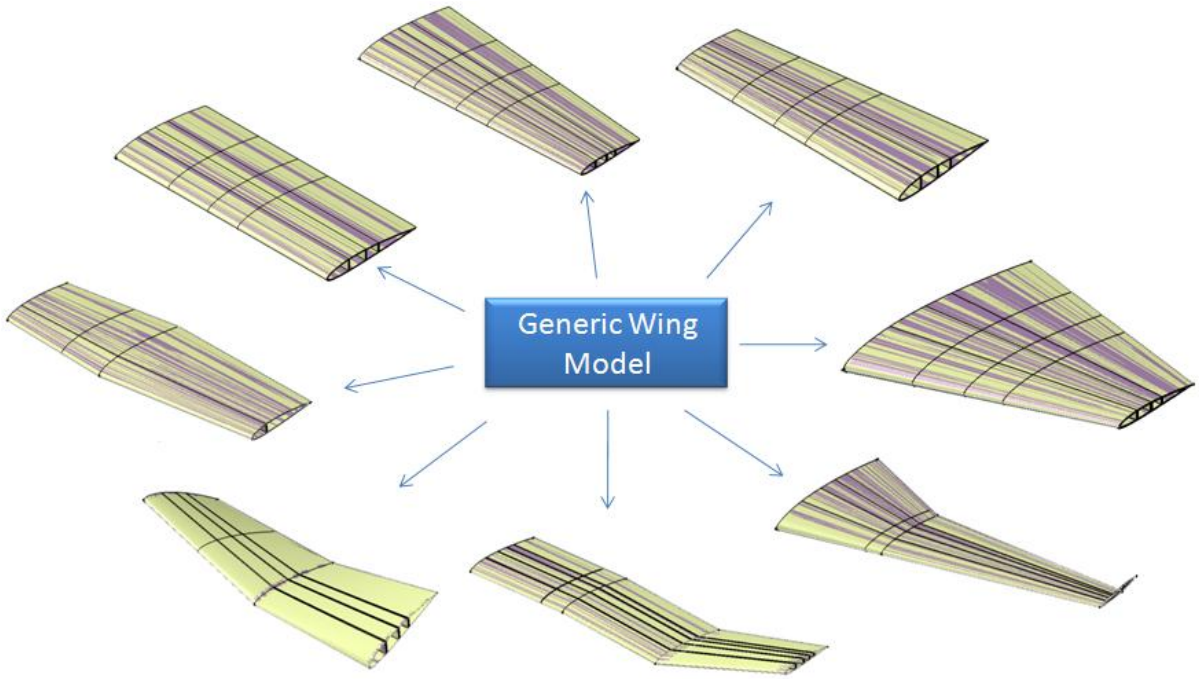


Figure 50: Examples of aircraft wing planform using generic wing model

Following are presented some examples of different aircraft wing planform and designs that can be achieved by using the generic wing model.

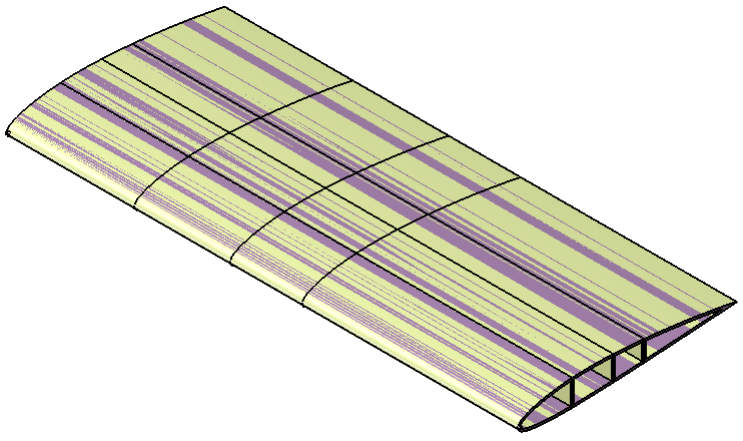


Figure 51: A straight rectangular wing design

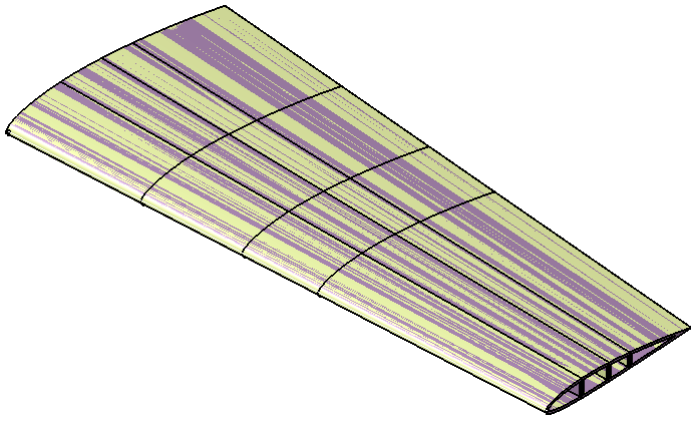


Figure 52: A tapered wing design

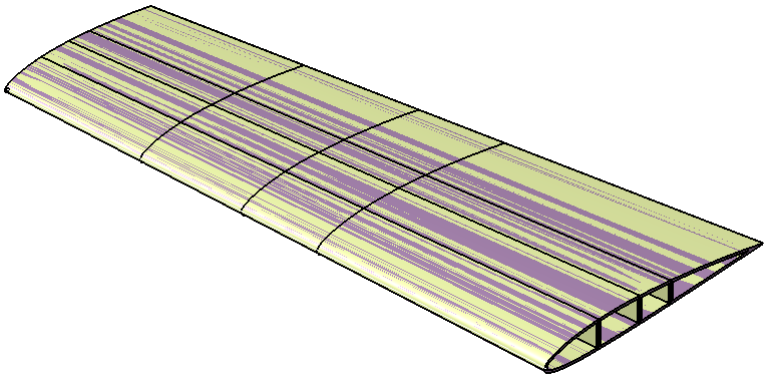


Figure 53: A reversed tapered wing design

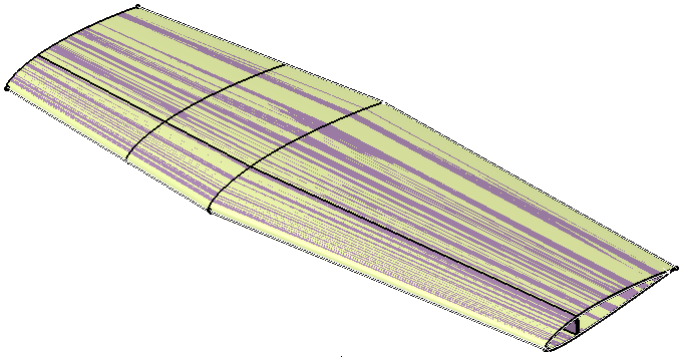


Figure 54: A compound tapered wing

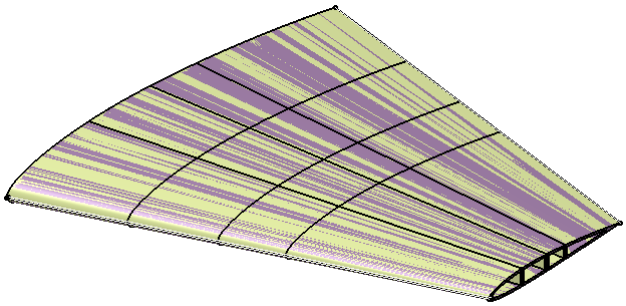


Figure 55: A trapezoidal wing design

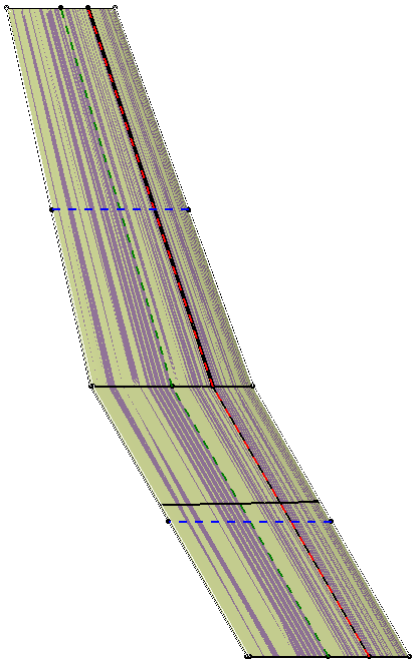


Figure 56: A crescent wing design (top view)

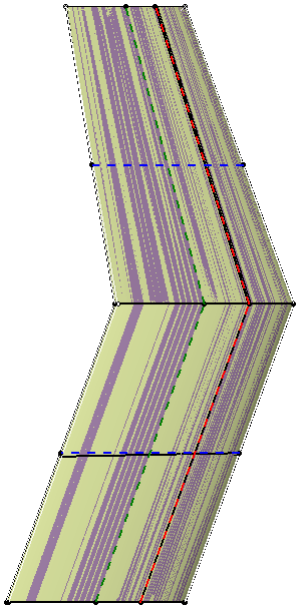


Figure 57: An M-wing design (top view)

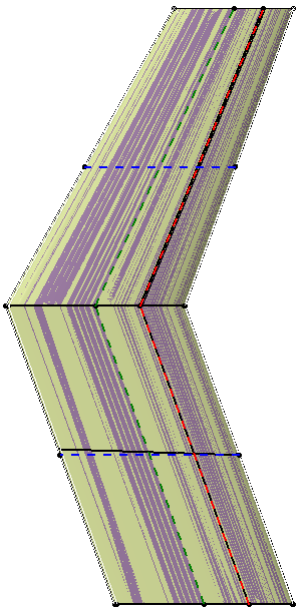


Figure 58: A W-wing design (top view)

