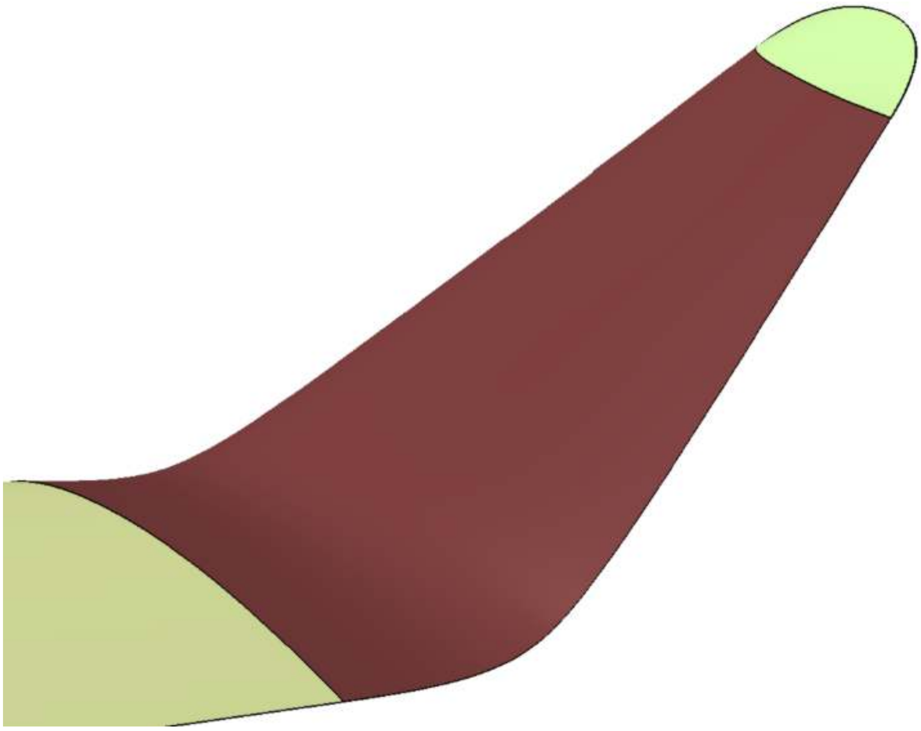
Linkoping studies in Science and Technology

Design of Parametric Winglets and Wing tip devices – A Conceptual Design Approach

Saravanan Rajendran



Linkoping University
Institute of Technology
Department of Management and Engineering (IEI)
SE-58183 Linköping, Sweden

Linkoping studies in Science and Technology

Design of Parametric Winglets and Wing tip devices – A Conceptual Design Approach

Saravanan Rajendran

LIU-IEI-TEK-A--12/01371--SE

Linkoping University
Institute of Technology
Department of Management and Engineering (IEI)
SE-58183 Linkoping, Sweden

Supervisor

Raghu Chaitanya M V
PhD Student – Linkoping University

Examiner

Christopher Jouannet
Assistant Professor, Linkoping University

Linkoping 2012

Acknowledgement

I wish to thank the Division of Fluid and Mechatronic Systems (FLUMES), Department of Management and Engineering (IEI), Linkoping University, Sweden for providing the facilities to carry out this master thesis work. My sincere thanks to Dr. Christopher Jouannet for giving me the opportunity to learn about design technology in aircraft field. I express my sincere gratitude to Mr. Raghu Chaitanya for providing continuous support and persistent guidance throughout my project period. Furthermore, I would also like to thank Mr.Patrick Berry for giving suggestions and feedbacks regarding this project. The valuable comments and reviews from the above mentioned people were very helpful in making this a complete work.

Finally, I am very grateful to all my friends who supported and motivated me regarding this thesis work.

Saravanan Rajendran

Linköping, May 2012

Abstract

Winglets being a small structure play an important role in reducing the induced drag in aircraft. Many types of winglets have been designed and their significance in reducing the drag is published. One of the main objectives of this master thesis work is to study about the winglet design and about their contribution in reducing induced drag. A brief overview of wing tip devices and their performance from the manufacturers as well as from airliner’s point of view are discussed. Moreover, the role of winglet in reducing the drag of commercial civil jet aircraft is studied and the percentage of drag reduction is calculated by a conceptual approach. A320 specifications are taken to perform induced drag reduction calculation with and without winglets. Indeed, the total drag count reduced with the help of winglets accounts for additional payload which will be an advantage for the aircraft operator.

Reducing the process time in design is one of the important criteria for any field and hence automation with help of CAD tools is very significant in reducing time. This study also aims at developing an automated model for different types of winglets and wing tip devices with the help of CAD technology focused on reducing design time during the initial design process. Knowledge based approach is used in this work and all the models are parameterized so each model could be varied with associated parameters. The generic model created would take different shapes and switches between different types of wing tip devices as per the user’s requirement with the help of available parameters. Knowledge Pattern (KP) approach is used to develop the automation process. User Defined Features (UDFs) are created for each type of winglet and tip devices. CATIA V5 R18 software is used to develop the models of winglets and tip devices.

Keywords: *Winglet, Induced drag, CATIA V5, Knowledge Pattern, Parameterization.*

List of abbreviations

NASA – National Aeronautics and Space Administration

CATIA - Computer Aided Three dimensional Interactive Application

VB – Visual Basic

KP – Knowledge Pattern

API – Aviation Partner Inc.

APB – Aviation Partners Boeing

CAE – Computer Aided Engineering

C_{d0} – Profile drag coefficient

C_{di} – Induced drag coefficient

C_d–Total drag coefficient

F–Form factor

Q–Interference factor

Swet–Wetted area of the surface

Sref –Wing area

Re_c = Reynolds number of the component

V = Velocity

l = component characteristics length

u = kinematic viscosity for that flight condition

e – Ostwald’s efficiency factor

AR – Aspect ratio

Cl – Lift coefficient

Λ – Sweep angle

UDF – User Defined Feature

Contents

List of Figures

1. Introduction

1.1. Induced drag reduction

Aircraft designers are performing research to improve the overall aircraft efficiency which would be beneficial for both aircraft manufacturer and the airlines. In case of aircraft design process, reducing the overall drag at a compromising level would be one of the challenging processes. Many experiments have been conducted in order to reduce the lift-induced drag. During the later stages of 19th century, F.W. Lanchester, a British scientist developed the end plate technique to reduce the induced drag [1]. Trefftz-plane theory by Prandtl in 1910 was developed in order to determine the induced drag and also this theory could determine the induced drag both locally and globally. Trefftz theory has been used to determine the effects of induced drag by many researchers [2]. Later, the concept of induced drag and the method to determine the induced drag by conformal mapping and electrical potential analogy method was developed [3]. According to Cone, the induced drag factor could be pronounced by the effective aspect ratio which could be determined by electrical analogy of the lifting system. Moreover, many wingtip devices and tip extensions were developed from the beginning of 19th century and researches are going on till date for induced drag reduction. The evolution and recent developments in winglet technology will be discussed in later section. Also, the main purpose of winglet from the perspective of both aircraft manufacturers and operators will be discussed.

1.2. Computer Aided Design

Aircraft design being a complex process has many phases in which Computer Aided Design (CAD) plays a significant role. Many aircraft manufacturers such as Boeing, Dassault, and Airbus have been adopting the CAD software tool like CATIA in order to minimize the lead time and to avoid prolonged duration in design process. CAD combined with Knowledge based engineering (KBE) aimed at reducing time the taken for design process in case of repetition [4]. Studies have been done on developing parameterized CAD models focusing to optimize the given model with less duration of time [5-7]. D operator and K operator were the two approaches developed with CAE tools for making repetitive process. VB script is associated with D operator, whereas Knowledge Pattern (KP) developed based on C++ programming language, is under the K operator approach [8]. KP has been implemented in Dassault systems software CATIA V5 R16. One of the main disadvantages in VB script for dynamic instantiation of the models is longer time consumption for scripting [8-10]. Studies revealed that automation for creating models and patterns dynamically were done based on Knowledge Pattern script where the time consumed for pattern creation and scripting were much lesser than VB approach [8-10].

1.3. Aim

The objective of this study is to develop an automated parametric model for different types of winglets and wingtip devices using the software CATIA V5. Knowledge Pattern approach is used for the automation process. The main tasks of this thesis work are:

- To find the induced drag contribution in civil jet transport aircraft by conceptual approach
- To analyze the aircraft performance with and without winglets
- To develop the parametric model of winglet and wingtip devices
- To implement Knowledge Pattern for automation

The main role of the model is to reduce the time taken by the designer during the design process. The developed generic model can switch between different types of wing tip devices as per the user's requirements with available parameters. The model created will be useful not only in aircraft wing design but also in propeller design for wind turbines.

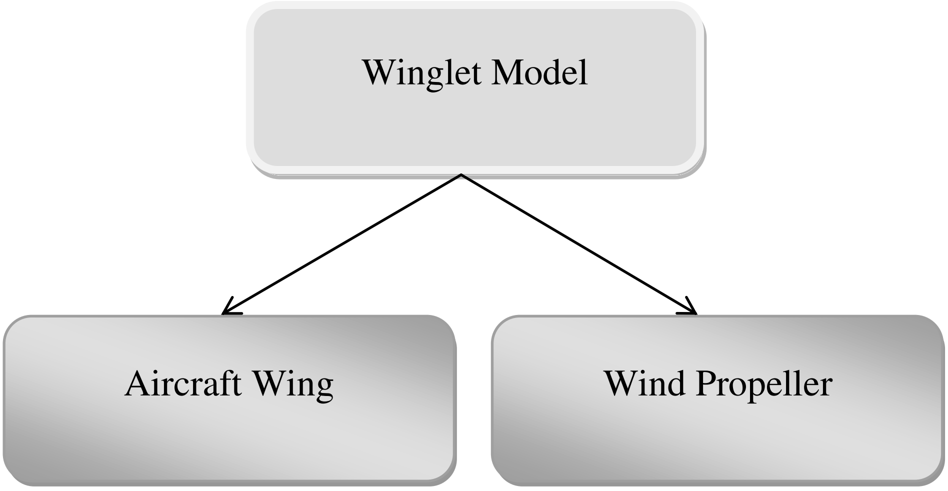


Figure 1 Applications of parameteric winglet model

2. Literature review

2.1. History of Wingtip devices and Winglets

Endplate theory was the first to propose wingtip device and was patented by Fredrick W. Lanchester, British Aerodynamicist in 1897. Unfortunately, his theory could not reduce the overall drag of aircraft despite reducing the induced drag. The increase in the viscous drag during cruise conditions outruns the reduction in induced drag[1]. In July 1976, Dr.Whitcomb made a research at NASA Langley research center and developed the concept of winglet technology. According to Whitcomb, winglet could be described as the small wing like vertical structures which extends from the wingtip, aiming at reduction in induced drag when compared to other wing tip devices or extensions. He also claimed in his research that the winglet shows 20% reduction in induced drag when compared to tip extension and also improved lift-to-drag ratio [1].

In 1994 Aviation Partners Inc. (API) developed an advance design of winglet called blended winglet. Louis B. Gratzer from Seattle has the patent for blended winglet and intention of the winglet is to reduce the interference drag due to sharp edges as seen in the Whitcomb's winglet [11]. Also, Gratzer has the patent for the invention of spiroid-tipped wing in April 7, 1992 [12]. Later, "wing grid" concept was developed by La Roche from Switzerland in 1996 and got the patent for his invention [13]. The main purpose of all the above inventions was to decrease the strength of wake vortex and to reduce induced drag.

2.2. Types of Winglets and Wingtip devices

After the invention of winglet by Whitcomb, many types of winglets and tip devices were developed by aircraft designers. Some of the inventions of winglets by the respective aircraft manufacturer are discussed in the following section.

2.2.1. Blended Winglet

Blended winglet was developed by Grazter from Seattle in 1994. The unique design in this winglet is no sharp edge found at the wing/winglet intersection and followed by smooth curve. Aviation Partners Inc. (API) and Boeing Company made collaboration in 1999 for the design of advance blended winglets in 1999. Mike Stowell, Executive vice president of APB mentioned about the interference drag, an aerodynamic phenomenon caused due to intersection of lifting surfaces, hence the winglet design was developed to overcome the interference drag formed at the junction of wing and winglet. The winglets were retrofitted in Boeing business jets and also in B737[1]. Now these flights have their services in American airlines (Southwest airlines) and also in European airlines (Ryanair)[1].

[1] http://spinoff.nasa.gov/Spinoff2010/t_5.html

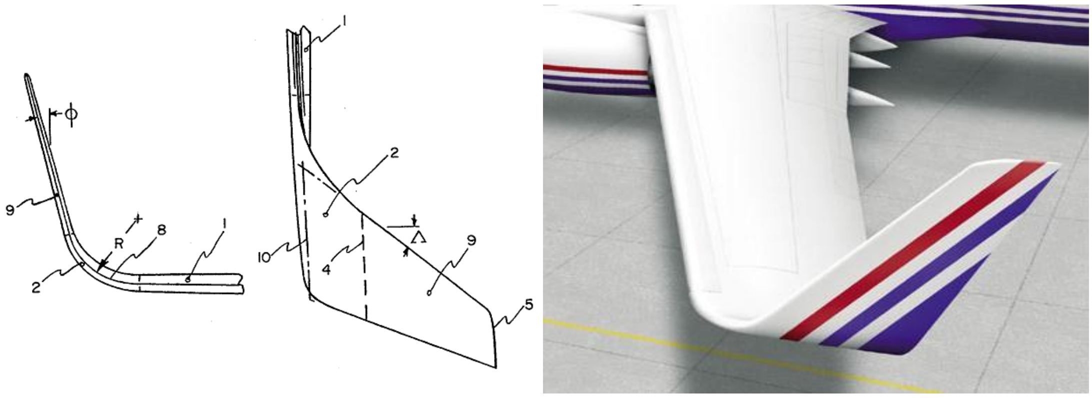


Figure 2 Blended Winglet[11]

2.2.2. Spiroid Winglet

Gratzer has developed the spiroid-tipped wing technology and got the patent in 1992. One end of the spiroid tip is attached with forward part of the wing tip and continues to form a spiral loop which ends at the aft portion of the wing tip. Hence it looks oval shaped when viewed from front. Spiroid tipped wing was created to reduce the induced drag and also to reduce the noise effects associated with the tip vortices [12]. API has made their flight test in Dassault Falcon 50 with spiroid tipped wing.

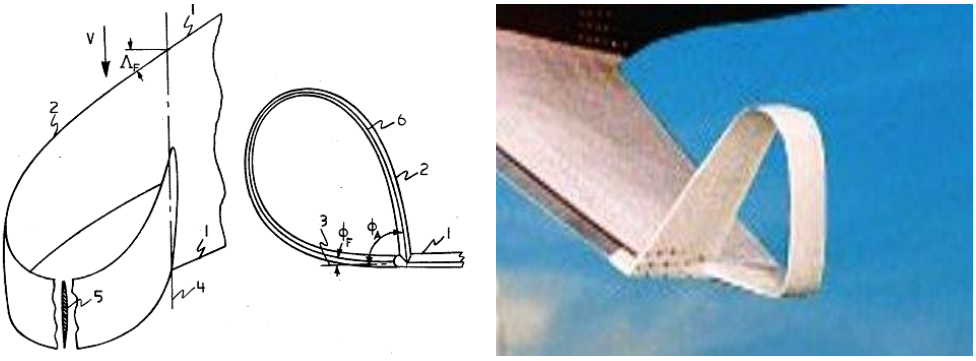


Figure 3 Spiroid tipped wing[12]

2.2.3. Wing-grid as wing tip

Wing grid geometry is defined by two or more wing like surfaces running parallel to each other from the end of wing section which forms the grid. La Roche from Switzerland held the patent for this invention since October, 1998. Instead of entire wing with no tip devices, wing grid at partial span could be replaced. La Roche claimed that wing grid could provide much reduction in induced drag when compared to wing span extension [13].

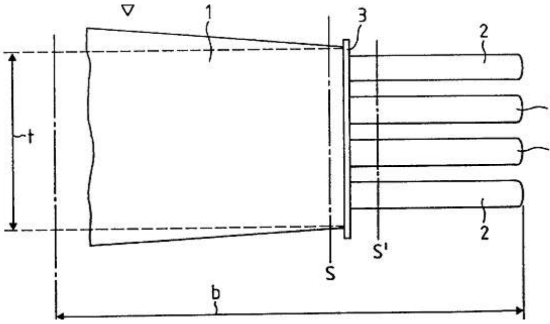


Figure 4 Wing grids as wing end section[13]

2.2.4. Wing tip devices

Raked wing tip from Boeing Company was designed by Herrick and got the patent in 2000 [14]. The raked tip is attached with the main wingtip with higher angle of sweep than the main wing. Boeing 777 long-range jets have been designed with raked wingtip.

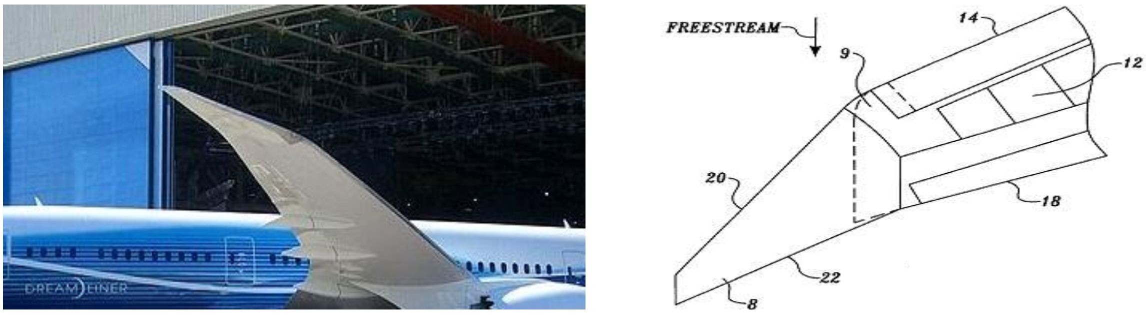


Figure 5 Raked wing tip[14]

Some of the conventional wingtips used in the aircrafts are mentioned in figure 6.