15 Appendix E

15.1 File Structure for PowerCopy with VB Scripting Approach

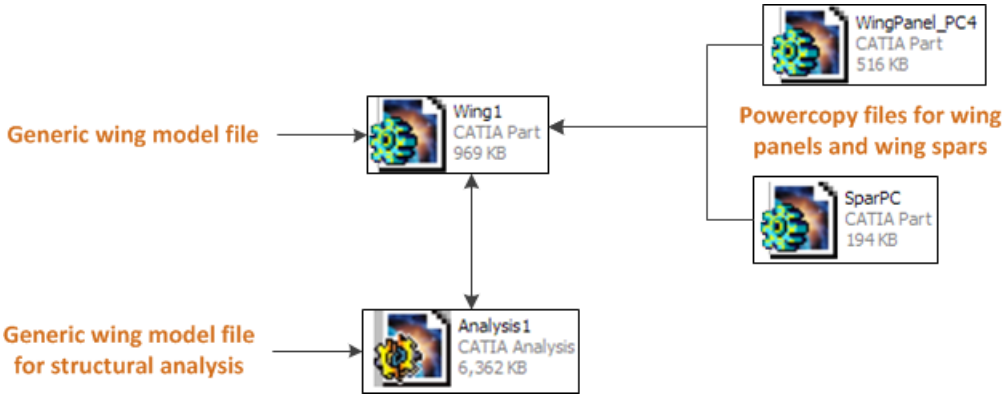


Figure 59: File structure for powercopy with VB scripting approach

15.2 File Structure for Knowledge Pattern Approach

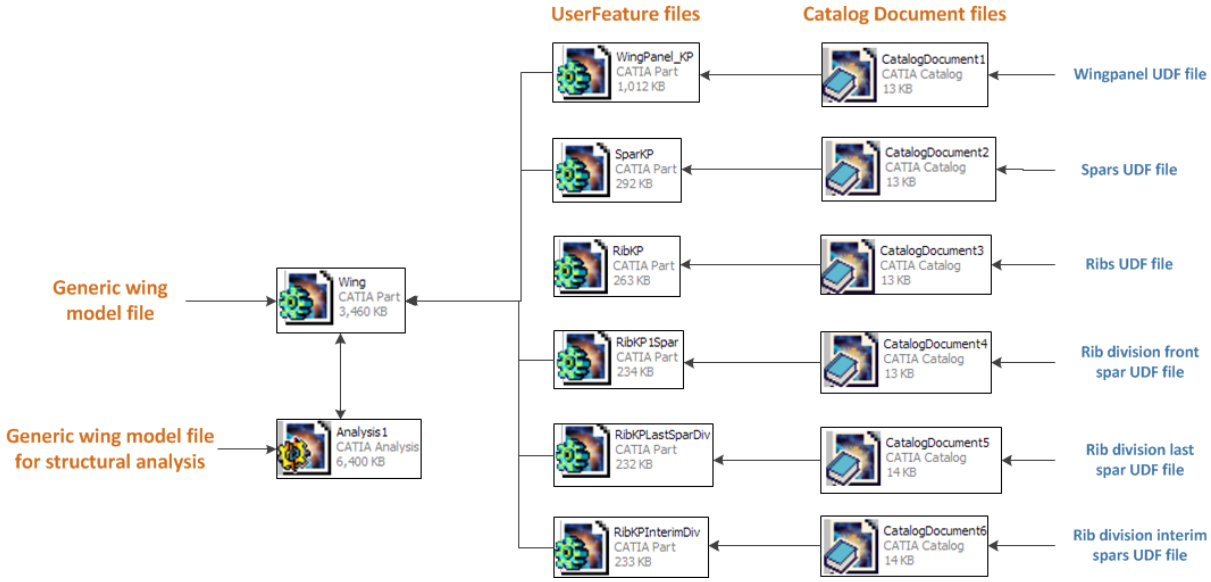


Figure 60: File structure for knowledge pattern approach

16 Appendix F

16.1 Knowledge Pattern (KP) Generic Wing Model Code

```
let udf (UserFeature)
let oldudf (UserFeature)
let sparudf (UserFeature)
let oldsparudf (UserFeature)
let ribudf (UserFeature)
let oldribudf (UserFeature)
let sparinterimudf (UserFeature)
let oldsparinterimudf (UserFeature)
let sparinterimmwpudf (UserFeature)
let oldsparinterimmwpudf (UserFeature)
let ribdivision (UserFeature)
let oldribdivision (UserFeature)
let xx (UserFeature)
let yy (UserFeature)

let i (Integer) /* Wing Panel */
let j (Integer) /* Spar */
let k (Integer) /* Rib */
let l (Integer) /* Join WP */
let b (Integer) /* Join WP LS */
let c (Integer) /* Join Line Rib Front */
let d (Integer) /* Join Line Rib Aft */
let e (Integer) /* Spar Interim UDF */
let x (Integer) /* count for wing panels*/
let g (Integer) /* spar interim more wing panel udf */
let y (Integer) /* count for wing panels*/
let m (Integer)
let f (Integer)
let s (Integer)
let q (Integer)
let w (Integer)
let u (Integer)
let h (Integer) /* Rib Join */
let vv (Integer) /* Spar Join */
let www (Integer) /*Linea Join */
let zzz (Integer) /*Lineb Join */

let surface1 (Surface)
let surface2 (Surface)
let surface3 (Surface)
let surface4 (Surface)
let surface5 (Surface)
let surface6 (Surface)
let join (Surface)
let joinls (Surface)
let linerib (Curve)
let linerib1 (Curve)
let loftguide (Curve)
let loftguide1(Curve)
let lineajoin (Curve)
let linebjoin (Curve)
let linejoin (Curve)
let linejoin1 (Curve)
let ribjoin (Surface)
let sparjoin (Surface)
let sparsurfacea (Surface)
let sparsurfaceb (Surface)
let interimspara (Surface)
let interimspara1 (Surface)
let interimsparb (Surface)
let interimsparb1 (Surface)
let interimsparajoin (Surface)
let interimsparbjoin (Surface)
```

```
let interimsparajoinx (Surface)
let interimsparbjoinx (Surface)

/* ********************************************** */
let loftcurve (Curve)
let loftcurve1 (Curve)
let loftnr (Integer)
let orient1 (Integer)
let orient2 (Integer)
let lofty (Surface)
let loftynr (Integer)
let guidey (Curve)
let guideynr (Integer)
let size (Integer)
let size1 (Integer)
let riblistsize (Integer)
let lastspar (Integer)

let wpls (Surface)
let wplsnr (Integer)
let wplsspine (Curve)
let wplsspinenr (Integer)
let wplsextp (Point)
let wplsextpnr (Integer)
let triggerv (Integer)
let lineawpls (Curve)
let linebwpls (Curve)
let lineawplsnr (Integer)
let linebwplsnr (Integer)
let lineaguidept (Point)
let linebguidept (Point)
let lineaguideptnr (Integer)
let linebguideptnr (Integer)
let ribiv (Surface)
let ribivnr (Integer)
let ribdivisionnr (Integer)
let countspar (Integer)
let countsparmr (Integer)
let interimsparanr (Integer)
let interimsparbnr (Integer)
let sizesparlist (Integer)
let num (Integer)
let macwp1 (Real)
let areawp1 (Real)
let macmarea1 (Real)
let macwp (Real)
let areawp (Real)
let macmarea (Real)
let totalarea (Real)

num = 1
ribivnr = 1
ribdivisionnr = 1
wplsnr = 1
wplsspinenr = 1
wplsextpnr = 1
lineawplsnr = 1
linebwplsnr = 1
triggerv = 1
lineaguideptnr = 1
linebguideptnr = 1


loftnr = 1
orient1 = 1
orient2 = 1
loftynr = 1
guideynr = 1
```

```
interimsparanr = 1
interimsparbnr = 1

/* ********************************************** */

i = 1
l = 1
b = 1
c = 1
d = 1
e = 1
x = 1
g = 1
y = 1
m = 1
f = 1
s = 1
q = 2
w = 1
x = 1
vv = 1
www = 1
zzz = 1

/**************/
/* Wing Panel */
/**************/

For i while i<=`New Number of Wing Panels`
{

if i ==1
       {
udf = CreateOrModifyTemplate("CatalogDocument1|UserFeature1",`Instantiated
Geometry`  ,`Relations\Knowledge Pattern.1\UDFs` ,i)

udf->SetAttributeObject("Point.1",`Base Geometry\Point.1`  )
udf->SetAttributeObject("Plane.1", `Base Geometry\Plane.1`  )
EndModifyTemplate(udf)

udf = `Relations\Knowledge Pattern.1\UDFs`  ->GetItem(i)
set surface1 = udf->GetAttributeObject("Multi-sections Surfacewing")
`Base Geometry\MeshSurfaceWing`  = surface1

udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(i)
set surface2 = udf->GetAttributeObject("WingPanelLowerSurface")
`Base Geometry\WingLowerSurface` = surface2

udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(i)
set linerib = udf->GetAttributeObject("LineforPlacingRibs")
`Base Geometry\LineforRibsFront` = linerib

udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(i)
set linerib1 = udf->GetAttributeObject("LineforPlacingRibs2")
`Base Geometry\LineforRibsAft` = linerib1

udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(i)
set loftguide = udf->GetAttributeObject("Linea")
`Base Geometry\Linea` =loftguide

udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(i)
set loftguide1 = udf->GetAttributeObject("Lineb")
`Base Geometry\Lineb` =loftguide1

`Total Wing Area` = udf->GetAttributeReal("WingPanel Area")
macwp1 = udf->GetAttributeReal("Mean Aerodynamic Chord")
areawp1 = udf->GetAttributeReal("WingPanel Area")
macmarea1 = macwp1*areawp1
```

```
`Wing Mean Aerodynamic Chord` = macmarea1/`Total Wing Area`
      }

if i>1
      {
udf =
CreateOrModifyTemplate("CatalogDocument1|UserFeature1",`Instantiated Geometry`
,`Relations\Knowledge Pattern.1\UDFs` ,i)

oldudf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(i-1)
udf->SetAttributeObject("Point.1", oldudf->GetAttributeObject("TALE"))
udf->SetAttributeObject("Plane.1", oldudf->GetAttributeObject("Tplane"))

udf->SetAttributeString("`Root Airfoil`", oldudf->GetAttributeString("`Tip
Airfoil`"))
udf->SetAttributeReal("`Root Airfoil Chord`", oldudf->GetAttributeReal("`Tip
Airfoil Chord`"))
udf->SetAttributeReal("`Root Airfoil Rotation Point along chord wrt z-axis`",
oldudf>GetAttributeReal("`Tip Airfoil Rotation Point along chord wrt z-axis`"))
udf->SetAttributeReal("`Root Airfoil Rotation Point along chord wrt y-axis`",
oldudf>GetAttributeReal("`Tip Airfoil Rotation Point along chord wrt y-axis`"))
udf->SetAttributeReal("`Root Airfoil Rotation wrt z-axis`",
oldudf>GetAttributeReal("`Tip Airfoil Rotation wrt z-axis`"))
udf->SetAttributeReal("`Root Airfoil Rotation wrt y-axis`",
oldudf>GetAttributeReal("`Tip Airfoil Rotation wrt y-axis`"))
udf->SetAttributeReal("`Root Airfoil Rotation wrt x-axis`",
oldudf>GetAttributeReal("`Tip Airfoil Rotation wrt x-axis`"))
EndModifyTemplate(udf)

wpls = CreateOrModifyDatum("Surface",`Temporary Geometry`  ,`Relations\Knowledge
Pattern.1\surfacewplower` ,wplsnr)
set wpls = udf->GetAttributeObject("WingPanelLowerSurface")
wpls.Name = "WingPanelLowerSurface." + ToString(wplsnr)
wplsnr = wplsnr + 1

wplsspine = CreateOrModifyDatum("Curve",`Temporary Geometry`  ,`Relations\Knowledge
Pattern.1\Curvelistwpls`  ,wplsspinenr)
set wplsspine = udf->GetAttributeObject("Splineb")
wplsspine.Name = "Splineb." + ToString(wplsspinenr)
wplsspinenr = wplsspinenr + 1

wplsextp = CreateOrModifyDatum("Point",`Temporary Geometry`  ,`Relations\Knowledge
Pattern.1\ExtremumPointlist`   ,wplsextpnr)
set wplsextp = udf->GetAttributeObject("ExtremumToUse")
wplsextp.Name = "ExtremumToUse." + ToString(wplsextpnr)
wplsextpnr = wplsextpnr + 1

lineawpls = CreateOrModifyDatum("Curve",`Temporary Geometry`  ,`Relations\Knowledge
Pattern.1\LineaList`   ,lineawplsnr)
set lineawpls = udf->GetAttributeObject("Linea")
lineawpls.Name = "Linea." + ToString(lineawplsnr)
lineawplsnr = lineawplsnr + 1

linebwpls = CreateOrModifyDatum("Curve",`Temporary Geometry`  ,`Relations\Knowledge
Pattern.1\linebList`    ,linebwplsnr)
set linebwpls = udf->GetAttributeObject("Lineb")
linebwpls.Name = "Lineb." + ToString(linebwplsnr)
linebwplsnr = linebwplsnr + 1

lineaguidept = CreateOrModifyDatum("Point",`Temporary Geometry`
,`Relations\Knowledge Pattern.1\Lineaptguidelist`    ,lineaguideptnr)
set lineaguidept = udf->GetAttributeObject("pLinea-2")
lineaguidept.Name = "pLinea." + ToString(lineaguideptnr)
lineaguideptnr = lineaguideptnr + 1

linebguidept = CreateOrModifyDatum("Point",`Temporary Geometry`
,`Relations\Knowledge Pattern.1\Linebptguidelist`    ,linebguideptnr)
set linebguidept = udf->GetAttributeObject("pLineb-2")
```

```
linebguidept.Name = "pLineb." + ToString(linebguideptnr)
linebguideptnr = linebguideptnr + 1


join = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\JoinWP` ,l)
udf = `Relations\Knowledge Pattern.1\UDFs`  ->GetItem(i)
set surface1 = udf->GetAttributeObject("Multi-sections Surfacewing")
set surface2 = `Base Geometry\MeshSurfaceWing`
join = assemble(surface1 ,surface2)
join.Name = "InterimJoinWing." + ToString(l)
l = l+1
`Base Geometry\MeshSurfaceWing`  = join

joinls = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\JoinWPLS` ,b)
udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(i)
set surface3 = udf->GetAttributeObject("WingPanelLowerSurface")
set surface4 = `Base Geometry\WingLowerSurface`
joinls = assemble(surface3,surface4)
joinls.Name = "InterimWingLowerSurfaceJoin." + ToString(i)
b=b+1
`Base Geometry\WingLowerSurface` = joinls

linejoin = CreateOrModifyDatum("Curve",`Base Geometry` ,`Relations\Knowledge
Pattern.1\JoinLineRibFront` ,c)
udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(i)
set linerib = udf->GetAttributeObject("LineforPlacingRibs")
set linerib1 = `Base Geometry\LineforRibsFront`
linejoin = assemble (linerib,linerib1)
c = c+1
`Base Geometry\LineforRibsFront` = linejoin

linejoin1 = CreateOrModifyDatum("Curve",`Base Geometry` ,`Relations\Knowledge
Pattern.1\JoinLineRibAft` ,d)
udf = `Relations\Knowledge Pattern.1\UDFs`  ->GetItem(i)
set linerib = udf->GetAttributeObject("LineforPlacingRibs2")
set linerib1 = `Base Geometry\LineforRibsAft`
linejoin1 = assemble(linerib,linerib1)
d = d+1
`Base Geometry\LineforRibsAft` = linejoin1

lineajoin = CreateOrModifyDatum("Curve",`Base Geometry` ,`Relations\Knowledge
Pattern.1\Lineajoin`  ,www)
udf = `Relations\Knowledge Pattern.1\UDFs`  ->GetItem(i)
set loftguide = udf->GetAttributeObject("Linea")
set loftguide1 =`Base Geometry\Linea`
lineajoin = assemble(loftguide,loftguide1)
www = www+1
`Base Geometry\Linea`  = lineajoin

linebjoin = CreateOrModifyDatum("Curve",`Base Geometry` ,`Relations\Knowledge
Pattern.1\Linebjoin`   ,zzz)
udf = `Relations\Knowledge Pattern.1\UDFs`  ->GetItem(i)
set loftguide = udf->GetAttributeObject("Lineb")
set loftguide1 =`Base Geometry\Lineb`
linebjoin = assemble(loftguide,loftguide1)
zzz = zzz+1
`Base Geometry\Lineb`   = linebjoin

triggerv = triggerv +1

`Total Wing Area` = `Total Wing Area`+ udf->GetAttributeReal("WingPanel Area")
macwp = udf->GetAttributeReal("Mean Aerodynamic Chord")
areawp = udf->GetAttributeReal("WingPanel Area")
macmarea1 = macmarea1 + macwp*areawp
`Wing Mean Aerodynamic Chord` = macmarea1/`Total Wing Area`
```

```
	}

udf.Name = "WingPanel." + ToString(i)
}

/*********/
/* Spars */
/*********/

j = 1

For j while j<=`New Number of Spars`
{
	/* ***************************** */
	/* IF THERE IS ONLY ONE WING PANEL */
	/* ***************************** */

if `New Number of Wing Panels` ==1
	{
if j==1
	{
udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(1)
sparudf = CreateOrModifyTemplate("CatalogDocument2|UserFeature1",`Instantiated
Geometry` ,`Relations\Knowledge Pattern.1\SparUDFs` ,j)
sparudf->SetAttributeObject("Line1",udf->GetAttributeObject("Line1") )
sparudf->SetAttributeObject("Sp1", udf->GetAttributeObject("SP1") )
sparudf->SetAttributeObject("Line2",udf->GetAttributeObject("Line2") )
sparudf->SetAttributeObject("Sp2",  udf->GetAttributeObject("SP3") )
sparudf-
>SetAttributeObject("PlaneWingPanelLefty",udf>GetAttributeObject("PlaneWingPanelLef
t") )
sparudf-
>SetAttributeObject("PlaneWingPanelRighty",udf>GetAttributeObject("PlaneWingPanelRi
ght")  )
sparudf->SetAttributeObject("Multi-sections Surfacewing",udf-
>GetAttributeObject("Multi-sections Surfacewing") )
sparudf-
>SetAttributeObject("WingPanelLowerSurface",udf>GetAttributeObject("WingPanelLowerS
urface")  )

EndModifyTemplate(sparudf)
sparudf.Name = "Spar." + ToString(j)

sparudf = `Relations\Knowledge Pattern.1\SparUDFs` ->GetItem(j)
set surface1 = sparudf->GetAttributeObject("SparSurface")
`Base Geometry\SparsSurface`  = surface1

interimspara = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\InterimSparSAlist` ,j)
set interimspara = sparudf->GetAttributeObject("SparSurfaceA")
interimspara.Name = "SparSurfaceA." + ToString(j)

interimsparb = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\InterimSparSBlist`  ,j)
set interimsparb = sparudf->GetAttributeObject("SparSurfaceB")
interimsparb.Name = "SparSurfaceB." + ToString(j)


		}

if j>1
		{
oldsparudf = `Relations\Knowledge Pattern.1\SparUDFs` .GetItem(j-1)
udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(1)

sparudf = CreateOrModifyTemplate("CatalogDocument2|UserFeature1",`Instantiated
Geometry` ,`Relations\Knowledge Pattern.1\SparUDFs` ,j)
sparudf->SetAttributeObject("Line1",udf->GetAttributeObject("Line1") )
```

```
sparudf->SetAttributeObject("Sp1", oldsparudf->GetAttributeObject("Pointa") )
sparudf->SetAttributeObject("Line2",udf->GetAttributeObject("Line2") )
sparudf->SetAttributeObject("Sp2", oldsparudf->GetAttributeObject("Pointb") )
sparudf->SetAttributeObject("PlaneWingPanelLefty",udf-
>GetAttributeObject("PlaneWingPanelLeft")   )
sparudf->SetAttributeObject("PlaneWingPanelRighty",udf-
>GetAttributeObject("PlaneWingPanelRight")   )
sparudf->SetAttributeObject("Multi-sections Surfacewing", udf-
>GetAttributeObject("Multi-sections Surfacewing")    )
sparudf->SetAttributeObject("WingPanelLowerSurface",udf-
>GetAttributeObject("WingPanelLowerSurface")   )

EndModifyTemplate(sparudf)
sparudf.Name = "Spar." + ToString(j)

sparjoin = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\SparJoin`  ,vv)
sparudf = `Relations\Knowledge Pattern.1\SparUDFs` ->GetItem(j)
set surface1 = sparudf->GetAttributeObject("SparSurface")
set surface2 = `Base Geometry\SparsSurface`
sparjoin = assemble(surface1,surface2)
sparjoin.Name = "SparJoinInterim." + ToString(vv)
vv=vv+1
`Base Geometry\SparsSurface`  = sparjoin

interimspara = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\InterimSparSAlist` ,j)
set interimspara = sparudf->GetAttributeObject("SparSurfaceA")
interimspara.Name = "SparSurfaceA." + ToString(j)


interimsparb = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\InterimSparSBlist`  ,j)
set interimsparb = sparudf->GetAttributeObject("SparSurfaceB")
interimsparb.Name = "SparSurfaceB." + ToString(j)

            }
      }


      /* ********************************** */
      /* IF THERE IS MORE THAN ONE WING PANEL */
      /* ********************************** */

if `New Number of Wing Panels` >1
      {
if j==1
      {
udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(1)

sparudf = CreateOrModifyTemplate("CatalogDocument2|UserFeature1",`Instantiated
Geometry` ,`Relations\Knowledge Pattern.1\SparUDFs` ,j)
sparudf->SetAttributeObject("Line1",udf->GetAttributeObject("Line1") )
sparudf->SetAttributeObject("Sp1", udf->GetAttributeObject("SP1") )

sparudf->SetAttributeObject("Line2",udf->GetAttributeObject("Line2") )
sparudf->SetAttributeObject("Sp2", udf->GetAttributeObject("SP3") )
sparudf->SetAttributeObject("PlaneWingPanelLefty",udf-
>GetAttributeObject("PlaneWingPanelLeft")   )
sparudf->SetAttributeObject("PlaneWingPanelRighty",udf-
>GetAttributeObject("PlaneWingPanelRight")   )
sparudf->SetAttributeObject("Multi-sections Surfacewing",udf-
>GetAttributeObject("Multi-sections Surfacewing")     )
sparudf->SetAttributeObject("WingPanelLowerSurface",udf-
>GetAttributeObject("WingPanelLowerSurface")    )

EndModifyTemplate(sparudf)
sparudf.Name = "Spar." + ToString(j)
```

```
sparudf = `Relations\Knowledge Pattern.1\SparUDFs` ->GetItem(j)
set surface1 = sparudf->GetAttributeObject("SparSurface")
`Base Geometry\SparsSurface`  = surface1

interimspara = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\InterimSparSAlist` ,j)
set interimspara = sparudf->GetAttributeObject("SparSurfaceA")
interimspara.Name = "SparSurfaceA." + ToString(j)


interimsparb = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\InterimSparSBlist`  ,j)
set interimsparb = sparudf->GetAttributeObject("SparSurfaceB")
interimsparb.Name = "SparSurfaceB." + ToString(j)

x=1
For x while x <`New Number of Wing Panels`
      {

udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(x+1)
oldudf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(x)

sparinterimudf =
CreateOrModifyTemplate("CatalogDocument2|UserFeature1",`Instantiated Geometry`
,`Relations\Knowledge Pattern.1\SparInterimUDFs` ,e)
sparinterimudf->SetAttributeObject("Line1",oldudf->GetAttributeObject("Line2") )
sparinterimudf->SetAttributeObject("Sp1", oldudf->GetAttributeObject("SP3") )

sparinterimudf->SetAttributeObject("Line2",udf->GetAttributeObject("Line2") )
sparinterimudf->SetAttributeObject("Sp2", udf->GetAttributeObject("SP3") )
sparinterimudf->SetAttributeObject("PlaneWingPanelLefty",udf-
>GetAttributeObject("PlaneWingPanelLeft")   )
sparinterimudf->SetAttributeObject("PlaneWingPanelRighty",udf-
>GetAttributeObject("PlaneWingPanelRight")   )
sparinterimudf->SetAttributeObject("Multi-sections Surfacewing",udf-
>GetAttributeObject("Multi-sections Surfacewing")    )
sparinterimudf->SetAttributeObject("WingPanelLowerSurface",udf-
>GetAttributeObject("WingPanelLowerSurface")     )


EndModifyTemplate(sparinterimudf)
sparinterimudf.Name = "Spari." + ToString(e)

sparjoin = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\SparJoin`  ,vv)
sparinterimudf = `Relations\Knowledge Pattern.1\SparInterimUDFs` ->GetItem(e)
set surface1 = sparinterimudf ->GetAttributeObject("SparSurface")
set surface2 = `Base Geometry\SparsSurface`
sparjoin = assemble(surface1,surface2)
sparjoin.Name = "SparJoinInterim." + ToString(vv)
vv=vv+1
`Base Geometry\SparsSurface`  = sparjoin

interimsparajoin = CreateOrModifyDatum("Surface",`Base Geometry`
,`Relations\Knowledge Pattern.1\InterimSparSAlist` ,j)
set interimspara = `Relations\Knowledge Pattern.1\InterimSparSAlist` ->GetItem(j)
set interimspara1 = sparinterimudf->GetAttributeObject("SparSurfaceA")
interimsparajoin = assemble(interimspara,interimspara1)
interimsparajoin.Name = "SparSurfaceA." + ToString(j)

interimsparbjoin = CreateOrModifyDatum("Surface",`Base Geometry`
,`Relations\Knowledge Pattern.1\InterimSparSBlist`  ,j)
set interimsparb = `Relations\Knowledge Pattern.1\InterimSparSBlist` ->GetItem(j)
set interimsparb1 = sparinterimudf->GetAttributeObject("SparSurfaceB")
interimsparbjoin = assemble(interimsparb,interimsparb1)
interimsparbjoin.Name = "SparSurfaceB." + ToString(j)
```

```
if x == 1
       {
oldsparudf = `Relations\Knowledge Pattern.1\SparUDFs` .GetItem(j)
sparinterimudf->SetAttributeReal("`SparPositionP1`",oldsparudf-
>GetAttributeReal("`SparPositionP2`")  )
       }

if x>1
       {
oldsparinterimudf = `Relations\Knowledge Pattern.1\SparInterimUDFs` .GetItem(e-1)
sparinterimudf->SetAttributeReal("`SparPositionP1`",oldsparinterimudf-
>GetAttributeReal("`SparPositionP2`")  )
       }
       e = e+1
       x = x+1

       }
}
e = 1
x =1

if j>1
              {
oldsparudf = `Relations\Knowledge Pattern.1\SparUDFs` .GetItem(j-1)
udf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(1)

sparudf = CreateOrModifyTemplate("CatalogDocument2|UserFeature1",`Instantiated
Geometry` ,`Relations\Knowledge Pattern.1\SparUDFs` ,j)
sparudf->SetAttributeObject("Line1",udf->GetAttributeObject("Line1") )
sparudf->SetAttributeObject("Sp1", oldsparudf->GetAttributeObject("Pointa") )
sparudf->SetAttributeObject("Line2",udf->GetAttributeObject("Line2") )
sparudf->SetAttributeObject("Sp2", oldsparudf->GetAttributeObject("Pointb") )
sparudf->SetAttributeObject("PlaneWingPanelLefty",udf-
>GetAttributeObject("PlaneWingPanelLeft")   )
sparudf->SetAttributeObject("PlaneWingPanelRighty",udf-
>GetAttributeObject("PlaneWingPanelRight")    )
sparudf->SetAttributeObject("Multi-sections Surfacewing",udf-
>GetAttributeObject("Multi-sections Surfacewing")     )
sparudf->SetAttributeObject("WingPanelLowerSurface",udf-
>GetAttributeObject("WingPanelLowerSurface")    )

EndModifyTemplate(sparudf)
sparudf.Name = "Spar." + ToString(j)

sparjoin = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\SparJoin`  ,vv)
sparudf = `Relations\Knowledge Pattern.1\SparUDFs` ->GetItem(j)
set surface1 = sparudf->GetAttributeObject("SparSurface")
set surface2 = `Base Geometry\SparsSurface`
sparjoin = assemble(surface1,surface2)
sparjoin.Name = "SparJoinInterim." + ToString(vv)
vv=vv+1
`Base Geometry\SparsSurface`  = sparjoin

interimspara = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\InterimSparSAlist`  ,j)
set interimspara = sparudf->GetAttributeObject("SparSurfaceA")
interimspara.Name = "SparSurfaceA." + ToString(j)

interimsparb = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\InterimSparSBlist`  ,j)
set interimsparb = sparudf->GetAttributeObject("SparSurfaceB")
interimsparb.Name = "SparSurfaceB." + ToString(j)


For x while x<`New Number of Wing Panels`
```

```
	{
if j==2
	{
oldudf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(x)
udf = `Relations\Knowledge Pattern.1\UDFs`  ->GetItem(x+1)

oldsparinterimudf = `Relations\Knowledge Pattern.1\SparInterimUDFs`.GetItem(e)

sparinterimmwpudf =
CreateOrModifyTemplate("CatalogDocument2|UserFeature1",`Instantiated Geometry`
,`Relations\Knowledge Pattern.1\SparInterimmwpUDFs`    ,g)
sparinterimmwpudf->SetAttributeObject("Line1",oldudf->GetAttributeObject("Line2") )
sparinterimmwpudf->SetAttributeObject("Sp1", oldsparinterimudf-
>GetAttributeObject("Pointa") )
sparinterimmwpudf->SetAttributeObject("Line2",udf->GetAttributeObject("Line2") )
sparinterimmwpudf->SetAttributeObject("Sp2", oldsparinterimudf-
>GetAttributeObject("Pointb") )
sparinterimmwpudf->SetAttributeObject("PlaneWingPanelLefty",udf-
>GetAttributeObject("PlaneWingPanelLeft")   )
sparinterimmwpudf->SetAttributeObject("PlaneWingPanelRighty",udf-
>GetAttributeObject("PlaneWingPanelRight")   )
sparinterimmwpudf->SetAttributeObject("Multi-sections Surfacewing",udf-
>GetAttributeObject("Multi-sections Surfacewing")    )
sparinterimmwpudf->SetAttributeObject("WingPanelLowerSurface",udf-
>GetAttributeObject("WingPanelLowerSurface")     )

EndModifyTemplate(sparinterimmwpudf)
sparinterimmwpudf.Name = "SparmInterim." +ToString(g)

set interimspara = `Relations\Knowledge Pattern.1\InterimSparSAlist` ->GetItem(j)
set interimspara1 = sparinterimmwpudf->GetAttributeObject("SparSurfaceA")
interimspara = assemble(interimspara,interimspara1)

set interimsparb = `Relations\Knowledge Pattern.1\InterimSparSBlist` ->GetItem(j)
set interimsparb1 = sparinterimmwpudf->GetAttributeObject("SparSurfaceB")
interimsparb = assemble(interimsparb,interimsparb1)


if x == 1
	{
oldsparudf = `Relations\Knowledge Pattern.1\SparUDFs` .GetItem(j)
sparinterimmwpudf->SetAttributeReal("`SparPositionP1`",oldsparudf-
>GetAttributeReal("`SparPositionP2`")  )
	}

else
	{
oldsparinterimudf = `Relations\Knowledge Pattern.1\SparInterimmwpUDFs` .GetItem(e-
1)
sparinterimmwpudf->SetAttributeReal("`SparPositionP1`",oldsparinterimudf-
>GetAttributeReal("`SparPositionP2`")  )
	}

sparjoin = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\SparJoin`  ,vv)
sparinterimmwpudf = `Relations\Knowledge Pattern.1\SparInterimmwpUDFs` ->GetItem(g)
set surface1 = sparinterimmwpudf ->GetAttributeObject("SparSurface")
set surface2 = `Base Geometry\SparsSurface`
sparjoin = assemble(surface1,surface2)
sparjoin.Name = "SparJoinInterim." + ToString(vv)
vv=vv+1
`Base Geometry\SparsSurface`  = sparjoin

e = e+1
g =g+1

	}
```

```
if j>2
       {
oldudf = `Relations\Knowledge Pattern.1\UDFs` ->GetItem(x)
udf = `Relations\Knowledge Pattern.1\UDFs`  ->GetItem(x+1)

oldsparinterimudf = `Relations\Knowledge Pattern.1\SparInterimmwpUDFs` .GetItem(s)

sparinterimmwpudf =
CreateOrModifyTemplate("CatalogDocument2|UserFeature1",`Instantiated Geometry`
,`Relations\Knowledge Pattern.1\SparInterimmwpUDFs`    ,g)
sparinterimmwpudf->SetAttributeObject("Line1",oldudf->GetAttributeObject("Line2") )
sparinterimmwpudf->SetAttributeObject("Sp1", oldsparinterimudf-
>GetAttributeObject("Pointa") )
sparinterimmwpudf->SetAttributeObject("Line2",udf->GetAttributeObject("Line2") )
sparinterimmwpudf->SetAttributeObject("Sp2", oldsparinterimudf-
>GetAttributeObject("Pointb") )
sparinterimmwpudf->SetAttributeObject("PlaneWingPanelLefty",udf-
>GetAttributeObject("PlaneWingPanelLeft")   )
sparinterimmwpudf->SetAttributeObject("PlaneWingPanelRighty",udf-
>GetAttributeObject("PlaneWingPanelRight")   )
sparinterimmwpudf->SetAttributeObject("Multi-sections Surfacewing",udf-
>GetAttributeObject("Multi-sections Surfacewing")    )
sparinterimmwpudf->SetAttributeObject("WingPanelLowerSurface",udf-
>GetAttributeObject("WingPanelLowerSurface")     )

EndModifyTemplate(sparinterimmwpudf)
sparinterimmwpudf.Name = "SparmInterim." +  ToString(g)

set interimspara = `Relations\Knowledge Pattern.1\InterimSparSAlist` ->GetItem(j)
set interimspara1 = sparinterimmwpudf->GetAttributeObject("SparSurfaceA")
interimspara = assemble(interimspara,interimspara1)

set interimsparb = `Relations\Knowledge Pattern.1\InterimSparSBlist` ->GetItem(j)
set interimsparb1 = sparinterimmwpudf->GetAttributeObject("SparSurfaceB")
interimsparb = assemble(interimsparb,interimsparb1)


if x == 1
       {