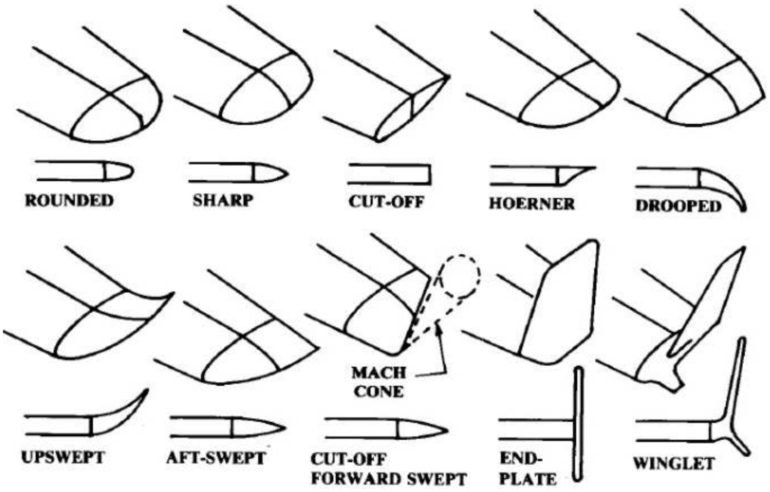


Figure 6 Different types of tip devices[15]

2.3. Advantages claimed on Winglets and tip devices

Ever since the winglet technology has been introduced, the advantages were being published. Dr.Whitcomb has performed an experiment with the winglet in which the winglet shows reduction in induced drag about 20%. In 1977, Heyson made an experiment to study the advantages of Whitcomb's winglet. His results indicate that winglets would reduce the induced drag more than tip extension and will be at its best when it is nearly vertical [16]. Later in 1980, R.T Jones made a research in winglets to determine its effect over the induced drag using Trefftz-plane theory and concluded that the vertical length of the winglet should be twice than the length of horizontal extension in order to have its gain over tip extension [2].

Aviation Partners Boeing announced that their APB blended winglet has saved more than 2 billion gallons of fuel in 2010. APB also added that the winglets could save 5 billion gallons of fuel by 2014 which also represents the total reduction in carbon emission. Indeed, APB blended winglet on B737 showed increased in range of about 5-7% due to overall reduction in drag. In case of spiroid tipped wing, API has made a flight test on Gulfstream II in 1993 and they achieved more than 10% of fuel efficiency during the cruise conditions[2]. Raked wingtip is a unique design for Boeing B777 family and it has improved the aircraft's performance by reducing the take-off field length, improved fuel efficiency and good climb performance. Raked wing tip could provide 2 % reduction in fuel burn which is compensated by 1.3 million of fuel saving per year and 3.9 million of carbon-di-oxide emission per year[3]. Sharklets is the recent invention from the Airbus Company for their A320 family. They claimed that sharklets would reduce fuel burn up to 3.4% and this corresponds to 700 tons of carbon emission per aircraft in a year. Airbus also added that A320 could lift off with more weight due to the performance of sharklets[4]. The research made with spiroid tipped wing indicates that it would disperse the vortex effects with in short span of time and therefore the time for take-off and landing between the aircrafts would be reduced [17].

Boeing Series advantages after using winglet (http://www.b737.org.uk/winglets.htm)		
Series	Range before	Range(Winglets)
B737-700	3250	3634
B737-800	2930	3060
B737-900	2670	2725

The above table shows that there was an increase in range when blended winglets were retrofitted.

[2] http://www.aviationpartners.com/
[3] http://www.boeing.com/commercial/777family/background/back4.html
[4] http://www.airbus.com/newsevents/news-events-single/detail/first-new-built-sharklet-equipped-a320-completed-in-toulouse/

Summarizing the advantages of winglets,

-Reduced induced drag
-Improved fuel efficiency
-Increased range and more payload
-Reduced noise effects due to vortex effects
-Reduced the amount of carbon emissions
-Helpful in air traffic control

2.4. Induced drag phenomena

In Aerodynamics, the four main forces which act on aircraft during the flight are Lift, Drag, Thrust and Weight. Drag is one of the most critical phenomena amongst all and is the opposing force of aircraft's forward motion. It could be classified briefly in to parasite drag (not due to lift) and lift induced drag [18]. In a civil transport aircraft, frictional drag and induced drag together contributes more than 80% of the total drag as represented in figure 8 [19], but the other forms of drag could not be excluded certainly.

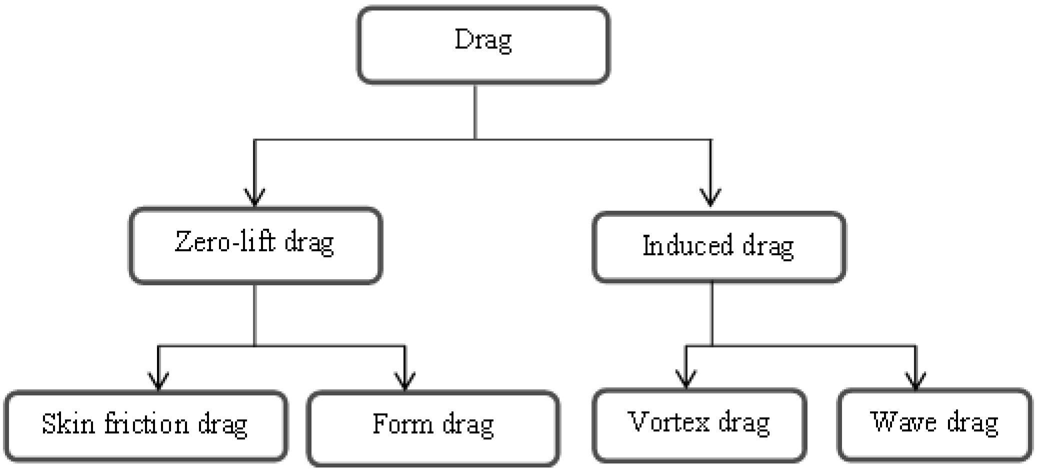


Figure 7Different forms of Drag

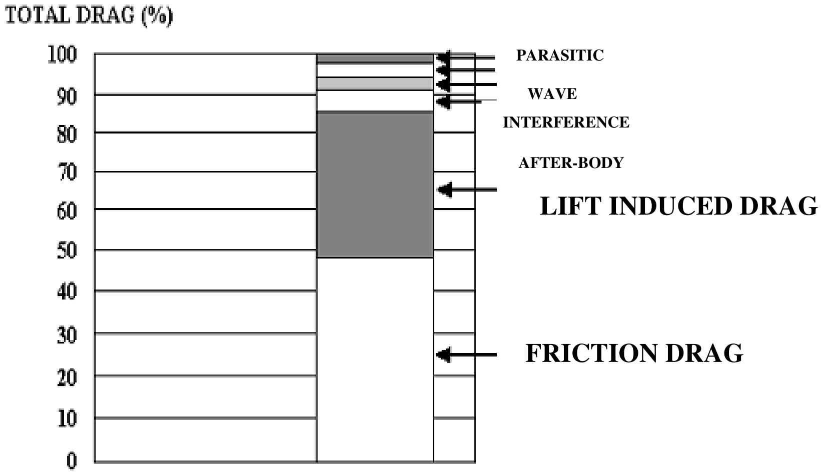
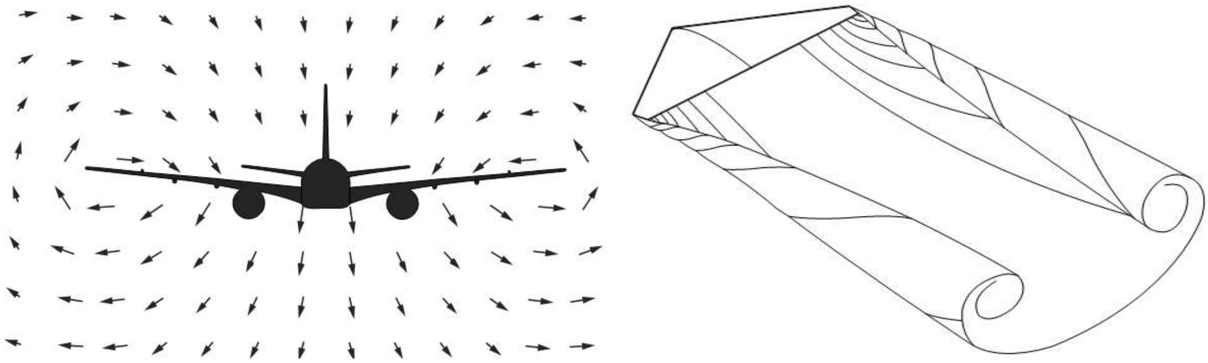


Figure 8 Distribution of drag components[19]

Induced drag produces kinetic energy which will cause the downward motion perpendicular to the airflow. This downward force could be recognized as the lift vector and this component is regarded as the induced drag [20]. Induced drag differs from the other forms of drag through a phenomenon of converting the dissipated kinetic energy into heat gradually. Vortex wake is a unique feature of induced drag. Dough Mclean has proposed the misinterpretations of the induced drag and the vortex wake produced by the wing [20]. Normally the vortex wake is produced from the flow pattern due to the difference in velocities at upper and lower surface of an aircraft. From the figure 9, it is shown that the velocities at the down surface move towards the upper surface and thus it create a circular flow pattern. This flow pattern is responsible for the vortex sheet that produced from the entire span of the wing (figure 10).



a *b*

Figure 9 (a)Flow pattern of the velocity[20] (b)Vortex sheet from trailing edge[20]

As mentioned by Doughlas, the common misunderstanding with the vortex sheet was that, the induced drag caused due to the theses vortices produced from the velocity flow pattern. Also, reducing the strength of vortex could not have a significant effect on the induced drag. Research has been conducted to reduce the induce drag and different methods have been developed to calculate this lift induced drag [1-3]. Using one of the proposed methods, induced drag for a commercial jet transport aircraft is calculated and will be discussed in upcoming section.

3. Methodology

3.1. Method for induced drag calculation

Airbus A320 aircraft's specifications are taken and the total drag of the aircraft is calculated using the following formula[18].

$$C_d = Cd0 + Cdi \qquad [1]$$

$$Cd0 = C_f\, FQ[S_{wet}/S_{ref}] \qquad [2]$$

$$C_f = [0.455]/[(\log Re_c)^{2.58}\ (1 + 0.144Mn^2)^{0.65}] \qquad [3]$$

$$Re_c = (Vl)/u \qquad [4]$$

$$F = (F^* - 1)\cos^2 \Lambda_{0.5c} + 1 \qquad [5]$$

$$F^* = 1 + 3.3\ (t/c) - 0.008\ (t/c)^2 + 27\ (t/c)^3 \qquad [6]$$

Where,

C_f -Skin friction drag coefficient
F-Form factor
Q-Interference factor
S_{wet}-Wetted area of the surface
S_{ref} -Wing area
Re_c = Reynolds number of the component
V = Velocity
l = component characteristics length
u = kinematic viscosity for that flight condition

$$C_{di} = KC_L^2 \qquad [7]$$

$$K = 1/\pi eAR \qquad [8]$$

e - Span efficiency or Ostwald's efficiency factor
AR – Aspect ratio
C_l – Lift coefficient,

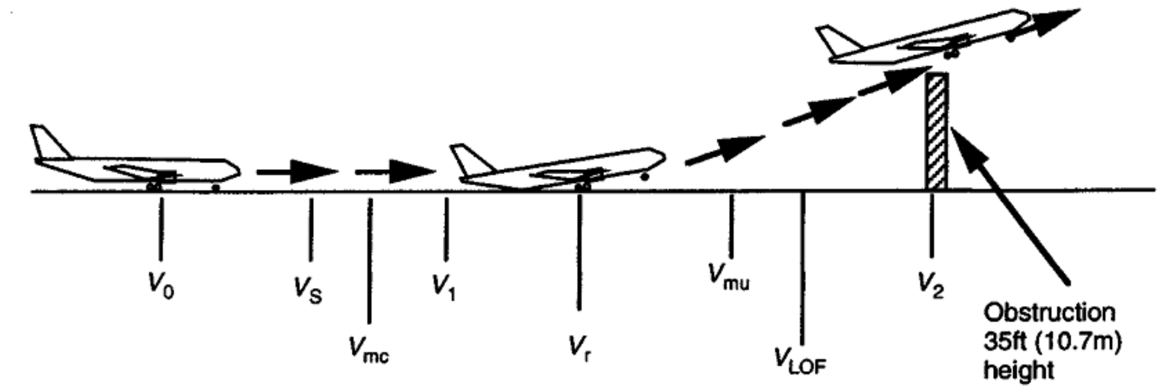
$$e = \{0.47 + (1/\sqrt{}AR)\} * (\cos\Lambda^{0.1}) \qquad [9]$$

Λ – Sweep angle.

Lift-induced drag due to vortex is based on the lift coefficient of the wing, aspect ratio and the span efficiency factor. The profile drag and induced drag are calculated from the above formula (from equation 1-9) with the values based on the aircraft's specification table (Appendix 1). The total drag is calculated for different altitudes and also for different velocities, starting from stall speed to cruise velocity. The total drag calculated will be used for determining the additional take-off weight for the same take-off distance.

3.2. Method for take-off performance calculation

The take-off distance of an aircraft is the total distance covered by an aircraft to clear the obstruction of height about 35 ft.



.

Figure 10Take-off of flight with different velocities

The above figure represents the standard take-off velocities that should be maintained during the take-off [18].

In this work, the motive is to find the parameters associated with the take-off performance of an aircraft. Airbus A320's specification is taken for the calculation and the take-off field length provided for the aircraft is 1500m.

There are three phases in the aircraft take-off mission i.e. Ground roll, Transition to climb and Climb. Normally, the ground roll, i.e. distance covered by the aircraft on the ground before climb, is calculated by resolving the forces horizontally and vertically. The approximation formula for the ground roll is given by[5],

$$S_G = \frac{1.44W^2}{g\rho C_{L,max}(T-D)} \qquad [10]$$

From the appendix 1,

W = 78000 kg,
$C_{L,max}$ = 2.1 (approximate)
T (during take-off) = 24494 kg
S = 123 sq.m(calculated from wing span)
g = 9.81 (acceleration due to gravity)
ρ = 1.225 kg/m^3 (density ground level)

Total surface area of wing is calculated from the root chord and span of main wing which are taken from the Appendix 1. A320 is operated with two CFM56-B engines where the take-off thrust for each engine is 23,500lbs. The total drag "D" in the above formula is calculated after

[5] http://adg.stanford.edu/aa241/performance/takeoff.html

substituting the values obtained from specification table. The total drag reduction is calculated from the total induced drag reduction for different velocities. Further, this total drag calculated with winglets is substituted in the ground distance formula in order to obtain the new total weight (W_{new}).
The simplified formula for the W_{new} is given by equation 11,

$$W_{new} = \frac{(T - D_{winglet}) * W^2}{T - D} \qquad [11]$$

The take-off weight value for the same take-off distance is calculated for different values of total drag.