oldsparudf = `Relations\Knowledge Pattern.1\SparUDFs` .GetItem(j)
sparinterimmwpudf->SetAttributeReal("`SparPositionP1`",oldsparudf-
>GetAttributeReal("`SparPositionP2`")  )
       }

if x>1
       {
oldsparinterimudf = `Relations\Knowledge Pattern.1\SparInterimmwpUDFs` .GetItem(g-
1)
sparinterimmwpudf->SetAttributeReal("`SparPositionP1`",oldsparinterimudf-
>GetAttributeReal("`SparPositionP2`")  )
       }

sparjoin = CreateOrModifyDatum("Surface",`Base Geometry` ,`Relations\Knowledge
Pattern.1\SparJoin`  ,vv)
sparinterimmwpudf = `Relations\Knowledge Pattern.1\SparInterimmwpUDFs` ->GetItem(g)
set surface1 = sparinterimmwpudf ->GetAttributeObject("SparSurface")
set surface2 = `Base Geometry\SparsSurface`
sparjoin = assemble(surface1,surface2)
sparjoin.Name = "SparJoinInterim." + ToString(vv)
vv=vv+1
`Base Geometry\SparsSurface`  = sparjoin

s = s+1
g =g+1

                              }
                       }
```

```
            }
      }
}


/********/
/* Ribs */
/********/

h =1
k = 1
countspar = 1
countsparmr = 1

lastspar =`New Number of Spars`

For k while k<=`New Number of Ribs`
{
if k==1
      {
ribudf = CreateOrModifyTemplate("CatalogDocument3|UserFeature1",`Instantiated
Geometry` ,`Relations\Knowledge Pattern.1\RibUDFs` ,k)
ribudf->SetAttributeObject("Line1",`Base Geometry\LineforRibsFront` )
ribudf->SetAttributeObject("Point1",`Base Geometry\RibPolylineFrontA` )
ribudf->SetAttributeObject("Line2",`Base Geometry\LineforRibsAft`  )
ribudf->SetAttributeObject("Point2",`Base Geometry\RibPolylineBackA` )
ribudf->SetAttributeObject("Multi-sections Surfacewing", `Base
Geometry\MeshSurfaceWing`  )
ribudf->SetAttributeObject("WingPanelLowerSurface", `Temporary Geometry\WPLS`    )

EndModifyTemplate(ribudf)
ribudf.Name = "Rib." + ToString(k)

ribudf = `Relations\Knowledge Pattern.1\RibUDFs`  ->GetItem(k)
set surface1 = ribudf->GetAttributeObject("RibSurface")
`Base Geometry\RibsSurface`   = surface1

For countspar while countspar <=`New Number of Spars`
      {
if countspar == 1
      {
surface3 = `Relations\Knowledge Pattern.1\InterimSparSAlist`  ->GetItem(1)

ribdivision = CreateOrModifyTemplate("CatalogDocument4|UserFeature1", `Instantiated
Geometry` ,`Relations\Knowledge Pattern.1\RibDivisionList` ,ribdivisionnr)
ribdivision->SetAttributeObject("Line1",`Base Geometry\LineforRibsFront` )
ribdivision->SetAttributeObject("Point1",`Base Geometry\RibPolylineFrontA`  )
ribdivision->SetAttributeObject("Line2",`Base Geometry\LineforRibsAft`  )
ribdivision->SetAttributeObject("Point2",`Base Geometry\RibPolylineBackA`  )
ribdivision->SetAttributeObject("WPLS", `Temporary Geometry\WPLS`  )
ribdivision->SetAttributeObject("SparsSurface", surface3   )

EndModifyTemplate(ribdivision)
ribdivision.Name = "RibDivision." + ToString(ribdivisionnr)
ribdivisionnr = ribdivisionnr+1

ribdivision->SetAttributeReal("`Rib Position P1`", ribudf->GetAttributeReal("`Rib
Position P1`"))
ribdivision->SetAttributeReal("`Rib Position P2`", ribudf->GetAttributeReal("`Rib
Position P2`"))
ribdivision->SetAttributeReal("`Rib Thickness`", ribudf->GetAttributeReal("`Rib
Thickness`"))

      }

if countspar > 1 and countspar <`New Number of Spars`
      {
surface5 = `Relations\Knowledge Pattern.1\InterimSparSAlist`   ->GetItem(countspar)
```

```
surface6 = `Relations\Knowledge Pattern.1\InterimSparSBlist` ->GetItem(countspar-1)

ribdivision = CreateOrModifyTemplate("CatalogDocument6|UserFeature1", `Instantiated
Geometry`  ,`Relations\Knowledge Pattern.1\RibDivisionList`  ,ribdivisionnr)
ribdivision->SetAttributeObject("Line1",`Base Geometry\LineforRibsFront` )
ribdivision->SetAttributeObject("Point1",`Base Geometry\RibPolylineFrontA`  )
ribdivision->SetAttributeObject("Line2",`Base Geometry\LineforRibsAft`  )
ribdivision->SetAttributeObject("Point2",`Base Geometry\RibPolylineBackA`  )
ribdivision->SetAttributeObject("WPLS", `Temporary Geometry\WPLS`  )
ribdivision->SetAttributeObject("SparSurfaceB", surface6   )
ribdivision->SetAttributeObject("SparSurfaceA", surface5   )

EndModifyTemplate(ribdivision)
ribdivision.Name = "RibDivision." + ToString(ribdivisionnr)
ribdivisionnr = ribdivisionnr+1

ribdivision->SetAttributeReal("`Rib Position P1`", ribudf->GetAttributeReal("`Rib
Position P1`"))
ribdivision->SetAttributeReal("`Rib Position P2`", ribudf->GetAttributeReal("`Rib
Position P2`"))
ribdivision->SetAttributeReal("`Rib Thickness`", ribudf->GetAttributeReal("`Rib
Thickness`"))

       }

if countspar ==`New Number of Spars`
       {
if `New Number of Spars` >1
       {
surface5 = `Relations\Knowledge Pattern.1\InterimSparSAlist`   ->GetItem(countspar)
surface6 = `Relations\Knowledge Pattern.1\InterimSparSBlist` ->GetItem(countspar-1)

ribdivision = CreateOrModifyTemplate("CatalogDocument6|UserFeature1", `Instantiated
Geometry`  ,`Relations\Knowledge Pattern.1\RibDivisionList`  ,ribdivisionnr)
ribdivision->SetAttributeObject("Line1",`Base Geometry\LineforRibsFront` )
ribdivision->SetAttributeObject("Point1",`Base Geometry\RibPolylineFrontA`  )
ribdivision->SetAttributeObject("Line2",`Base Geometry\LineforRibsAft`  )
ribdivision->SetAttributeObject("Point2",`Base Geometry\RibPolylineBackA`  )
ribdivision->SetAttributeObject("WPLS", `Temporary Geometry\WPLS`  )
ribdivision->SetAttributeObject("SparSurfaceB", surface6   )
ribdivision->SetAttributeObject("SparSurfaceA", surface5   )

EndModifyTemplate(ribdivision)
ribdivision.Name = "RibDivision." + ToString(ribdivisionnr)
ribdivisionnr = ribdivisionnr+1

ribdivision->SetAttributeReal("`Rib Position P1`", ribudf->GetAttributeReal("`Rib
Position P1`"))
ribdivision->SetAttributeReal("`Rib Position P2`", ribudf->GetAttributeReal("`Rib
Position P2`"))
ribdivision->SetAttributeReal("`Rib Thickness`", ribudf->GetAttributeReal("`Rib
Thickness`"))

       }

surface3 = `Relations\Knowledge Pattern.1\InterimSparSBlist`   ->GetItem(lastspar)
ribdivision = CreateOrModifyTemplate("CatalogDocument5|UserFeature1", `Instantiated
Geometry` ,`Relations\Knowledge Pattern.1\RibDivisionList`  ,ribdivisionnr)
ribdivision->SetAttributeObject("Line1",`Base Geometry\LineforRibsFront` )
ribdivision->SetAttributeObject("Point1",`Base Geometry\RibPolylineFrontA`  )
ribdivision->SetAttributeObject("Line2",`Base Geometry\LineforRibsAft`  )
ribdivision->SetAttributeObject("Point2",`Base Geometry\RibPolylineBackA`  )
ribdivision->SetAttributeObject("WPLS", `Temporary Geometry\WPLS`  )
ribdivision->SetAttributeObject("SparsSurface", surface3   )

EndModifyTemplate(ribdivision)
ribdivision.Name = "RibDivision." + ToString(ribdivisionnr)
ribdivisionnr = ribdivisionnr+1
```

```
ribdivision->SetAttributeReal("`Rib Position P1`", ribudf->GetAttributeReal("`Rib
Position P1`"))
ribdivision->SetAttributeReal("`Rib Position P2`", ribudf->GetAttributeReal("`Rib
Position P2`"))
ribdivision->SetAttributeReal("`Rib Thickness`", ribudf->GetAttributeReal("`Rib
Thickness`"))

                    }
              }

       }


if k>1
       {
countsparmr = 1
num =`Relations\Knowledge Pattern.1\RibDivisionList` ->Size()

oldribudf = `Relations\Knowledge Pattern.1\RibUDFs` ->GetItem(k-1)
ribudf = CreateOrModifyTemplate("CatalogDocument3|UserFeature1",`Instantiated
Geometry`,`Relations\Knowledge Pattern.1\RibUDFs`  ,k)
ribudf->SetAttributeObject("Line1",`Base Geometry\LineforRibsFront` )
ribudf->SetAttributeObject("Point1",oldribudf->GetAttributeObject("Pointa") )
ribudf->SetAttributeObject("Line2",`Base Geometry\LineforRibsAft`  )
ribudf->SetAttributeObject("Point2",oldribudf->GetAttributeObject("Pointb") )
ribudf->SetAttributeObject("Multi-sections Surfacewing", `Base
Geometry\MeshSurfaceWing`  )
ribudf->SetAttributeObject("WingPanelLowerSurface", `Temporary Geometry\WPLS`    )

EndModifyTemplate(ribudf)
ribudf.Name = "Rib." + ToString(k)

For countsparmr while countsparmr<=`New Number of Spars`
       {
if countsparmr == 1
       {
surface3 = `Relations\Knowledge Pattern.1\InterimSparSAlist`  ->GetItem(1)

ribdivision = CreateOrModifyTemplate("CatalogDocument4|UserFeature1", `Instantiated
Geometry` ,`Relations\Knowledge Pattern.1\RibDivisionList`  ,num+1)
ribdivision->SetAttributeObject("Line1",`Base Geometry\LineforRibsFront` )
ribdivision->SetAttributeObject("Point1",oldribudf->GetAttributeObject("Pointa"))
ribdivision->SetAttributeObject("Line2",`Base Geometry\LineforRibsAft`  )
ribdivision->SetAttributeObject("Point2",oldribudf->GetAttributeObject("Pointb"))
ribdivision->SetAttributeObject("WPLS", `Temporary Geometry\WPLS`  )
ribdivision->SetAttributeObject("SparsSurface", surface3   )

EndModifyTemplate(ribdivision)
ribdivision.Name = "RibDivision." + ToString(num+1)
num = num+1

ribdivision->SetAttributeReal("`Rib Position P1`", ribudf->GetAttributeReal("`Rib
Position P1`"))
ribdivision->SetAttributeReal("`Rib Position P2`", ribudf->GetAttributeReal("`Rib
Position P2`"))
ribdivision->SetAttributeReal("`Rib Thickness`", ribudf->GetAttributeReal("`Rib
Thickness`"))

       }
if countsparmr >1 and countsparmr <`New Number of Spars`
       {
surface5 = `Relations\Knowledge Pattern.1\InterimSparSAlist`   -
>GetItem(countsparmr)
surface6 = `Relations\Knowledge Pattern.1\InterimSparSBlist` ->GetItem(countsparmr-
1)
```

```
ribdivision = CreateOrModifyTemplate("CatalogDocument6|UserFeature1", `Instantiated
Geometry`  ,`Relations\Knowledge Pattern.1\RibDivisionList`  ,num+1)
ribdivision->SetAttributeObject("Line1",`Base Geometry\LineforRibsFront` )
ribdivision->SetAttributeObject("Point1",oldribudf->GetAttributeObject("Pointa"))
ribdivision->SetAttributeObject("Line2",`Base Geometry\LineforRibsAft`  )
ribdivision->SetAttributeObject("Point2",oldribudf->GetAttributeObject("Pointb"))
ribdivision->SetAttributeObject("WPLS", `Temporary Geometry\WPLS`  )
ribdivision->SetAttributeObject("SparSurfaceB", surface6   )
ribdivision->SetAttributeObject("SparSurfaceA", surface5   )

EndModifyTemplate(ribdivision)
ribdivision.Name = "RibDivision." + ToString(num+1)
num = num+1

ribdivision->SetAttributeReal("`Rib Position P1`", ribudf->GetAttributeReal("`Rib
Position P1`"))
ribdivision->SetAttributeReal("`Rib Position P2`", ribudf->GetAttributeReal("`Rib
Position P2`"))
ribdivision->SetAttributeReal("`Rib Thickness`", ribudf->GetAttributeReal("`Rib
Thickness`"))

       }

if countsparmr ==`New Number of Spars`
       {
if `New Number of Spars` >1
       {
surface5 = `Relations\Knowledge Pattern.1\InterimSparSAlist`   -
>GetItem(countsparmr)
surface6 = `Relations\Knowledge Pattern.1\InterimSparSBlist` ->GetItem(countsparmr-
1)

ribdivision = CreateOrModifyTemplate("CatalogDocument6|UserFeature1", `Instantiated
Geometry`  ,`Relations\Knowledge Pattern.1\RibDivisionList`  ,num+1)
ribdivision->SetAttributeObject("Line1",`Base Geometry\LineforRibsFront` )
ribdivision->SetAttributeObject("Point1",oldribudf->GetAttributeObject("Pointa"))
ribdivision->SetAttributeObject("Line2",`Base Geometry\LineforRibsAft`  )
ribdivision->SetAttributeObject("Point2",oldribudf->GetAttributeObject("Pointb"))
ribdivision->SetAttributeObject("WPLS", `Temporary Geometry\WPLS`  )
ribdivision->SetAttributeObject("SparSurfaceB", surface6   )
ribdivision->SetAttributeObject("SparSurfaceA", surface5   )

EndModifyTemplate(ribdivision)
ribdivision.Name = "RibDivision." + ToString(num+1)
num = num+1

ribdivision->SetAttributeReal("`Rib Position P1`", ribudf->GetAttributeReal("`Rib
Position P1`"))
ribdivision->SetAttributeReal("`Rib Position P2`", ribudf->GetAttributeReal("`Rib
Position P2`"))
ribdivision->SetAttributeReal("`Rib Thickness`", ribudf->GetAttributeReal("`Rib
Thickness`"))

       }

surface3 = `Relations\Knowledge Pattern.1\InterimSparSBlist`   ->GetItem(lastspar)
ribdivision = CreateOrModifyTemplate("CatalogDocument5|UserFeature1", `Instantiated
Geometry` ,`Relations\Knowledge Pattern.1\RibDivisionList`  ,num+1)
ribdivision->SetAttributeObject("Line1",`Base Geometry\LineforRibsFront` )
ribdivision->SetAttributeObject("Point1",oldribudf->GetAttributeObject("Pointa"))
ribdivision->SetAttributeObject("Line2",`Base Geometry\LineforRibsAft`  )
ribdivision->SetAttributeObject("Point2",oldribudf->GetAttributeObject("Pointb"))
ribdivision->SetAttributeObject("WPLS", `Temporary Geometry\WPLS`  )
ribdivision->SetAttributeObject("SparsSurface", surface3   )

EndModifyTemplate(ribdivision)
ribdivision.Name = "RibDivision." + ToString(num+1)
num = num+1
```

```
ribdivision->SetAttributeReal("`Rib Position P1`", ribudf->GetAttributeReal("`Rib
Position P1`"))
ribdivision->SetAttributeReal("`Rib Position P2`", ribudf->GetAttributeReal("`Rib
Position P2`"))
ribdivision->SetAttributeReal("`Rib Thickness`", ribudf->GetAttributeReal("`Rib
Thickness`"))

                    }
              }

       }

}

`Trigger` =triggerv
`Old Number of Wing Panels` =`New Number of Wing Panels
```