3.3. CAD Methodology

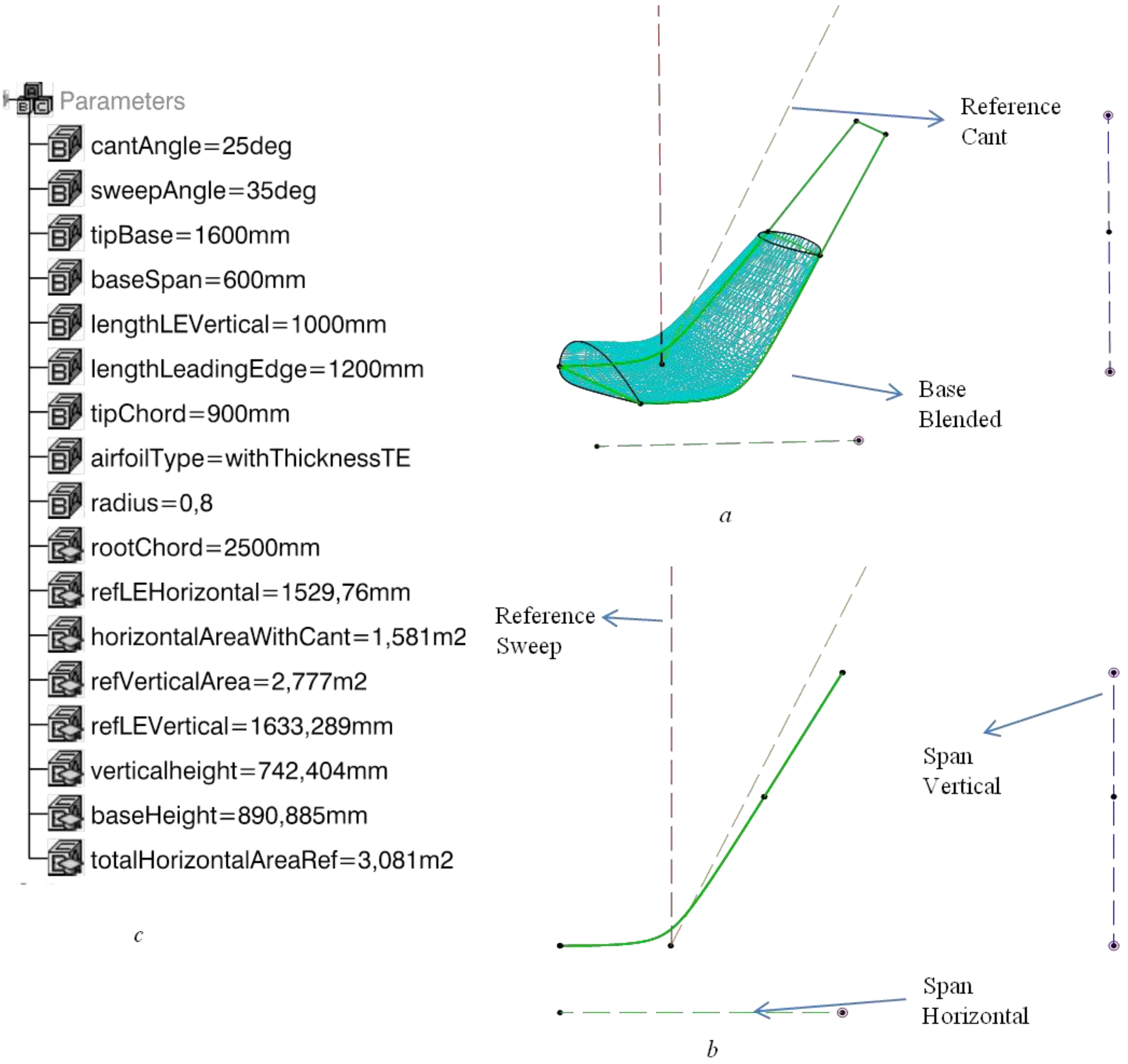
CATIA V5 R18 tool is used to develop the templates of different winglets and tip devices. Each winglet has been defined by a user feature where the user could choose between the types of winglet and modify the parameters associated with respective winglet model. A detailed description of the winglet templates are discussed in the following section.

3.3.1. Blended Winglet Template

Blended winglet template has been modeled with two different sections namely blended base and blended vertical. Blended base is a shape design model where it forms the intersection at the main wing tip chord and continues as a smooth curve to avoid the interference drag. The leading edge of blended base is formed by the conic section which depends on sweep plane, two tangent lines and two points which lies on the same plane. Radius of the leading edge could be varied by a real parameter and also it depends on base height associated with height parameter. The other parameters controlling the blended base were base span, cant angle, sweep angle and tip chord length (figure 11 c). Cant and Sweep angles are made with respect to the base span line and not from root chord of the base.

The later section i.e. the skeleton model of blended vertical is created in dependent with the blended base. The leading edge is made tangent with the base leading edge curve in order to obtain a smooth surface. Blended vertical is also associated with a set of parameters which are height and tip chord length (11 c).

Vertical and horizontal reference area of the blended template model are calculated and given as an output parameter (figure 11 c). The height parameter for both sections of blended template does not represent the vertical height but the length of respective lines. The exact vertical height for the blended template is given as an output parameter named “reference leading edge vertical” from which the reference area is calculated based on the root chord and tip chord of the respective section. Similarly, the horizontal span of entire blended section varies with the cant angle and also with base span length. The total projected span is calculated by measuring the length of projected leading edge line which shown as green dotted line in figure 11a and figure 11b.



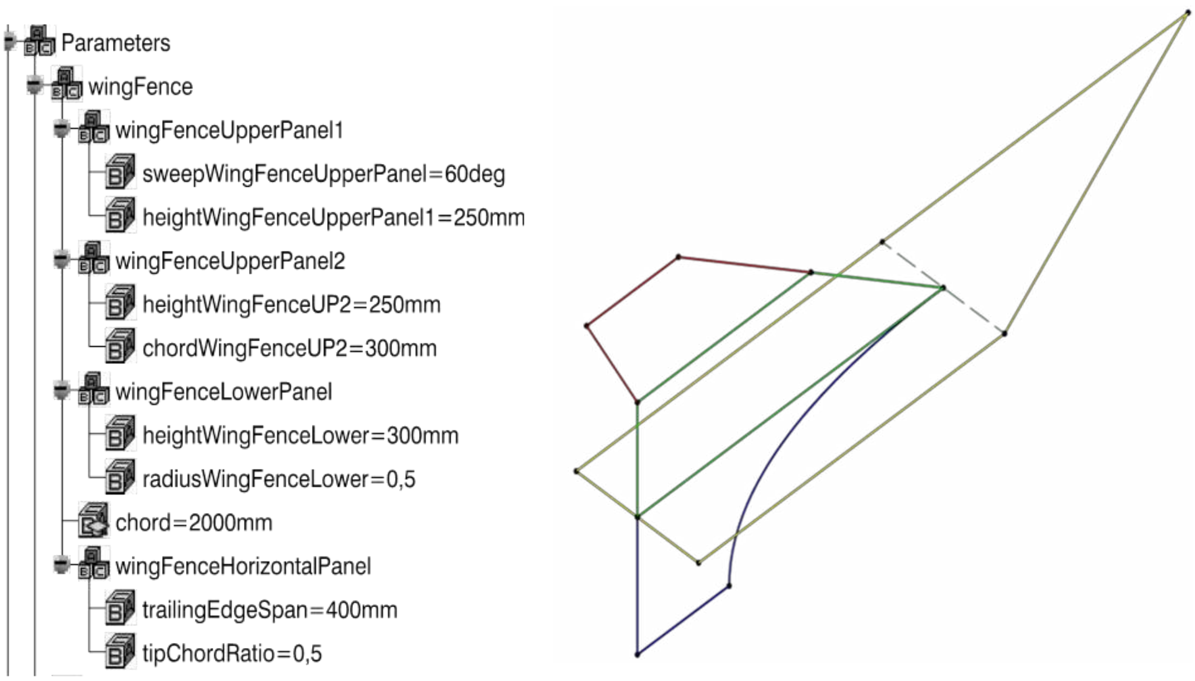
Parameters
- cantAngle=25deg
- sweepAngle=35deg
- tipBase=1600mm
- baseSpan=600mm
- lengthLEVertical=1000mm
- lengthLeadingEdge=1200mm
- tipChord=900mm
- airfoilType=withThicknessTE
- radius=0,8
- rootChord=2500mm
- refLEHorizontal=1529,76mm
- horizontalAreaWithCant=1,581m2
- refVerticalArea=2,777m2
- refLEVertical=1633,289mm
- verticalheight=742,404mm
- baseHeight=890,885mm
- totalHorizontalAreaRef=3,081m2

Figure 11 Blended Winglet Template: (a) Blended base and Vertical Isometric view; (b) front view; (c) parameters blended template

3.3.2. Wing Fence

The model of wing fence has been classified based on four different sections namely tip extension, wing fence upper 1, wing fence upper 2 and wing fence lower section. In figure 12b, the yellow lines represent the tip extension model. Trailing edge of tip extension starts from the trailing edge point on tip chord of main wing. Taper ratio parameter controls the tip chord of tip extension. The span of trailing edge could be varied by a length parameter. The line connecting the points on tip chord of main wing and on tip chord of extension is the leading edge. It is made with respect to the taper ratio and the trailing edge span length which have been shown in the figure 13a The tip extension acts as the support for the other three sections, where the root chord is same for both wing fence upper 1 and wing fence lower, which lies exactly at midway between tip chord of main wing and tip extension.