16.2 PowerCopy with VBA Scripting Generic Wing Model Code

16.2.1 Wing Panels Code

```
Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Item("Wing1.CATPart")
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies
Set parameters1 = part1.Parameters
Set relations1 = part1.Relations

Set selection1 = partDocument1.Selection
selection1.Clear

Set old_nr_ins = parameters1.Item("Old Number of WingPanels")
Set new_nr_ins = parameters1.Item("New Number of WingPanels")
Set old_spars = parameters1.Item("Old Number of Spars")
Set new_spars = parameters1.Item("New Number of Spars")
Set old_ribs = parameters1.Item("Old Number of Ribs")
Set new_ribs = parameters1.Item("New Number of Ribs")
Set nr = parameters1.Item("Number")
Set trigger_spars = parameters1.Item("TriggerSpars")
Set trigger_ribs = parameters1.Item("TriggerRibs")


If old_nr_ins.Value < new_nr_ins.Value Then

For I_nr = old_nr_ins.Value + 1 To new_nr_ins.Value
Set hybridBody1 = hybridBodies1.Add()
hybridBody1.Name = "WingPanel." & I_nr

' Code added for the Powercopy
Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Item("Wing1.CATPart")
Set PartDest = partDocument1.Part
Set parameters1 = PartDest.Parameters
Set relations1 = PartDest.Relations
Set Factory = PartDest.GetCustomerFactory("InstanceFactory")
Factory.BeginInstanceFactory "PowerCopyWingFEM", "D:\Documents and
Settings\Sohaib\Desktop\Thesis Project Files\June\12-July-
2011\Powercopy\WingPanel_PC4.CATPart"
Factory.BeginInstantiate

Set Item_to_set1 = PartDest.FindObjectByName("Plane.1")
Factory.PutInputData "Plane.1", Item_to_set1
Set Item_to_set2 = PartDest.FindObjectByName("Point.1")
Factory.PutInputData "Point.1", Item_to_set2

Set Instance = Factory.Instantiate
Factory.EndInstantiate
Factory.EndInstanceFactory

Set Item_to_set1 = PartDest.FindObjectByName("Plane.1")
Item_to_set1.Name = "Plane"
Set Item_to_set1 = PartDest.FindObjectByName("Point.1")
Item_to_set1.Name = "Point"

Set Item_to_set1 = PartDest.FindObjectByName("Tplane")
Item_to_set1.Name = "Plane.1"
Set Item_to_set1 = PartDest.FindObjectByName("TALE")
Item_to_set1.Name = "Point.1"

PartDest.Update

Set nr = parameters1.Item("Number")
nr.Value = nr.Value + 1
'PartDest.Update
```

```
If nr.Value > 1 Then
Set string1 = parameters1.Item("Root Airfoil." & nr.Value)
Set Formula1 = relations1.CreateFormula("xy", "", string1, "`Tip Airfoil." &
(nr.Value - 1) & "`")

Set angle1 = parameters1.Item("Root Airfoil Rotation wrt z-axis." & nr.Value)
Set Formula1 = relations1.CreateFormula("xy", "", angle1, "`Tip Airfoil Rotation
wrt z-axis." & (nr.Value - 1) & "`")
Set angle1 = parameters1.Item("Root Airfoil Rotation wrt y-axis." & nr.Value)
Set Formula1 = relations1.CreateFormula("xy", "", angle1, "`Tip Airfoil Rotation
wrt y-axis." & (nr.Value - 1) & "`")
Set angle1 = parameters1.Item("Root Airfoil Rotation wrt x-axis." & nr.Value)
Set Formula1 = relations1.CreateFormula("xy", "", angle1, "`Tip Airfoil Rotation
wrt x-axis." & (nr.Value - 1) & "`")

Set Length1 = parameters1.Item("Root Airfoil Chord." & nr.Value)
Set Formula1 = relations1.CreateFormula("xy", "", Length1, "`Tip Airfoil Chord." &
(nr.Value - 1) & "`")

Set real1 = parameters1.Item("Tip Airfoil Rotation Point along chord wrt z-axis." &
nr.Value)
Set Formula1 = relations1.CreateFormula("xy", "", real1, "`Root Airfoil Rotation
Point along chord wrt z-axis." & (nr.Value - 1) & "`")
Set real2 = parameters1.Item("Tip Airfoil Rotation Point along chord wrt y-axis." &
nr.Value)
Set Formula1 = relations1.CreateFormula("xy", "", real2, "`Root Airfoil Rotation
Point along chord wrt y-axis." & (nr.Value - 1) & "`")

End If

' code added for powercopy ends here


' ************************************'
' Code to add new panel to the original join '

Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Item("Wing1.CATPart")
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Wing")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeAssemble1 = hybridShapes1.Item("Wing")

Set hybridBody3 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes3 = hybridBody3.HybridShapes
Set hybridShapeLoft3 = hybridShapes3.Item("Multi-sections Surfacewing")
Set reference3 = part1.CreateReferenceFromObject(hybridShapeLoft3)
hybridShapeAssemble1.AddElement reference3

hybridShapeAssemble1.SetConnex 1
hybridShapeAssemble1.SetManifold 0
hybridShapeAssemble1.SetSimplify 0
hybridShapeAssemble1.SetSuppressMode 0
hybridShapeAssemble1.SetDeviation 0.001
hybridShapeAssemble1.SetAngularToleranceMode 0
hybridShapeAssemble1.SetAngularTolerance 0.5
hybridShapeAssemble1.SetFederationPropagation 0
part1.InWorkObject = hybridShapeAssemble1
part1.Update

' ************************************************'

' add new points to rib line

Dim nr_ref_sparpolyline As Long
```

```
Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridBodies2 = hybridBody1.HybridBodies
Set hybridBody2 = hybridBodies2.Item("PointsforSpar")
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapeProject1 = hybridShapes1.Item("SP3")
Set hybridShapeProject2 = hybridShapes1.Item("SP4")
Set ref1 = part1.CreateReferenceFromObject(hybridShapeProject1)
Set ref2 = part1.CreateReferenceFromObject(hybridShapeProject2)

Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePolyline1 = hybridShapes1.Item("LineforPlacingRibs")
Set hybridShapePolyline2 = hybridShapes1.Item("LineforPlacingRibs2")

'Set ref2= part1.CreateReferenceFromObject(hybridShapePolyline1)

nr_ref_sparpolyline = hybridShapePolyline1.NumberOfElements
nr_ref_sparpolyline = nr_ref_sparpolyline + 1

hybridShapePolyline1.InsertElement ref1, nr_ref_sparpolyline
hybridShapePolyline1.Closure = False
hybridShapePolyline2.InsertElement ref2, nr_ref_sparpolyline
hybridShapePolyline1.Closure = False
part1.Update

' add new points to GuidesDim partDocument1 As Document
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePolyline1 = hybridShapes1.Item("GuideA_WLS")
Set hybridShapePolyline2 = hybridShapes1.Item("GuideB_WLS")

Set hybridBody3 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes3 = hybridBody3.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes3.Item("pLinea-2")
Set hybridShapePointOnCurve2 = hybridShapes3.Item("pLineb-2")
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
Set ref2 = part1.CreateReferenceFromObject(hybridShapePointOnCurve2)

nr_ref_sparpolyline = hybridShapePolyline1.NumberOfElements
nr_ref_sparpolyline = nr_ref_sparpolyline + 1

hybridShapePolyline1.InsertElement ref1, nr_ref_sparpolyline
hybridShapePolyline1.Closure = False

nr_ref_sparpolyline = hybridShapePolyline2.NumberOfElements
nr_ref_sparpolyline = nr_ref_sparpolyline + 1

hybridShapePolyline2.InsertElement ref2, nr_ref_sparpolyline
hybridShapePolyline2.Closure = False

' TEST %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

' add spline to use and extremum to use to the WingLowerSurface
Set hybridBody1 = hybridBodies1.Item("Wing")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeLoft1 = hybridShapes1.Item("WingLowerSurface")

Set hybridBody2 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes2 = hybridBody2.HybridShapes
Set hybridShapeSpline3 = hybridShapes2.Item("SplineToUse")
Set reference5 = part1.CreateReferenceFromObject(hybridShapeSpline3)
Set hybridShapeExtremum3 = hybridShapes2.Item("ExtremumToUse")
Set reference6 = part1.CreateReferenceFromObject(hybridShapeExtremum3)
hybridShapeLoft1.AddSectionToLoft reference5, -1, reference6
part1.Update
Next
' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
'%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
'%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

' SPAR CODE ADDED HERE

' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
' new nr of wing panels is greater than one
' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

If new_nr_ins.Value > 1 Then
For I_nr = old_nr_ins.Value + 1 To new_nr_ins.Value

' Add new points to the first spar
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeLinePtPt1 = hybridShapes1.Item("Line2")
Set reference1 = part1.CreateReferenceFromObject(hybridShapeLinePtPt1)
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointOnCurve1 =
hybridShapeFactory1.AddNewPointOnCurveFromPercent(reference1, 0.25, False)
hybridShapePointOnCurve1.Name = "SparP2" & "1" & I_nr

' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
' if we were to add to the point inside the wing panel hybrid body
Set hybridBody2 = hybridBodies1.Item("Spar.1")
' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

hybridBody2.AppendHybridShape hybridShapePointOnCurve1
part1.InWorkObject = hybridShapePointOnCurve1
part1.Update

' %%%%%%%%%%%%%%%
' Code To hide a geometry start
' %%%%%%%%%%%%%%%

Set selection1 = partDocument1.Selection
Set visPropertySet1 = selection1.VisProperties
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & "1" & I_nr)
Set hybridShapes1 = hybridShapePointOnCurve1.Parent
Dim bSTR1
bSTR1 = hybridShapePointOnCurve1.Name
selection1.Add hybridShapePointOnCurve1
Set visPropertySet1 = visPropertySet1.Parent
Dim bSTR2
bSTR2 = visPropertySet1.Name
Dim bSTR3
bSTR3 = visPropertySet1.Name
visPropertySet1.SetShow 1
selection1.Clear

' %%%%%%%%%%%%%%%
' Code To hide a geometry end
' %%%%%%%%%%%%%%%

' Creates Parameter and Formula for Position of Points
Set spar_pos = parameters1.CreateReal("", 0.25)
spar_pos.Rename "SparPositionP2." & "1" & I_nr
Set realParam2 = parameters1.Item("Wing\Spar." & "1" & "\SparP2" & "1" & I_nr &
"\Ratio")
Set Formula1 = relations1.CreateFormula("sparfP2" & "1" & I_nr, "", realParam2,
"`SparPositionP2." & "1" & I_nr & "` ")
part1.Update
```

```
If I_nr = new_nr_ins.Value Then
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, 5000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & "1" & I_nr)
hybridShapePointCoord1.Name = "SparP2" & "1" & I_nr & "-z"
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePointCoord1.PtRef = reference1
hybridBody1.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
part1.Update

' Delete last polyline point and add new points to polyline created above

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePolyline1 = hybridShapes1.Item("SparPolyline")
Dim nrpoly As Long
nrpoly = hybridShapePolyline1.NumberOfElements
hybridShapePolyline1.RemoveElement nrpoly

Dim nrpoly_new As Long
nrpoly_new = hybridShapePolyline1.NumberOfElements

i = 1

For I_nrx = old_nr_ins.Value + 1 To new_nr_ins.Value
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & "1" & I_nrx)
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement ref1, nrpoly_new + i
i = i + 1
Next
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & "1" & I_nr & "-z")
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement ref1, nrpoly_new + i

' replace the plane in the split spar
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitSpar")

Set hybridBody1 = hybridBodies1.Item("WingPanel." & old_nr_ins.Value)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcut = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcutnew = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in Split Solid spar
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitWithWingPanelPlaneRight")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in SparSurfaceA and SparSurfaceB
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
```

```
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceA")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceB")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

End If

Next


End If

trigger_spars.Value = trigger_spars.Value + 1
'old_nr_ins.Value = new_nr_ins.Value
part1.Update



End If



' ****************************
' Deletion Code of Wing Panels
' ****************************


If old_nr_ins.Value > new_nr_ins.Value Then

' %%%%%%%%%%%%%%%%%%%%%%%%%%
' If there is one spar only
' %%%%%%%%%%%%%%%%%%%%%%%%%%

If new_spars.Value = old_spars.Value And new_spars.Value = 1 Then
Dim wnr As Long
'Dim nrpoly As Long
wnr = old_nr_ins.Value - new_nr_ins.Value

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePolyline1 = hybridShapes1.Item("SparPolyline")

nrpoly = hybridShapePolyline1.NumberOfElements
For I_nr = 1 To wnr + 1
hybridShapePolyline1.RemoveElement nrpoly
nrpoly = nrpoly - 1
Next

nrpoly = hybridShapePolyline1.NumberOfElements

If new_nr_ins.Value = 1 Then
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP211-z")
hybridShapePolyline1.InsertElement hybridShapePointCoord1, nrpoly + 1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitSpar")

Set hybridBody1 = hybridBodies1.Item("WingPanel." & old_nr_ins.Value)
```

```
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcut = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

Set hybridBody2 = hybridBodies1.Item("WingPanel." & "1")
Set hybridShapes2 = hybridBody2.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes2.Item("PlaneWingPanelRight")
Set refcutnew = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in Split Solid spar
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitWithWingPanelPlaneRight")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in SparSurfaceA and SparSurfaceB
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceA")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceB")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

part1.Update

' add geometries to selection and then delete them
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & old_nr_ins.Value & "-
z")
selection1.Add hybridShapePointCoord1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & old_nr_ins.Value)
selection1.Add hybridShapePointCoord1

Set Pi = parameters1.Item("SparPositionP2.1" & old_nr_ins.Value)
selection1.Add Pi

Set F1 = part1.FindObjectByName("sparfP21" & old_nr_ins.Value)
selection1.Add F1

Dim r As Long

r = 0

For I_nr = new_nr_ins.Value + 1 To old_nr_ins.Value - 1
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & wnr - r)
selection1.Add hybridShapePointCoord1

Set Pi = parameters1.Item("SparPositionP2.1" & wnr - r)
selection1.Add Pi
```

```
Set F1 = part1.FindObjectByName("sparfP21" & wnr - r)
selection1.Add F1

r = r + 1
Next

selection1.Delete
selection1.Clear

End If

If new_nr_ins.Value > 1 Then
' create a new point with a -z

Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, 5000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP21" & new_nr_ins.Value)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePointCoord1.PtRef = reference1
hybridBody1.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
hybridShapePointCoord1.Name = "SparP21" & new_nr_ins.Value & "-z"
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & new_nr_ins.Value & "-
z")
hybridShapePolyline1.InsertElement hybridShapePointCoord1, nrpoly + 1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitSpar")

Set hybridBody1 = hybridBodies1.Item("WingPanel." & old_nr_ins.Value)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcut = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

Set hybridBody2 = hybridBodies1.Item("WingPanel." & new_nr_ins.Value)
Set hybridShapes2 = hybridBody2.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes2.Item("PlaneWingPanelRight")
Set refcutnew = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in Split Solid spar
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitWithWingPanelPlaneRight")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in SparSurfaceA and SparSurfaceB
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceA")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceB")
```

```
hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

part1.Update

' add geometries to selection and then delete them
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & old_nr_ins.Value & "-
z")
selection1.Add hybridShapePointCoord1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & old_nr_ins.Value)
selection1.Add hybridShapePointCoord1

Set Pi = parameters1.Item("SparPositionP2.1" & old_nr_ins.Value)
selection1.Add Pi

Set F1 = part1.FindObjectByName("sparfP21" & old_nr_ins.Value)
selection1.Add F1

Dim v As Long
v = 0

For I_nr = new_nr_ins.Value + 1 To old_nr_ins.Value - 1
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & old_nr_ins.Value - 1 +
v)
selection1.Add hybridShapePointCoord1

Set Pi = parameters1.Item("SparPositionP2.1" & old_nr_ins.Value - 1 + v)
selection1.Add Pi

Set F1 = part1.FindObjectByName("sparfP21" & old_nr_ins.Value - 1 + v)
selection1.Add F1


v = v - 1

Next

selection1.Delete
selection1.Clear

End If

For I_nr = new_nr_ins.Value + 1 To old_nr_ins.Value

Set hybridBody1 = hybridBodies1.Item("Wing")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeLoft1 = hybridShapes1.Item("WingLowerSurface")