The green lines showed in figure 12b makes the wing fence upper 1. Leading edge curve has been given with an angle of rotation with respect to the green dotted line where the angle of rotation was decided by a sweep parameter (figure 12). In this case, the tip chord of upper panel 1 section depends on the sweep angle and the height parameter associated with wing fence upper panel 1 section. Next section starts with continuation of panel 1, where tip chord of panel 1 acts as root chord of the panel 2 section. The leading edge again made tangent with previous leading edge as shown in red lines in figure 13b. Furthermore, tip chord length and height are the parameters used to modify the upper panel 2 section.



```
Parameters
  wingFence
    wingFenceUpperPanel1
      sweepWingFenceUpperPanel=60deg
      heightWingFenceUpperPanel1=250mm
    wingFenceUpperPanel2
      heightWingFenceUP2=250mm
      chordWingFenceUP2=300mm
    wingFenceLowerPanel
      heightWingFenceLower=300mm
      radiusWingFenceLower=0,5
    chord=2000mm
    wingFenceHorizontalPanel
      trailingEdgeSpan=400mm
      tipChordRatio=0,5
```

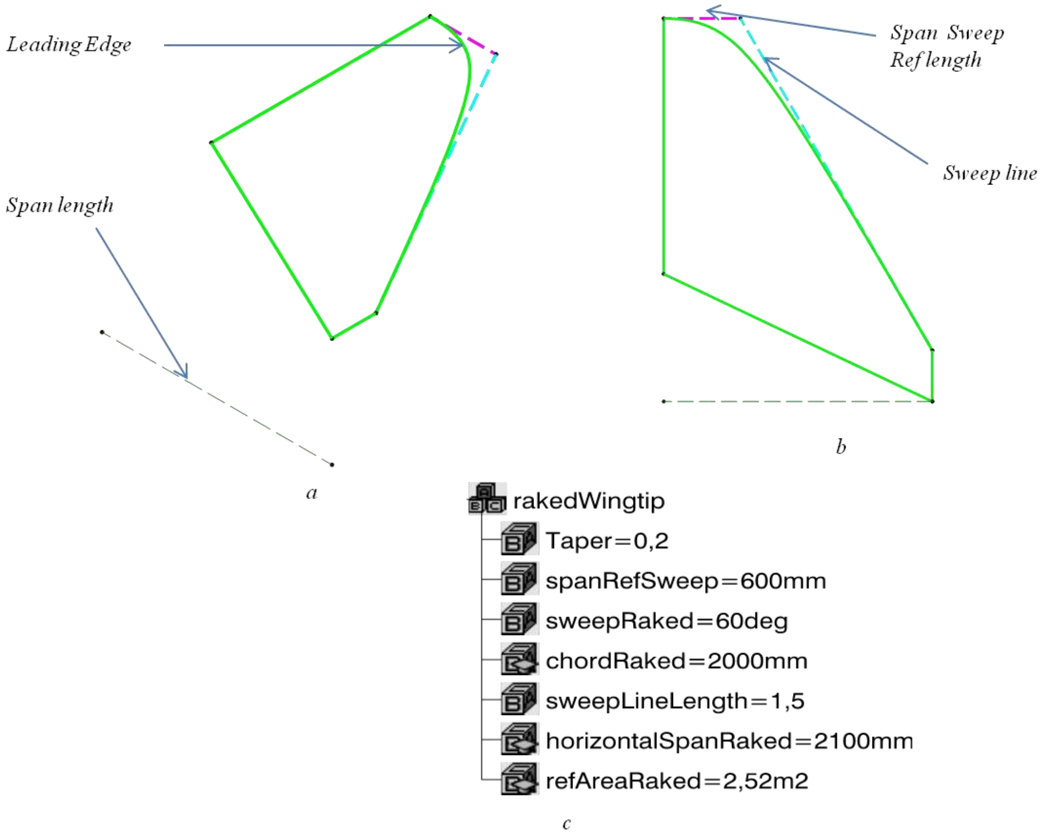
a *b*

Figure 12 Wing Fence: (a) wing fence parameters; (b) Skeleton model of wing fence

The final section, wing fence lower panel is represented by the blue lines in figure 12b. As mentioned earlier in this section, the root chord of the upper panel acts as the root chord for lower panel. Here the leading edge is made by a conic section, where the radius could be modified by a radius parameter.

3.3.3. Raked Wingtip

Raked wing tip as mentioned earlier, the leading edge of the tip section has greater sweep angle than the leading edge of main wing. In figure 13a and 13b, leading edge of raked wing tip has been made in dependent with sweep reference line and sweep line. The sweep reference line shown in figure 13 b represent the angle of sweep controlled by an angle parameter named sweep raked. Moreover the tip chord of raked wing tip depends on taper ratio parameter. Also, the real parameter sweep line length along with the tip chord of main wing controls the span of sweep line. The actual horizontal length of raked wing tip is given as an output parameter named “horizontal span raked” from the projected line. The parameters associated with raked wing tip were shown in figure 13c.



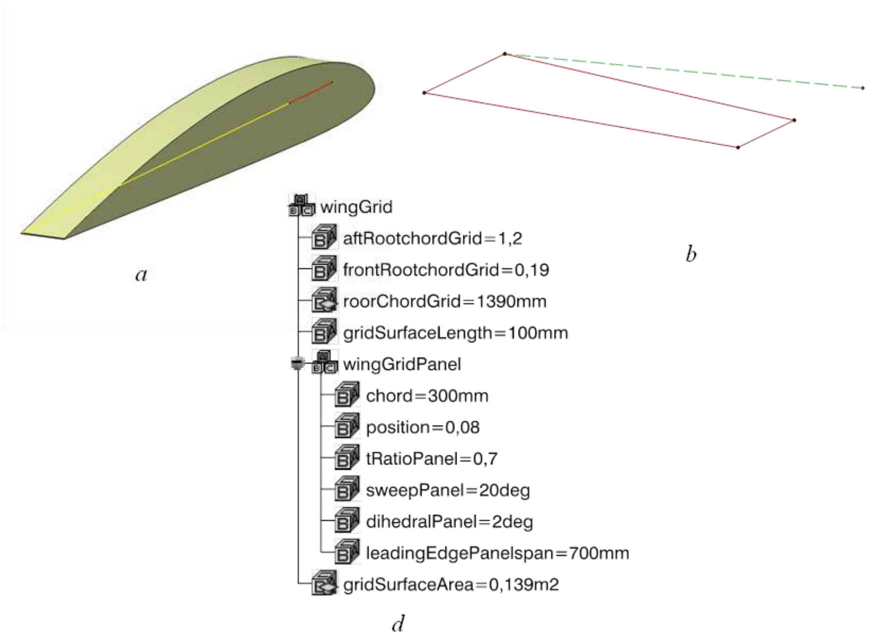
rakedWingtip
Taper=0,2
spanRefSweep=600mm
sweepRaked=60deg
chordRaked=2000mm
sweepLineLength=1,5
horizontalSpanRaked=2100mm
refAreaRaked=2,52m2

Figure 13 Raked wing tip: (a) Isometric view; (b) Top View; (c) raked wing tip parameters

3.3.4. Wing Grid

The template of wing grid has two sections namely wing grid root and grid Panel. In case of wing grid root, the front root chord (red line) and aft root chord (yellow line) of the grid were made according with the tip chord span of main wing (figure 14a). Two real parameters namely aft root chord grid and front root chord grid decide the position and length of two root chords of grid. An airfoil is placed which has a common chord length comes from the total length of two chords mentioned earlier. Then a surface is developed in the “y” component direction, where the length could be varied by a length parameter “grid surface length”. The surface made will be acting as the platform for the grid panels which are to be created.

Upon deciding the root chord length of grid, the grid panels (red lines) are created on them. The position of grid panel on the above mentioned chord depends on the real parameter, “position”. The geometry of grid panel is almost similar to the main wing. The sweep for the leading edge is made with respective to the sweep reference line, a dotted line shown in figure 14b.



wingGrid
aftRootchordGrid=1,2
frontRootchordGrid=0,19
roorChordGrid=1390mm
gridSurfaceLength=100mm
wingGridPanel
chord=300mm
position=0,08
tRatioPanel=0,7
sweepPanel=20deg
dihedralPanel=2deg
leadingEdgePanelspan=700mm
gridSurfaceArea=0,139m2

Figure 14 Wing Grid Template: (a) wing grid panel; (b) wing grid root; (c) wing grid parameters

Similar to previous templates, the length of “panel leading edge” is controlled by a length parameter “leading Edge Panel Span”. The other parameters which decide the panel geometry are shown in the figure 14 c. Moreover, the number of panels the user needs to place on the root chord grid is depend on a global parameter, “no of grid panels”, which will be discussed in automation section.