Set hybridBody1 = hybridBodies1.Item("WingPanel." & nr.Value)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeExtremum1 = hybridShapes1.Item("ExtremumToUse")
Set ref = part1.CreateReferenceFromObject(hybridShapeExtremum1)

Set hybridShapeExtremum2 = hybridShapes1.Item("SplineToUse")
Set ref1 = part1.CreateReferenceFromObject(hybridShapeExtremum2)
hybridShapeLoft1.RemoveSection ref1
part1.Update

Set F1 = part1.FindObjectByName("Formulaa." & nr.Value)
selection1.Add F1
```

```
Set F2 = part1.FindObjectByName("Formulab." & nr.Value)
selection1.Add F2
Set F3 = part1.FindObjectByName("Formulac." & nr.Value)
selection1.Add F3
Set F4 = part1.FindObjectByName("Formulad." & nr.Value)
selection1.Add F4
Set F5 = part1.FindObjectByName("Formulae." & nr.Value)
selection1.Add F5
Set F6 = part1.FindObjectByName("Formulaf." & nr.Value)
selection1.Add F6
Set F7 = part1.FindObjectByName("Formulag." & nr.Value)
selection1.Add F7
Set F8 = part1.FindObjectByName("Formulah." & nr.Value)
selection1.Add F8
Set F9 = part1.FindObjectByName("Formulai." & nr.Value)
selection1.Add F9
Set F10 = part1.FindObjectByName("Formulaj." & nr.Value)
selection1.Add F10
Set F11 = part1.FindObjectByName("Formulak." & nr.Value)
selection1.Add F11
Set F12 = part1.FindObjectByName("Formulal." & nr.Value)
selection1.Add F12
Set F13 = part1.FindObjectByName("Formulam." & nr.Value)
selection1.Add F13
Set F14 = part1.FindObjectByName("Formulan." & nr.Value)
selection1.Add F14
Set F15 = part1.FindObjectByName("Formulao." & nr.Value)
selection1.Add F15
Set F16 = part1.FindObjectByName("Formulap." & nr.Value)
selection1.Add F16
Set F17 = part1.FindObjectByName("Formulaq." & nr.Value)
selection1.Add F17
Set F18 = part1.FindObjectByName("Formular." & nr.Value)
selection1.Add F18
Set F19 = part1.FindObjectByName("Formulas." & nr.Value)
selection1.Add F19
Set F20 = part1.FindObjectByName("Formulat." & nr.Value)
selection1.Add F20
Set F21 = part1.FindObjectByName("Formulau." & nr.Value)
selection1.Add F21
Set F22 = part1.FindObjectByName("Formulav." & nr.Value)
selection1.Add F22
Set F23 = part1.FindObjectByName("Formulaw." & nr.Value)
selection1.Add F23
Set F24 = part1.FindObjectByName("Formulax." & nr.Value)
selection1.Add F24
Set F25 = part1.FindObjectByName("Formulay." & nr.Value)
selection1.Add F25
Set F26 = part1.FindObjectByName("Formulaz." & nr.Value)
selection1.Add F26
Set F27 = part1.FindObjectByName("Formulaaa." & nr.Value)
selection1.Add F27
Set F28 = part1.FindObjectByName("Formulaab." & nr.Value)
selection1.Add F28
Set F29 = part1.FindObjectByName("Formulaac." & nr.Value)
selection1.Add F29
Set R1 = relations1.Item("Root Airfoil Rule." & nr.Value)
selection1.Add R1
Set P1 = parameters1.Item("Root Airfoil." & nr.Value)
selection1.Add P1
Set P2 = parameters1.Item("Tip Airfoil." & nr.Value)
selection1.Add P2
Set P3 = parameters1.Item("WingPanel Span." & nr.Value)
selection1.Add P3
Set P4 = parameters1.Item("Root Airfoil Chord." & nr.Value)
selection1.Add P4
Set P5 = parameters1.Item("Tip Airfoil Chord." & nr.Value)
selection1.Add P5
```

```
Set P6 = parameters1.Item("WingPanel Leading Edge Sweep Angle." & nr.Value)
selection1.Add P6
Set P7 = parameters1.Item("Dihedral Angle." & nr.Value)
selection1.Add P7
Set P8 = parameters1.Item("Root Airfoil Rotation Point along chord wrt z-axis." &
nr.Value)
selection1.Add P8
Set P9 = parameters1.Item("Tip Airfoil Rotation Point along chord wrt z-axis." &
nr.Value)
selection1.Add P9
Set P10 = parameters1.Item("Root Airfoil Rotation Point along chord wrt y-axis." &
nr.Value)
selection1.Add P10
Set P11 = parameters1.Item("Tip Airfoil Rotation Point along chord wrt y-axis." &
nr.Value)
selection1.Add P11
Set P12 = parameters1.Item("Root Airfoil Rotation wrt z-axis." & nr.Value)
selection1.Add P12
Set P13 = parameters1.Item("Tip Airfoil Rotation wrt z-axis." & nr.Value)
selection1.Add P13
Set P14 = parameters1.Item("Root Airfoil Rotation wrt x-axis." & nr.Value)
selection1.Add P14
Set P15 = parameters1.Item("Tip Airfoil Rotation wrt x-axis." & nr.Value)
selection1.Add P15
Set P16 = parameters1.Item("Root Airfoil Rotation wrt y-axis." & nr.Value)
selection1.Add P16
Set P17 = parameters1.Item("Tip Airfoil Rotation wrt y-axis." & nr.Value)
selection1.Add P17
Set P18 = parameters1.Item("WingPanel Area." & nr.Value)
selection1.Add P18
Set P19 = parameters1.Item("Taper Ratio." & nr.Value)
selection1.Add P19
Set P20 = parameters1.Item("Mean Aerodynamic Chord." & nr.Value)
selection1.Add P20
Set P21 = parameters1.Item("x_MAC." & nr.Value)
selection1.Add P21
Set P22 = parameters1.Item("y_MAC." & nr.Value)
selection1.Add P22
Set P23 = parameters1.Item("Aspect Ratio." & nr.Value)
selection1.Add P23
Set P24 = parameters1.Item("Skin Thickness." & nr.Value)
selection1.Add P24
Set hybridBody1 = hybridBodies1.Item("WingPanel." & nr.Value)
selection1.Add hybridBody1
On Error Resume Next
selection1.Delete
On Error Resume Next
selection1.Clear
On Error Resume Next
nr.Value = nr.Value - 1

Next

' Avoids problem with changing of the name
If new_nr_ins.Value >= 1 Then
Set hybridBody1 = hybridBodies1.Item("WingPanel." & new_nr_ins.Value)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlaneOffset1 = hybridShapes1.Item("Plane")
hybridShapePlaneOffset1.Name = "Plane.1"

Set hybridBody1 = hybridBodies1.Item("WingPanel." & new_nr_ins.Value)
Set hybridShapePointOnPlane1 = hybridShapes1.Item("Point")
hybridShapePointOnPlane1.Name = "Point.1"
End If
End If
```

```
' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
' If there are more than one spar or more than one ribs
' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

If new_spars.Value = old_spars.Value And new_spars.Value > 1 Or new_ribs.Value =
new_ribs.Value And new_ribs.Value > 1 Then
Dim nrspar As Long
Dim nrribs As Long
Dim oldnrwp As Long
Dim newnrwp As Long

oldnrwp = old_nr_ins.Value
newnrwp = new_nr_ins.Value

nrspar = new_spars.Value
nrribs = new_ribs.Value

new_spars.Value = 1
new_ribs.Value = 1
' Add trigger for new spars here ****************************************
trigger_spars.Value = trigger_spars.Value + 1
' Add trigger for new ribs here ****************************************
trigger_ribs.Value = trigger_ribs.Value + 1

wnr = oldnrwp - newnrwp

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePolyline1 = hybridShapes1.Item("SparPolyline")

nrpoly = hybridShapePolyline1.NumberOfElements
For I_nr = 1 To wnr + 1
hybridShapePolyline1.RemoveElement nrpoly
nrpoly = nrpoly - 1
Next

nrpoly = hybridShapePolyline1.NumberOfElements

If newnrwp = 1 Then
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP211-z")
hybridShapePolyline1.InsertElement hybridShapePointCoord1, nrpoly + 1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitSpar")

Set hybridBody1 = hybridBodies1.Item("WingPanel." & oldnrwp)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcut = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

Set hybridBody2 = hybridBodies1.Item("WingPanel." & "1")
Set hybridShapes2 = hybridBody2.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes2.Item("PlaneWingPanelRight")
Set refcutnew = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in Split Solid spar
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitWithWingPanelPlaneRight")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1
```

```
' replace the plane in SparSurfaceA and SparSurfaceB
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceA")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceB")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

part1.Update

' add geometries to selection and then delete them
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & oldnrwp & "-z")
selection1.Add hybridShapePointCoord1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & oldnrwp)
selection1.Add hybridShapePointCoord1

Set Pi = parameters1.Item("SparPositionP2.1" & oldnrwp)
selection1.Add Pi

Set F1 = part1.FindObjectByName("sparfP21" & oldnrwp)
selection1.Add F1

r = 0

For I_nr = newnrwp + 1 To oldnrwp - 1
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & wnr - r)
selection1.Add hybridShapePointCoord1

Set Pi = parameters1.Item("SparPositionP2.1" & wnr - r)
selection1.Add Pi

Set F1 = part1.FindObjectByName("sparfP21" & wnr - r)
selection1.Add F1

r = r + 1
Next

selection1.Delete
selection1.Clear

End If

If newnrwp > 1 Then
' create a new point with a -z

Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, 5000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP21" & newnrwp)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
```

```
hybridShapePointCoord1.PtRef = reference1
hybridBody1.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
hybridShapePointCoord1.Name = "SparP21" & newnrwp & "-z"
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & newnrwp & "-z")
hybridShapePolyline1.InsertElement hybridShapePointCoord1, nrpoly + 1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitSpar")

Set hybridBody1 = hybridBodies1.Item("WingPanel." & oldnrwp)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcut = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

Set hybridBody2 = hybridBodies1.Item("WingPanel." & newnrwp)
Set hybridShapes2 = hybridBody2.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes2.Item("PlaneWingPanelRight")
Set refcutnew = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in Split Solid spar
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitWithWingPanelPlaneRight")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in SparSurfaceA and SparSurfaceB
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceA")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceB")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

part1.Update

' add geometries to selection and then delete them
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & oldnrwp & "-z")
selection1.Add hybridShapePointCoord1

Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & oldnrwp)
selection1.Add hybridShapePointCoord1

Set Pi = parameters1.Item("SparPositionP2.1" & oldnrwp)
selection1.Add Pi

Set F1 = part1.FindObjectByName("sparfP21" & oldnrwp)
selection1.Add F1
```

```
v = 0

For I_nr = newnrwp + 1 To oldnrwp - 1
Set hybridBody1 = hybridBodies1.Item("Spar.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointCoord1 = hybridShapes1.Item("SparP21" & oldnrwp - 1 + v)
selection1.Add hybridShapePointCoord1

Set Pi = parameters1.Item("SparPositionP2.1" & oldnrwp - 1 + v)
selection1.Add Pi

Set F1 = part1.FindObjectByName("sparfP21" & oldnrwp - 1 + v)
selection1.Add F1


v = v - 1

Next

selection1.Delete
selection1.Clear

End If

For I_nr = newnrwp + 1 To oldnrwp

Set hybridBody1 = hybridBodies1.Item("Wing")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeLoft1 = hybridShapes1.Item("WingLowerSurface")

Set hybridBody1 = hybridBodies1.Item("WingPanel." & nr.Value)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeExtremum1 = hybridShapes1.Item("ExtremumToUse")
Set ref = part1.CreateReferenceFromObject(hybridShapeExtremum1)

Set hybridShapeExtremum2 = hybridShapes1.Item("SplineToUse")
Set ref1 = part1.CreateReferenceFromObject(hybridShapeExtremum2)
hybridShapeLoft1.RemoveSection ref1
part1.Update

Set F1 = part1.FindObjectByName("Formulaa." & nr.Value)
selection1.Add F1
Set F2 = part1.FindObjectByName("Formulab." & nr.Value)
selection1.Add F2
Set F3 = part1.FindObjectByName("Formulac." & nr.Value)
selection1.Add F3
Set F4 = part1.FindObjectByName("Formulad." & nr.Value)
selection1.Add F4
Set F5 = part1.FindObjectByName("Formulae." & nr.Value)
selection1.Add F5
Set F6 = part1.FindObjectByName("Formulaf." & nr.Value)
selection1.Add F6
Set F7 = part1.FindObjectByName("Formulag." & nr.Value)
selection1.Add F7
Set F8 = part1.FindObjectByName("Formulah." & nr.Value)
selection1.Add F8
Set F9 = part1.FindObjectByName("Formulai." & nr.Value)
selection1.Add F9
Set F10 = part1.FindObjectByName("Formulaj." & nr.Value)
selection1.Add F10
Set F11 = part1.FindObjectByName("Formulak." & nr.Value)
selection1.Add F11
Set F12 = part1.FindObjectByName("Formulal." & nr.Value)
selection1.Add F12
Set F13 = part1.FindObjectByName("Formulam." & nr.Value)
selection1.Add F13
Set F14 = part1.FindObjectByName("Formulan." & nr.Value)
selection1.Add F14
```

```
Set F15 = part1.FindObjectByName("Formulao." & nr.Value)
selection1.Add F15
Set F16 = part1.FindObjectByName("Formulap." & nr.Value)
selection1.Add F16
Set F17 = part1.FindObjectByName("Formulaq." & nr.Value)
selection1.Add F17
Set F18 = part1.FindObjectByName("Formular." & nr.Value)
selection1.Add F18
Set F19 = part1.FindObjectByName("Formulas." & nr.Value)
selection1.Add F19
Set F20 = part1.FindObjectByName("Formulat." & nr.Value)
selection1.Add F20
Set F21 = part1.FindObjectByName("Formulau." & nr.Value)
selection1.Add F21
Set F22 = part1.FindObjectByName("Formulav." & nr.Value)
selection1.Add F22
Set F23 = part1.FindObjectByName("Formulaw." & nr.Value)
selection1.Add F23
Set F24 = part1.FindObjectByName("Formulax." & nr.Value)
selection1.Add F24
Set F25 = part1.FindObjectByName("Formulay." & nr.Value)
selection1.Add F25
Set F26 = part1.FindObjectByName("Formulaz." & nr.Value)
selection1.Add F26
Set F27 = part1.FindObjectByName("Formulaaa." & nr.Value)
selection1.Add F27
Set F28 = part1.FindObjectByName("Formulaab." & nr.Value)
selection1.Add F28
Set F29 = part1.FindObjectByName("Formulaac." & nr.Value)
selection1.Add F29
Set R1 = relations1.Item("Root Airfoil Rule." & nr.Value)
selection1.Add R1
Set P1 = parameters1.Item("Root Airfoil." & nr.Value)
selection1.Add P1
Set P2 = parameters1.Item("Tip Airfoil." & nr.Value)
selection1.Add P2
Set P3 = parameters1.Item("WingPanel Span." & nr.Value)
selection1.Add P3
Set P4 = parameters1.Item("Root Airfoil Chord." & nr.Value)
selection1.Add P4
Set P5 = parameters1.Item("Tip Airfoil Chord." & nr.Value)
selection1.Add P5
Set P6 = parameters1.Item("WingPanel Leading Edge Sweep Angle." & nr.Value)
selection1.Add P6
Set P7 = parameters1.Item("Dihedral Angle." & nr.Value)
selection1.Add P7
Set P8 = parameters1.Item("Root Airfoil Rotation Point along chord wrt z-axis." &
nr.Value)
selection1.Add P8
Set P9 = parameters1.Item("Tip Airfoil Rotation Point along chord wrt z-axis." &
nr.Value)
selection1.Add P9
Set P10 = parameters1.Item("Root Airfoil Rotation Point along chord wrt y-axis." &
nr.Value)
selection1.Add P10
Set P11 = parameters1.Item("Tip Airfoil Rotation Point along chord wrt y-axis." &
nr.Value)
selection1.Add P11
Set P12 = parameters1.Item("Root Airfoil Rotation wrt z-axis." & nr.Value)
selection1.Add P12
Set P13 = parameters1.Item("Tip Airfoil Rotation wrt z-axis." & nr.Value)
selection1.Add P13
Set P14 = parameters1.Item("Root Airfoil Rotation wrt x-axis." & nr.Value)
selection1.Add P14
Set P15 = parameters1.Item("Tip Airfoil Rotation wrt x-axis." & nr.Value)
selection1.Add P15
Set P16 = parameters1.Item("Root Airfoil Rotation wrt y-axis." & nr.Value)
selection1.Add P16
```

```
Set P17 = parameters1.Item("Tip Airfoil Rotation wrt y-axis." & nr.Value)
selection1.Add P17
Set P18 = parameters1.Item("WingPanel Area." & nr.Value)
selection1.Add P18
Set P19 = parameters1.Item("Taper Ratio." & nr.Value)
selection1.Add P19
Set P20 = parameters1.Item("Mean Aerodynamic Chord." & nr.Value)
selection1.Add P20
Set P21 = parameters1.Item("x_MAC." & nr.Value)
selection1.Add P21
Set P22 = parameters1.Item("y_MAC." & nr.Value)
selection1.Add P22
Set P23 = parameters1.Item("Aspect Ratio." & nr.Value)
selection1.Add P23
Set P24 = parameters1.Item("Skin Thickness." & nr.Value)
selection1.Add P24
Set hybridBody1 = hybridBodies1.Item("WingPanel." & nr.Value)
selection1.Add hybridBody1
On Error Resume Next
selection1.Delete
On Error Resume Next
selection1.Clear
On Error Resume Next
nr.Value = nr.Value - 1

Next

' Avoids problem with changing of the name
If newnrwp >= 1 Then
Set hybridBody1 = hybridBodies1.Item("WingPanel." & newnrwp)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlaneOffset1 = hybridShapes1.Item("Plane")
hybridShapePlaneOffset1.Name = "Plane.1"

Set hybridBody1 = hybridBodies1.Item("WingPanel." & newnrwp)
Set hybridShapePointOnPlane1 = hybridShapes1.Item("Point")
hybridShapePointOnPlane1.Name = "Point.1"
End If

new_spars.Value = nrspar
new_ribs.Value = nrribs
' Add trigger for new spars here ****************************************
trigger_spars.Value = trigger_spars.Value + 1
' Add trigger for new ribs here ****************************************
trigger_ribs.Value = trigger_ribs.Value + 1


End If
End If
old_nr_ins.Value = new_nr_ins.Value

End Sub
```