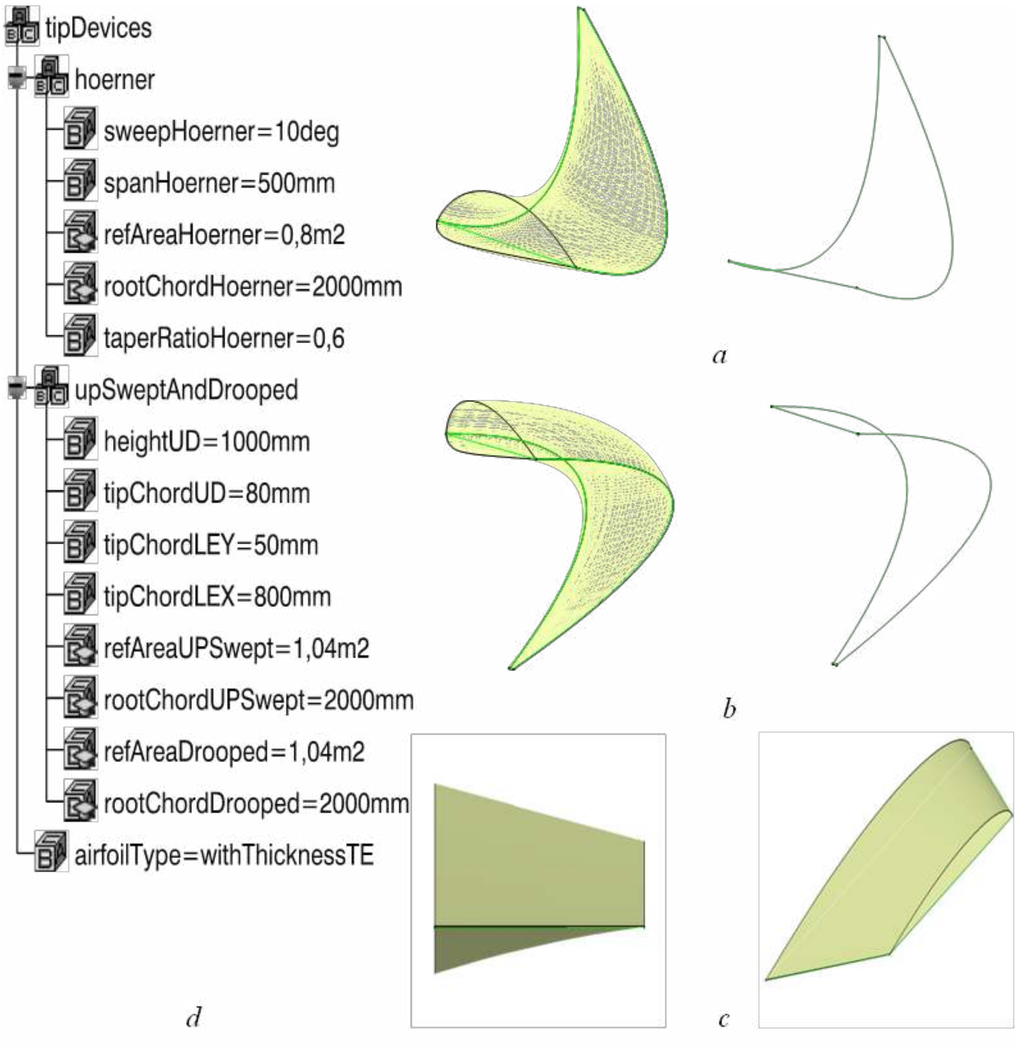
3.3.5. Tip Devices

A large number of tip devices had been developed so far in aircraft industry and some of the tip devices were shown in earlier sections (figure 6). The three tip devices which modeled in this work are,

1. Upswept
2. Drooped
3. Hoerner

The design of Upswept (figure 15 a) and Drooped (figure 15 b) are almost similar but only differs with the height parameter. In case of upswept, the height will be in positive direction whereas in drooped, it will be in negative direction with the same value. The leading and trailing edges of both surfaces were tangent to leading and trailing edges of main wing respectively. The position of the tip chord in X and Y coordinates could be changed by the two length parameters as showed in figure 15d.

The other parameters that decide the upswept and drooped tip devices are shown in figure 15d under their respective names.



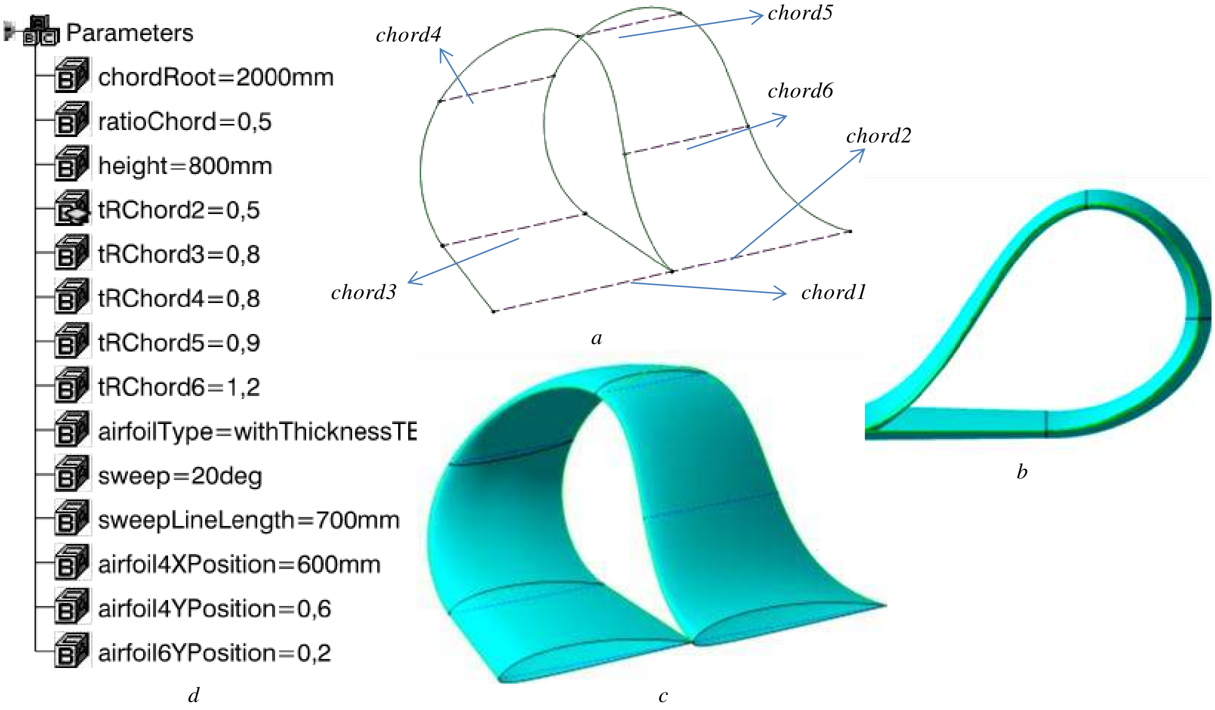
tipDevices
hoerner
sweepHoerner=10deg
spanHoerner=500mm
refAreaHoerner=0,8m2
rootChordHoerner=2000mm
taperRatioHoerner=0,6
upSweptAndDrooped
heightUD=1000mm
tipChordUD=80mm
tipChordLEY=50mm
tipChordLEX=800mm
refAreaUPSwept=1,04m2
rootChordUPSwept=2000mm
refAreaDrooped=1,04m2
rootChordDrooped=2000mm
airfoilType=withThicknessTE

Figure 15 Tip Devices: (a) Upswept; (b) Drooped; (c) Hoerner; (d) Tip device parameters

When defining Hoerner geometry, one of the two airfoils would not have the bottom surface which is a major design criterion (figure 15 c). The span, sweep angle of leading edge and taper ratio are the parameters deciding Hoerner geometry.

3.3.6. Spiroid Tipped Wing

Spiroid model has six chords to make the geometry more flexible. Also, the length of chord 1 and chord 2 depends on the ratio chord parameter. Similarly, length of chord depends on taper ratio with respect to the chord 1 length and successive chord length depends on the taper ratio with respect to previous chord. Position of second chord could be varied in X and Y direction with a length and real parameter respectively. Figure 16b shows the front view of spiroid model, where the height and chord 4 position decide the shape of model. Leading edge of the model starts from chord 1 and passes through other chords and ends at chord 2, thus forming a loop (figure 16 a).



Parameters
- chordRoot=2000mm
- ratioChord=0,5
- height=800mm
- tRChord2=0,5
- tRChord3=0,8
- tRChord4=0,8
- tRChord5=0,9
- tRChord6=1,2
- airfoilType=withThicknessTE
- sweep=20deg
- sweepLineLength=700mm
- airfoil4XPosition=600mm
- airfoil4YPosition=0,6
- airfoil6YPosition=0,2

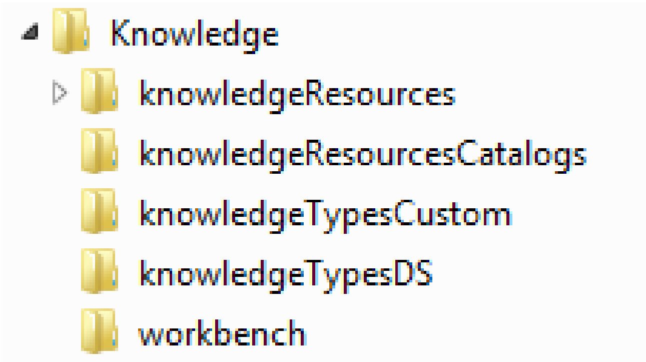
Figure 16 Spiroid Tipped Wing: (a) Skeleton model showing the chords of the airfoil; (b) Front view of the shape design; (c) Isometric view of Spiroid tipped wing

The other parameters which control the shape of spiroid model are represented in the above figure (figure 16d).

3.4. Automation of Winglets and Tip devices

Once the geometry of each winglet and tip devices have been modeled, automation of these models are done. In order to automate, the first step would be creating a UDF template for each parameterized model. Every winglet has their own inputs and corresponding outputs with respect to the inputs given for panel instantiation.