16.2.2 Wing Spars Code

```
Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Item("Wing1.CATPart")
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies
Set parameters1 = part1.Parameters
Set relations1 = part1.Relations

Set new_nr_ins = parameters1.Item("New Number of WingPanels")
Set old_nr_ins = parameters1.Item("Old Number of WingPanels")

Set new_spars = parameters1.Item("New Number of Spars")
Set old_spars = parameters1.Item("Old Number of Spars")

Set new_ribs = parameters1.Item("New Number of Ribs")
Set old_ribs = parameters1.Item("Old Number of Ribs")

Set trigger_spars = parameters1.Item("TriggerSpars")
Set trigger_ribs = parameters1.Item("TriggerRibs")

Set selection1 = partDocument1.Selection
selection1.Clear

If old_spars.Value < new_spars.Value Then

' ****************************** '
' If there is only one wing panel
' ****************************** '

If old_nr_ins.Value = 1 And new_nr_ins.Value = 1 Then
I_nr = 1
For NRS = old_spars.Value + 1 To new_spars.Value

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeLine1 = hybridShapes1.Item("Line1")
Set hybridShapeLine2 = hybridShapes1.Item("Line2")

' Create Points on Line 1 and Line 2 for the Wing Panel
' Line1

' Create some references to create the new point between the old point and SP2
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP1" & NRS - 1 & I_nr)
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridBodies2 = hybridBody1.HybridBodies
Set hybridBody2 = hybridBodies2.Item("PointsforSpar")
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapeProject1 = hybridShapes1.Item("SP2")
Set ref2 = part1.CreateReferenceFromObject(hybridShapeProject1)
'end of references
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointBetween1 = hybridShapeFactory1.AddNewPointBetween(ref1, ref2,
0.25, 1)
hybridBody1.AppendHybridShape hybridShapePointBetween1
part1.InWorkObject = hybridShapePointBetween1
hybridShapePointBetween1.Name = "SparP1" & NRS & I_nr
part1.Update

' %%%%%%%%%%%%%%%
' Code To hide a geometry start
' %%%%%%%%%%%%%%%

Set selection1 = partDocument1.Selection
Set visPropertySet1 = selection1.VisProperties
```

```
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP1" & NRS & I_nr)
Set hybridShapes1 = hybridShapePointOnCurve1.Parent
Dim bSTR1
bSTR1 = hybridShapePointOnCurve1.Name
selection1.Add hybridShapePointOnCurve1
Set visPropertySet1 = visPropertySet1.Parent
Dim bSTR2
bSTR2 = visPropertySet1.Name
Dim bSTR3
bSTR3 = visPropertySet1.Name
visPropertySet1.SetShow 1
selection1.Clear

' %%%%%%%%%%%%%%%
' Code To hide a geometry end
' %%%%%%%%%%%%%%%

' Line2
' Create some references to create the new point between the old point and SP4
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve3 = hybridShapes1.Item("SparP2" & NRS - 1 & I_nr)
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve3)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridBodies2 = hybridBody1.HybridBodies
Set hybridBody2 = hybridBodies2.Item("PointsforSpar")
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapeProject2 = hybridShapes1.Item("SP4")
Set ref2 = part1.CreateReferenceFromObject(hybridShapeProject2)
'end of references

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointBetween2 = hybridShapeFactory1.AddNewPointBetween(ref1, ref2,
0.25, 1)
hybridBody1.AppendHybridShape hybridShapePointBetween2
part1.InWorkObject = hybridShapePointBetween2

hybridShapePointBetween2.Name = "SparP2" & NRS & I_nr
part1.Update

' %%%%%%%%%%%%%%%
' Code To hide a geometry start
' %%%%%%%%%%%%%%%

Set selection1 = partDocument1.Selection
Set visPropertySet1 = selection1.VisProperties
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr)
Set hybridShapes1 = hybridShapePointOnCurve1.Parent

bSTR1 = hybridShapePointOnCurve1.Name
selection1.Add hybridShapePointOnCurve1
Set visPropertySet1 = visPropertySet1.Parent

bSTR2 = visPropertySet1.Name

bSTR3 = visPropertySet1.Name
visPropertySet1.SetShow 1
selection1.Clear
```

```
' %%%%%%%%%%%%%%%
' Code To hide a geometry end
' %%%%%%%%%%%%%%%

' Creates Parameter and Formula for Position of Points
' Point of Line1
Set spar_pos = parameters1.CreateReal("", 0.25)
spar_pos.Rename "SparPositionP1." & NRS & I_nr
Set realParam2 = parameters1.Item("Wing\WingPanel." & I_nr & "\SparP1" & NRS & I_nr
& "\Ratio")
Set Formula1 = relations1.CreateFormula("sparfP1" & NRS & I_nr, "", realParam2,
"`SparPositionP1." & NRS & I_nr & "` ")
part1.Update

' Point of Line2
Set spar_pos = parameters1.CreateReal("", 0.25)
spar_pos.Rename "SparPositionP2." & NRS & I_nr
Set realParam2 = parameters1.Item("Wing\WingPanel.1" & "\SparP2" & NRS & I_nr &
"\Ratio")
Set Formula1 = relations1.CreateFormula("sparfP2" & NRS & I_nr, "", realParam2,
"`SparPositionP2." & NRS & I_nr & "` ")
part1.Update

Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Add()
hybridBody1.Name = "Spar." & NRS
part1.Update

' add points which are in z direction to points created
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, -10000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointBetween1 = hybridShapes1.Item("SparP1" & NRS & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointBetween1)
hybridShapePointCoord1.PtRef = reference1
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
hybridBody2.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
hybridShapePointCoord1.Name = "SparP1" & NRS & I_nr & "-z"
part1.Update

' %%%%%%%%%%%%%%%
' Code To hide a geometry start
' %%%%%%%%%%%%%%%

Set selection1 = partDocument1.Selection
Set visPropertySet1 = selection1.VisProperties
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP1" & NRS & I_nr & "-z")
Set hybridShapes1 = hybridShapePointOnCurve1.Parent

bSTR1 = hybridShapePointOnCurve1.Name
selection1.Add hybridShapePointOnCurve1
Set visPropertySet1 = visPropertySet1.Parent

bSTR2 = visPropertySet1.Name

bSTR3 = visPropertySet1.Name
visPropertySet1.SetShow 1
selection1.Clear

' %%%%%%%%%%%%%%%
' Code To hide a geometry end
```

```
' %%%%%%%%%%%%%%

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, 10000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointBetween1 = hybridShapes1.Item("SparP2" & NRS & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointBetween1)
hybridShapePointCoord1.PtRef = reference1
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
hybridBody2.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
hybridShapePointCoord1.Name = "SparP2" & NRS & I_nr & "-z"
part1.Update

' %%%%%%%%%%%%%%
' Code To hide a geometry start
' %%%%%%%%%%%%%%

Set selection1 = partDocument1.Selection
Set visPropertySet1 = selection1.VisProperties
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr & "-z")
Set hybridShapes1 = hybridShapePointOnCurve1.Parent

bSTR1 = hybridShapePointOnCurve1.Name
selection1.Add hybridShapePointOnCurve1
Set visPropertySet1 = visPropertySet1.Parent

bSTR2 = visPropertySet1.Name

bSTR3 = visPropertySet1.Name
visPropertySet1.SetShow 1
selection1.Clear

' %%%%%%%%%%%%%%
' Code To hide a geometry end
' %%%%%%%%%%%%%%

Next

' Create Polyline to connect points

For NRS = old_spars.Value + 1 To new_spars.Value

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePolyline1 = hybridShapeFactory1.AddNewPolyline()
Set hybridBodies1 = part1.HybridBodies

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP1" & NRS & I_nr & "-z")
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP1" & NRS & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS + 1

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr)
```

```
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS + 2

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr & "-z")
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS + 3

hybridShapePolyline1.Closure = False
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
hybridBody1.AppendHybridShape hybridShapePolyline1
hybridShapePolyline1.Name = "SparPolyline"
part1.InWorkObject = hybridShapePolyline1
part1.Update

' %%%%%%%%%%%%%%%
' Code To hide a geometry start
' %%%%%%%%%%%%%%%

Set selection1 = partDocument1.Selection
Set visPropertySet1 = selection1.VisProperties
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparPolyline")
Set hybridShapes1 = hybridShapePointOnCurve1.Parent

bSTR1 = hybridShapePointOnCurve1.Name
selection1.Add hybridShapePointOnCurve1
Set visPropertySet1 = visPropertySet1.Parent

bSTR2 = visPropertySet1.Name

bSTR3 = visPropertySet1.Name
visPropertySet1.SetShow 1
selection1.Clear

' %%%%%%%%%%%%%%%
' Code To hide a geometry end
' %%%%%%%%%%%%%%%

Next

Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Item("Wing1.CATPart")
Set part1 = partDocument1.Part
Set parameters1 = part1.Parameters
Set relations1 = part1.Relations

Set new_spars = parameters1.Item("New Number of Spars")
Set old_spars = parameters1.Item("Old Number of Spars")

For NRS = old_spars.Value + 1 To new_spars.Value

Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Wing")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeAssemble1 = hybridShapes1.Item("Wing")

Set hybridBody1 = hybridBodies1.Item("Wing")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeLoft1 = hybridShapes1.Item("WingLowerSurface")

Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
```

```
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points2 = hybridShapes1.Item("PlaneWingPanelLeft")
Set hybridBodies1 = part1.HybridBodies
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapePolyline1 = hybridShapes1.Item("SparPolyline")

part1.InWorkObject = hybridBody2

Set Factory = part1.GetCustomerFactory("InstanceFactory")
Factory.BeginInstanceFactory "PowerCopy", "D:\Documents and
Settings\Sohaib\Desktop\Thesis Project Files\June\12-July-
2011\Powercopy\SparPC.CATPart"
Factory.BeginInstantiate

Set Item_to_set1 = hybridShapePolyline1
Factory.PutInputData "SparPolyline", Item_to_set1
Set Item_to_set2 = hybridShapeAssemble1
Factory.PutInputData "Wing", Item_to_set2
Set Item_to_set3 = hybridShapePlane3Points2
Factory.PutInputData "PlaneWingPanelLeft", Item_to_set3
Set Item_to_set4 = hybridShapePlane3Points1
Factory.PutInputData "PlaneWingPanelRight", Item_to_set4
Set Item_to_set5 = hybridShapeLoft1
Factory.PutInputData "WingLowerSurface", Item_to_set5

Set Instance = Factory.Instantiate
Factory.EndInstantiate
Factory.EndInstanceFactory

part1.Update

' SPAR JOIN CODE HERE '
' ************************************'
' Code to add new spar to the original spar join '

Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spars")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeAssemble1 = hybridShapes1.Item("Spars")

Set hybridBody3 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes3 = hybridBody3.HybridShapes
Set hybridShapeLoft3 = hybridShapes3.Item("SplitSpar")
Set reference3 = part1.CreateReferenceFromObject(hybridShapeLoft3)
hybridShapeAssemble1.AddElement reference3

hybridShapeAssemble1.SetConnex 0
hybridShapeAssemble1.SetManifold 0
hybridShapeAssemble1.SetSimplify 0
hybridShapeAssemble1.SetSuppressMode 0
hybridShapeAssemble1.SetDeviation 0.001
hybridShapeAssemble1.SetAngularToleranceMode 0
hybridShapeAssemble1.SetAngularTolerance 0.5
hybridShapeAssemble1.SetFederationPropagation 0
part1.InWorkObject = hybridShapeAssemble1
part1.Update

Next
trigger_ribs.Value = trigger_ribs.Value + 1
End If
```

```
' ****************************************************************************
'
' If there is more than one wing panel | and new nr ins are greater than old nr ins
' ****************************************************************************
'
If new_nr_ins.Value > 1 And new_nr_ins.Value > old_nr_ins.Value Then
I_nr = 1
For NRS = old_spars.Value + 1 To new_spars.Value

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeLine1 = hybridShapes1.Item("Line1")
Set hybridShapeLine2 = hybridShapes1.Item("Line2")

' Create Points on Line 1 and Line 2 for the Wing Panel
' Line1

' Create some references to create the new point between the old point and SP2
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP1" & NRS - 1 & I_nr)
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridBodies2 = hybridBody1.HybridBodies
Set hybridBody2 = hybridBodies2.Item("PointsforSpar")
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapeProject1 = hybridShapes1.Item("SP2")
Set ref2 = part1.CreateReferenceFromObject(hybridShapeProject1)
'end of references
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointBetween1 = hybridShapeFactory1.AddNewPointBetween(ref1, ref2,
0.25, 1)
hybridBody1.AppendHybridShape hybridShapePointBetween1
part1.InWorkObject = hybridShapePointBetween1

hybridShapePointBetween1.Name = "SparP1" & NRS & I_nr
part1.Update

' Line2
' Create some references to create the new point between the old point and SP4
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve3 = hybridShapes1.Item("SparP2" & NRS - 1 & I_nr)
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve3)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridBodies2 = hybridBody1.HybridBodies
Set hybridBody2 = hybridBodies2.Item("PointsforSpar")
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapeProject2 = hybridShapes1.Item("SP4")
Set ref2 = part1.CreateReferenceFromObject(hybridShapeProject2)
'end of references

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointBetween2 = hybridShapeFactory1.AddNewPointBetween(ref1, ref2,
0.25, 1)
hybridBody1.AppendHybridShape hybridShapePointBetween2
part1.InWorkObject = hybridShapePointBetween2

hybridShapePointBetween2.Name = "SparP2" & NRS & I_nr
part1.Update

' Creates Parameter and Formula for Position of Points
' Point of Line1
Set spar_pos = parameters1.CreateReal("", 0.25)
spar_pos.Rename "SparPositionP1." & NRS & I_nr
```

```
Set realParam2 = parameters1.Item("Wing\WingPanel." & I_nr & "\SparP1" & NRS & I_nr
& "\Ratio")
Set Formula1 = relations1.CreateFormula("sparfP1" & NRS & I_nr, "", realParam2,
"`SparPositionP1." & NRS & I_nr & "` ")
part1.Update

' Point of Line2
Set spar_pos = parameters1.CreateReal("", 0.25)
spar_pos.Rename "SparPositionP2." & NRS & I_nr
Set realParam2 = parameters1.Item("Wing\WingPanel.1" & "\SparP2" & NRS & I_nr &
"\Ratio")
Set Formula1 = relations1.CreateFormula("sparfP2" & NRS & I_nr, "", realParam2,
"`SparPositionP2." & NRS & I_nr & "` ")
part1.Update

Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Add()
hybridBody1.Name = "Spar." & NRS
part1.Update

' add points which are in z direction to points created
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, -10000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointBetween1 = hybridShapes1.Item("SparP1" & NRS & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointBetween1)
hybridShapePointCoord1.PtRef = reference1
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
hybridBody2.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
hybridShapePointCoord1.Name = "SparP1" & NRS & I_nr & "-z"
part1.Update

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, 10000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointBetween1 = hybridShapes1.Item("SparP2" & NRS & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointBetween1)
hybridShapePointCoord1.PtRef = reference1
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
hybridBody2.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
hybridShapePointCoord1.Name = "SparP2" & NRS & I_nr & "-z"
part1.Update

Next

' Create Polyline to connect points

For NRS = old_spars.Value + 1 To new_spars.Value

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePolyline1 = hybridShapeFactory1.AddNewPolyline()
Set hybridBodies1 = part1.HybridBodies

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP1" & NRS & I_nr & "-z")
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP1" & NRS & I_nr)
```

```
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS + 1

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS + 2

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr & "-z")
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS + 3

hybridShapePolyline1.Closure = False
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
hybridBody1.AppendHybridShape hybridShapePolyline1
hybridShapePolyline1.Name = "SparPolyline"
part1.InWorkObject = hybridShapePolyline1
part1.Update

Next

Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Item("Wing1.CATPart")
Set part1 = partDocument1.Part
Set parameters1 = part1.Parameters
Set relations1 = part1.Relations

Set new_spars = parameters1.Item("New Number of Spars")
Set old_spars = parameters1.Item("Old Number of Spars")

For NRS = old_spars.Value + 1 To new_spars.Value

Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Wing")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeAssemble1 = hybridShapes1.Item("Wing")

Set hybridBody1 = hybridBodies1.Item("Wing")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeLoft1 = hybridShapes1.Item("WingLowerSurface")
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")

Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points2 = hybridShapes1.Item("PlaneWingPanelLeft")
Set hybridBodies1 = part1.HybridBodies
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapePolyline1 = hybridShapes1.Item("SparPolyline")

part1.InWorkObject = hybridBody2

Set Factory = part1.GetCustomerFactory("InstanceFactory")
Factory.BeginInstanceFactory "PowerCopy", "D:\Documents and
Settings\Sohaib\Desktop\Thesis Project Files\June\12-July-
2011\Powercopy\SparPC.CATPart"
Factory.BeginInstantiate

Set Item_to_set1 = hybridShapePolyline1
Factory.PutInputData "SparPolyline", Item_to_set1
Set Item_to_set2 = hybridShapeAssemble1
Factory.PutInputData "Wing", Item_to_set2
Set Item_to_set3 = hybridShapePlane3Points2
```

```
Factory.PutInputData "PlaneWingPanelLeft", Item_to_set3
Set Item_to_set4 = hybridShapePlane3Points1
Factory.PutInputData "PlaneWingPanelRight", Item_to_set4
Set Item_to_set5 = hybridShapeLoft1
Factory.PutInputData "WingLowerSurface", Item_to_set5

Set Instance = Factory.Instantiate
Factory.EndInstantiate
Factory.EndInstanceFactory

part1.Update

' SPAR JOIN CODE HERE '
' ************************************'
' Code to add new spar to the original spar join '

Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spars")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeAssemble1 = hybridShapes1.Item("Spars")

Set hybridBody3 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes3 = hybridBody3.HybridShapes
Set hybridShapeLoft3 = hybridShapes3.Item("SplitSpar")
Set reference3 = part1.CreateReferenceFromObject(hybridShapeLoft3)
hybridShapeAssemble1.AddElement reference3

hybridShapeAssemble1.SetConnex 0
hybridShapeAssemble1.SetManifold 0
hybridShapeAssemble1.SetSimplify 0
hybridShapeAssemble1.SetSuppressMode 0
hybridShapeAssemble1.SetDeviation 0.001
hybridShapeAssemble1.SetAngularToleranceMode 0
hybridShapeAssemble1.SetAngularTolerance 0.5
hybridShapeAssemble1.SetFederationPropagation 0
part1.InWorkObject = hybridShapeAssemble1
part1.Update

If new_nr_ins.Value > 1 Then
For I_nr = old_nr_ins.Value + 1 To new_nr_ins.Value

' Add new points to the first spar
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS - 1)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS - 1 & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridBodies2 = hybridBody1.HybridBodies
Set hybridBody2 = hybridBodies2.Item("PointsforSpar")
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapeProject1 = hybridShapes1.Item("SP4")
Set ref2 = part1.CreateReferenceFromObject(hybridShapeProject1)

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointBetween1 = hybridShapeFactory1.AddNewPointBetween(reference1,
ref2, 0.25, 1)
hybridShapePointBetween1.Name = "SparP2" & NRS & I_nr


' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
' if we were to add to the point inside the wing panel hybrid body
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
```

```
' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

hybridBody2.AppendHybridShape hybridShapePointBetween1
part1.InWorkObject = hybridShapePointBetween1
part1.Update

' Creates Parameter and Formula for Position of Points
Set spar_pos = parameters1.CreateReal("", 0.25)
spar_pos.Rename "SparPositionP2." & NRS & I_nr
Set realParam2 = parameters1.Item("Wing\Spar." & NRS & "\SparP2" & NRS & I_nr &
"\Ratio")
Set Formula1 = relations1.CreateFormula("sparfP2" & NRS & I_nr, "", realParam2,
"`SparPositionP2." & NRS & I_nr & "` ")
part1.Update

If I_nr = new_nr_ins.Value Then
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, 5000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr)
hybridShapePointCoord1.Name = "SparP2" & NRS & I_nr & "-z"
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePointCoord1.PtRef = reference1
hybridBody1.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
part1.Update

' Delete last polyline point and add new points to polyline created above

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePolyline1 = hybridShapes1.Item("SparPolyline")
Dim nrpoly As Long
nrpoly = hybridShapePolyline1.NumberOfElements
hybridShapePolyline1.RemoveElement nrpoly

Dim nrpoly_new As Long
nrpoly_new = hybridShapePolyline1.NumberOfElements

i = 1

For I_nrx = old_nr_ins.Value + 1 To new_nr_ins.Value
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nrx)
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement ref1, nrpoly_new + i
i = i + 1
Next
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr & "-z")
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement ref1, nrpoly_new + i

' replace the plane in the split spar
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitSpar")

Set hybridBody1 = hybridBodies1.Item("WingPanel." & old_nr_ins.Value)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcut = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcutnew = part1.CreateReferenceFromObject(hybridShapePlane3Points1)
```

```
hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in Split Solid spar
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitWithWingPanelPlaneRight")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in SparSurfaceA and SparSurfaceB
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceA")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceB")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

End If
Next

End If

Next
End If

'
************************************************************************************
************************ '
' If there is more than one wing panel and new_nr_ins and old_nr_ins are equal and
newspars are greater than 1
'
************************************************************************************
********************** '
If new_nr_ins.Value = old_nr_ins.Value And new_nr_ins.Value > 1 And new_spars.Value
> 1 Then

For NRS = old_spars.Value + 1 To new_spars.Value
I_nr = 1
Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeLine1 = hybridShapes1.Item("Line1")
Set hybridShapeLine2 = hybridShapes1.Item("Line2")

' Create Points on Line 1 and Line 2 for the Wing Panel
' Line1

' Create some references to create the new point between the old point and SP2
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP1" & NRS - 1 & I_nr)
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridBodies2 = hybridBody1.HybridBodies
Set hybridBody2 = hybridBodies2.Item("PointsforSpar")
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapeProject1 = hybridShapes1.Item("SP2")
Set ref2 = part1.CreateReferenceFromObject(hybridShapeProject1)
'end of references
```

```
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointBetween1 = hybridShapeFactory1.AddNewPointBetween(ref1, ref2,
0.25, 1)
hybridBody1.AppendHybridShape hybridShapePointBetween1
part1.InWorkObject = hybridShapePointBetween1
hybridShapePointBetween1.Name = "SparP1" & NRS & I_nr
part1.Update

' Line2
' Create some references to create the new point between the old point and SP4
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve3 = hybridShapes1.Item("SparP2" & NRS - 1 & I_nr)
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve3)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridBodies2 = hybridBody1.HybridBodies
Set hybridBody2 = hybridBodies2.Item("PointsforSpar")
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapeProject2 = hybridShapes1.Item("SP4")
Set ref2 = part1.CreateReferenceFromObject(hybridShapeProject2)
'end of references

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointBetween2 = hybridShapeFactory1.AddNewPointBetween(ref1, ref2,
0.25, 1)
hybridBody1.AppendHybridShape hybridShapePointBetween2
part1.InWorkObject = hybridShapePointBetween2

hybridShapePointBetween2.Name = "SparP2" & NRS & I_nr
part1.Update

' Creates Parameter and Formula for Position of Points
' Point of Line1
Set spar_pos = parameters1.CreateReal("", 0.25)
spar_pos.Rename "SparPositionP1." & NRS & I_nr
Set realParam2 = parameters1.Item("Wing\WingPanel." & I_nr & "\SparP1" & NRS & I_nr
& "\Ratio")
Set Formula1 = relations1.CreateFormula("sparfP1" & NRS & I_nr, "", realParam2,
"`SparPositionP1." & NRS & I_nr & "` ")
part1.Update

' Point of Line2
Set spar_pos = parameters1.CreateReal("", 0.25)
spar_pos.Rename "SparPositionP2." & NRS & I_nr
Set realParam2 = parameters1.Item("Wing\WingPanel.1" & "\SparP2" & NRS & I_nr &
"\Ratio")
Set Formula1 = relations1.CreateFormula("sparfP2" & NRS & I_nr, "", realParam2,
"`SparPositionP2." & NRS & I_nr & "` ")
part1.Update

Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Add()
hybridBody1.Name = "Spar." & NRS
part1.Update

' add points which are in z direction to points created
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, -10000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointBetween1 = hybridShapes1.Item("SparP1" & NRS & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointBetween1)
hybridShapePointCoord1.PtRef = reference1
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
hybridBody2.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
```

```
hybridShapePointCoord1.Name = "SparP1" & NRS & I_nr & "-z"
part1.Update

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, 10000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointBetween1 = hybridShapes1.Item("SparP2" & NRS & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointBetween1)
hybridShapePointCoord1.PtRef = reference1
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
hybridBody2.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
hybridShapePointCoord1.Name = "SparP2" & NRS & I_nr & "-z"
part1.Update
Next

' Create Polyline to connect points

For NRS = old_spars.Value + 1 To new_spars.Value

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePolyline1 = hybridShapeFactory1.AddNewPolyline()
Set hybridBodies1 = part1.HybridBodies

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP1" & NRS & I_nr & "-z")
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP1" & NRS & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS + 1

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS + 2

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr & "-z")
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement reference1, NRS + 3

hybridShapePolyline1.Closure = False
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
hybridBody1.AppendHybridShape hybridShapePolyline1
hybridShapePolyline1.Name = "SparPolyline"
part1.InWorkObject = hybridShapePolyline1
part1.Update

Next

Set documents1 = CATIA.Documents
Set partDocument1 = documents1.Item("Wing1.CATPart")
Set part1 = partDocument1.Part
Set parameters1 = part1.Parameters
Set relations1 = part1.Relations

Set new_spars = parameters1.Item("New Number of Spars")
Set old_spars = parameters1.Item("Old Number of Spars")
```

```
For NRS = old_spars.Value + 1 To new_spars.Value

Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Wing")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeAssemble1 = hybridShapes1.Item("Wing")

Set hybridBody1 = hybridBodies1.Item("Wing")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeLoft1 = hybridShapes1.Item("WingLowerSurface")
Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")

Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points2 = hybridShapes1.Item("PlaneWingPanelLeft")
Set hybridBodies1 = part1.HybridBodies
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapePolyline1 = hybridShapes1.Item("SparPolyline")

part1.InWorkObject = hybridBody2

Set Factory = part1.GetCustomerFactory("InstanceFactory")
Factory.BeginInstanceFactory "PowerCopy", "D:\Documents and
Settings\Sohaib\Desktop\Thesis Project Files\June\12-July-
2011\Powercopy\SparPC.CATPart"
Factory.BeginInstantiate

Set Item_to_set1 = hybridShapePolyline1
Factory.PutInputData "SparPolyline", Item_to_set1
Set Item_to_set2 = hybridShapeAssemble1
Factory.PutInputData "Wing", Item_to_set2
Set Item_to_set3 = hybridShapePlane3Points2
Factory.PutInputData "PlaneWingPanelLeft", Item_to_set3
Set Item_to_set4 = hybridShapePlane3Points1
actory.PutInputData "PlaneWingPanelRight", Item_to_set4
Set Item_to_set5 = hybridShapeLoft1
Factory.PutInputData "WingLowerSurface", Item_to_set5

Set Instance = Factory.Instantiate
Factory.EndInstantiate
Factory.EndInstanceFactory

part1.Update

' SPAR JOIN CODE HERE '
' ************************************'
' Code to add new spar to the original spar join '

Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spars")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeAssemble1 = hybridShapes1.Item("Spars")

Set hybridBody3 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes3 = hybridBody3.HybridShapes
Set hybridShapeLoft3 = hybridShapes3.Item("SplitSpar")
Set reference3 = part1.CreateReferenceFromObject(hybridShapeLoft3)
hybridShapeAssemble1.AddElement reference3

hybridShapeAssemble1.SetConnex 0
hybridShapeAssemble1.SetManifold 0
hybridShapeAssemble1.SetSimplify 0
hybridShapeAssemble1.SetSuppressMode 0
```

```
hybridShapeAssemble1.SetDeviation 0.001
hybridShapeAssemble1.SetAngularToleranceMode 0
hybridShapeAssemble1.SetAngularTolerance 0.5
hybridShapeAssemble1.SetFederationPropagation 0
part1.InWorkObject = hybridShapeAssemble1
part1.Update


If new_nr_ins.Value > 1 Then

For I_nr = 2 To new_nr_ins.Value

' Add new points to the first spar
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS - 1)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS - 1 & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridBodies2 = hybridBody1.HybridBodies
Set hybridBody2 = hybridBodies2.Item("PointsforSpar")
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapeProject1 = hybridShapes1.Item("SP4")
Set ref2 = part1.CreateReferenceFromObject(hybridShapeProject1)

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointBetween1 = hybridShapeFactory1.AddNewPointBetween(reference1,
ref2, 0.25, 1)
hybridShapePointBetween1.Name = "SparP2" & NRS & I_nr


' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
' if we were to add to the point inside the wing panel hybrid body
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

hybridBody2.AppendHybridShape hybridShapePointBetween1
part1.InWorkObject = hybridShapePointBetween1
part1.Update

' Creates Parameter and Formula for Position of Points
Set spar_pos = parameters1.CreateReal("", 0.25)
spar_pos.Rename "SparPositionP2." & NRS & I_nr
Set realParam2 = parameters1.Item("Wing\Spar." & NRS & "\SparP2" & NRS & I_nr &
"\Ratio")
Set Formula1 = relations1.CreateFormula("sparfP2" & NRS & I_nr, "", realParam2,
"`SparPositionP2." & NRS & I_nr & "` ")
part1.Update

If I_nr = new_nr_ins.Value Then
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, 5000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr)
hybridShapePointCoord1.Name = "SparP2" & NRS & I_nr & "-z"
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePointCoord1.PtRef = reference1
hybridBody1.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
part1.Update

' Delete last polyline point and add new points to polyline created above
```

```
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePolyline1 = hybridShapes1.Item("SparPolyline")
'Dim nrpoly As Long
nrpoly = hybridShapePolyline1.NumberOfElements
hybridShapePolyline1.RemoveElement nrpoly

'Dim nrpoly_new As Long
nrpoly_new = hybridShapePolyline1.NumberOfElements

i = 1

For I_nrx = 2 To new_nr_ins.Value
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nrx)
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement ref1, nrpoly_new + i
i = i + 1
Next
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr & "-z")
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement ref1, nrpoly_new + i

' replace the plane in the split spar
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitSpar")

Set hybridBody1 = hybridBodies1.Item("WingPanel." & "1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcut = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcutnew = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in Split Solid spar
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitWithWingPanelPlaneRight")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in SparSurfaceA and SparSurfaceB
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceA")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceB")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1
End If
Next

End If

Next
```

```
End If



End If


If old_spars.Value = new_spars.Value And new_spars.Value > 1 And new_nr_ins.Value >
1 Then

' %%%%%%%%%%%%%%%%%%%%%%%%%%%%
' new nr of wing panels is greater than one
' %%%%%%%%%%%%%%%%%%%%%%%%%%%%

For NRS = 2 To new_spars.Value
For I_nr = old_nr_ins.Value + 1 To new_nr_ins.Value

' Add new points to the first spar
Set part1 = partDocument1.Part
Set hybridBodies1 = part1.HybridBodies

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS - 1)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS - 1 & I_nr)
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridBodies2 = hybridBody1.HybridBodies
Set hybridBody2 = hybridBodies2.Item("PointsforSpar")
Set hybridShapes1 = hybridBody2.HybridShapes
Set hybridShapeProject1 = hybridShapes1.Item("SP4")
Set ref2 = part1.CreateReferenceFromObject(hybridShapeProject1)

Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointBetween1 = hybridShapeFactory1.AddNewPointBetween(reference1,
ref2, 0.25, 1)
hybridShapePointBetween1.Name = "SparP2" & NRS & I_nr

' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
' if we were to add to the point inside the wing panel hybrid body
Set hybridBody2 = hybridBodies1.Item("Spar." & NRS)
' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

hybridBody2.AppendHybridShape hybridShapePointBetween1
part1.InWorkObject = hybridShapePointBetween1
part1.Update

' Creates Parameter and Formula for Position of Points
Set spar_pos = parameters1.CreateReal("", 0.25)
spar_pos.Rename "SparPositionP2." & NRS & I_nr
Set realParam2 = parameters1.Item("Wing\Spar." & NRS & "\SparP2" & NRS & I_nr &
"\Ratio")
Set Formula1 = relations1.CreateFormula("sparfP2" & NRS & I_nr, "", realParam2,
"`SparPositionP2." & NRS & I_nr & "` ")
part1.Update

If I_nr = new_nr_ins.Value Then
Set part1 = partDocument1.Part
Set hybridShapeFactory1 = part1.HybridShapeFactory
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(0, 0, 5000)
Set hybridBodies1 = part1.HybridBodies
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr)
hybridShapePointCoord1.Name = "SparP2" & NRS & I_nr & "-z"
```

```
Set reference1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePointCoord1.PtRef = reference1
hybridBody1.AppendHybridShape hybridShapePointCoord1
part1.InWorkObject = hybridShapePointCoord1
part1.Update

' Delete last polyline point and add new points to polyline created above

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePolyline1 = hybridShapes1.Item("SparPolyline")
'Dim nrpoly As Long
nrpoly = hybridShapePolyline1.NumberOfElements
hybridShapePolyline1.RemoveElement nrpoly

'Dim nrpoly_new As Long
nrpoly_new = hybridShapePolyline1.NumberOfElements

i = 1

For I_nrx = old_nr_ins.Value + 1 To new_nr_ins.Value
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nrx)
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement ref1, nrpoly_new + i
i = i + 1
Next
Set hybridShapePointOnCurve1 = hybridShapes1.Item("SparP2" & NRS & I_nr & "-z")
Set ref1 = part1.CreateReferenceFromObject(hybridShapePointOnCurve1)
hybridShapePolyline1.InsertElement ref1, nrpoly_new + i

' replace the plane in the split spar
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitSpar")

Set hybridBody1 = hybridBodies1.Item("WingPanel." & old_nr_ins.Value)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcut = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

Set hybridBody1 = hybridBodies1.Item("WingPanel." & I_nr)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePlane3Points1 = hybridShapes1.Item("PlaneWingPanelRight")
Set refcutnew = part1.CreateReferenceFromObject(hybridShapePlane3Points1)

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in Split Solid spar
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SplitWithWingPanelPlaneRight")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

' replace the plane in SparSurfaceA and SparSurfaceB
Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceA")

hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

Set hybridBody1 = hybridBodies1.Item("Spar." & NRS)
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapeSplit1 = hybridShapes1.Item("SparSurfaceB")
```

```
hybridShapeSplit1.RemoveCuttingElem refcut
hybridShapeSplit1.AddCuttingElem refcutnew, 1

End If
Next
Next


' %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
'%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
'%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

' SPAR CODE COMPLETED HERE

End If

Dim v As Long
Dim r As Long

v = 1
r = 0
Dim y As Long
Dim c As Long
Dim nrribs As Long

If old_spars.Value > new_spars.Value Then
trigger_ribs.Value = trigger_ribs.Value + 1
If new_ribs.Value > 1 Then

nrribs = new_ribs.Value

new_ribs.Value = 1
trigger_ribs.Value = trigger_ribs.Value + 1

End If

For NRS = new_spars.Value + 1 To old_spars.Value

y = old_spars.Value - new_spars.Value

Set Pi = parameters1.Item("SparPositionP1." & y + (NRS + r) - v & "1")
selection1.Add Pi

Set F1 = part1.FindObjectByName("sparfP1" & y + (NRS + r) - v & "1")
selection1.Add F1

Set F2 = part1.FindObjectByName("sparthickfa." & y + (NRS + r) - v)
selection1.Add F2
Set F3 = part1.FindObjectByName("sparthickfb." & y + (NRS + r) - v)
selection1.Add F3
Set F4 = part1.FindObjectByName("sparoffsetfa." & y + (NRS + r) - v)
selection1.Add F4
Set F5 = part1.FindObjectByName("sparoffsetfb." & y + (NRS + r) - v)
selection1.Add F5

Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointBetween1 = hybridShapes1.Item("SparP1" & y + (NRS + r) - v &
"1")
selection1.Add hybridShapePointBetween1

Set hybridBody1 = hybridBodies1.Item("WingPanel.1")
Set hybridShapes1 = hybridBody1.HybridShapes
Set hybridShapePointBetween1 = hybridShapes1.Item("SparP2" & y + (NRS + r) - v &
"1")
selection1.Add hybridShapePointBetween1

For I_nr = 1 To new_nr_ins.Value
```

```
y = old_spars.Value - new_spars.Value
c = old_nr_ins.Value - new_nr_ins.Value

Set Pi = parameters1.Item("SparPositionP2." & y + (NRS + r) - v & I_nr)
selection1.Add Pi

Set F2 = part1.FindObjectByName("sparfP2" & y + (NRS + r) - v & I_nr)
selection1.Add F2
Next

On Error Resume Next
CATIA.DisplayAlerts = False

y = old_spars.Value - new_spars.Value

Set P1 = parameters1.Item("SparThickness." & y + (NRS + r) - v)
selection1.Add P1

Set hybridBody1 = hybridBodies1.Item("Spar." & y + (NRS + r) - v)
selection1.Add hybridBody1

v = v + 1
r = r - 1

selection1.Delete
selection1.Clear
Next
part1.Update

If nrribs > 1 Then
new_ribs.Value = nrribs
trigger_ribs.Value = trigger_ribs.Value + 1
End If

End If

old_spars.Value = new_spars.Value
old_nr_ins.Value = new_nr_ins.Value
part1.Update

End Sub
```