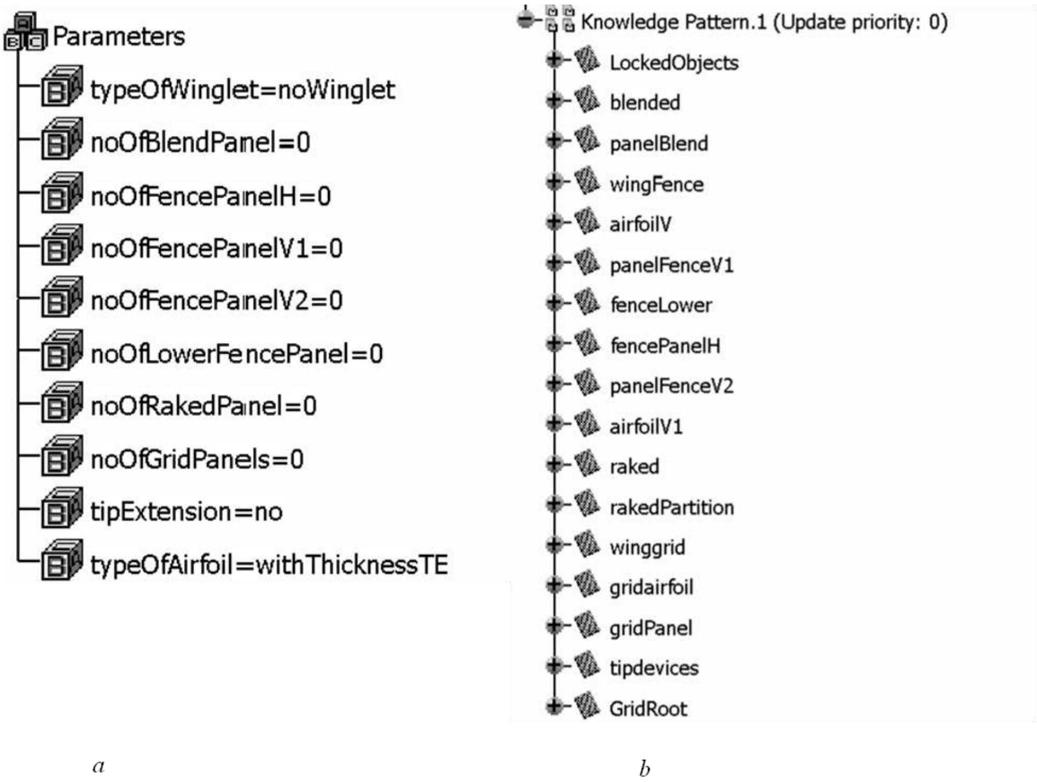
The next step in automation is done by implementing KP to the previous made UDFs templates. A catalog document named "Winglet Catalog" is created which includes all the UDFs of winglets and tip devices. A typical knowledge folder where all the UDFs and the catalog document should be saved is shown in figure 17. All UDFs should be saved inside the "knowledgeResources" folder and the catalogs should be saved in "knowledgeResourcesCatalogs" folder. A set of global parameters are created in order to control/guide the KP program which are described in figure 18a.



Knowledge
knowledgeResources
knowledgeResourcesCatalogs
knowledgeTypesCustom
knowledgeTypesDS
workbench

Figure 17 Knowledge Folder for KP

Also a KP lists has been created where the instantiated outputs stored and the model of the lists is showed in figure 18b. The outputs stored in these lists would act as input for the next instantiation.



Parameters
typeOfWinglet=noWinglet
noOfBlendPanel=0
noOfFencePanelH=0
noOfFencePanelV1=0
noOfFencePanelV2=0
noOfLowerFencePanel=0
noOfRakedPanel=0
noOfGridPanels=0
tipExtension=no
typeOfAirfoil=withThicknessTE

Knowledge Pattern.1 (Update priority: 0)
LockedObjects
blended
panelBlend
wingFence
airfoilV
panelFenceV1
fenceLower
fencePanelH
panelFenceV2
airfoilV1
raked
rakedPartition
winggrid
gridairfoil
gridPanel
tipdevices
GridRoot

a *b*

Figure 18 Knowledge Pattern: (a) Global Parameters for automation; (b) Lists created for automation

The structure of automation process is shown in figure 19. From the global parameters, the user could choose between different types of winglet. If the user has chosen the blended winglet, the base surface along with the skeleton model of blended vertical will be instantiated. In case, if the type of winglet is wing grid, then only the grid surface will be instantiated. Further, the number of grid panels to be instantiated is determined by the global parameter, "no of grid panels". The position of the first panel is depends on the leading edge point of root chord grid (Ref section 3.3.4). But the successive panels get the position with respect to trailing edge point on chord of previous panel.

In all the case of repetition, if number of instantiation is "N" number of times, then (N-1)th output will serve as an input for Nth instantiation. Also the outputs of respective winglet part are stored in the respective lists as previously mentioned.

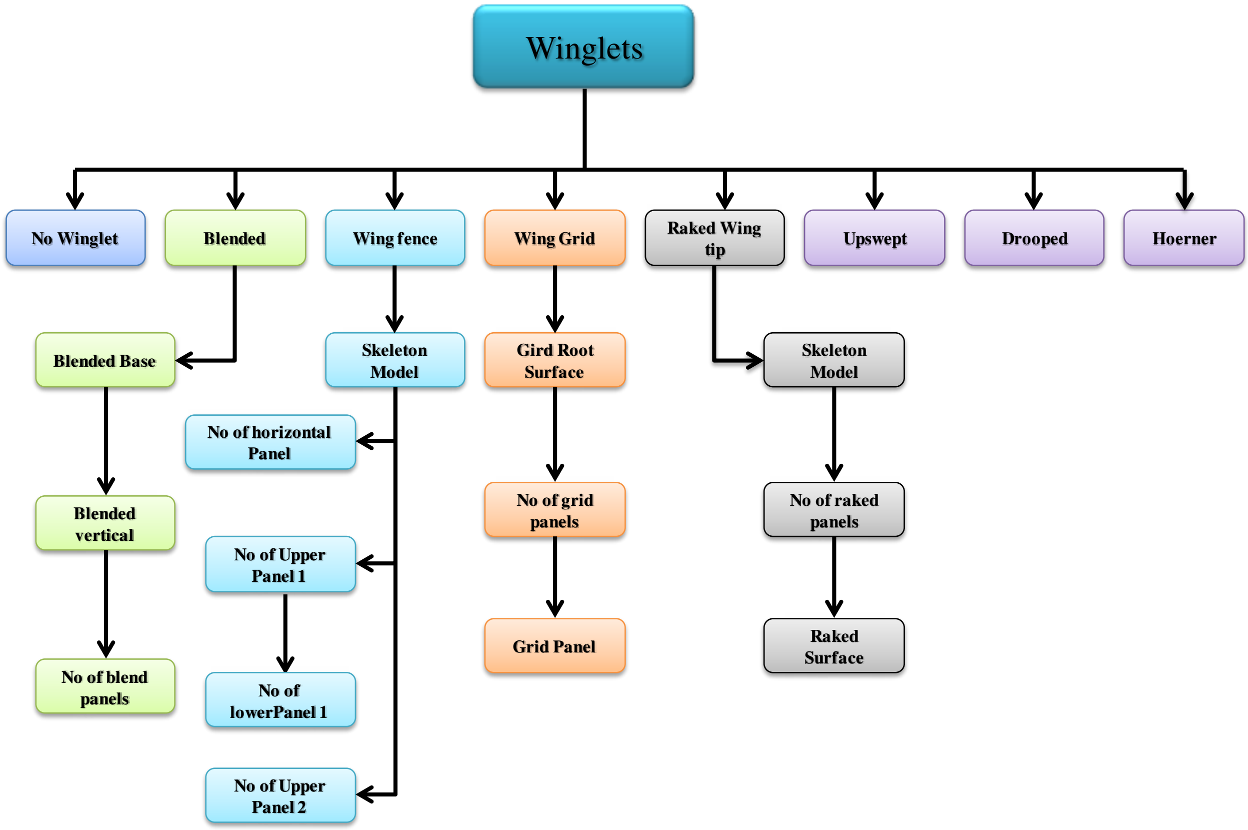


Figure 19 Automation Framework for Winglets

All types of winglet have a repetition process except for the tip devices. If the user needs a tip device to be at the wing end section, the whole surface model would be instantiated and user defined parameters only could be modified to fix the geometry.

4. Results and Discussion

4.1. Induced drag contribution

The induced drag and the total drag are calculated for different velocities. Based on the calculation, the percentage of induced drag in total drag was calculated. The results are plotted against the velocity which could be seen in the figure 20. From the graph shown in figure 20, the contribution of the induced drag in total drag is found to be more than 80% when the velocity is given around 60m/s. As the velocity increases the percentage of induced drag is observed to be decrease and about less than 20% at the velocity, 250 m/s. The result shows that the effect of induced drag would be much higher during the take-off condition and lesser effect during cruise condition.

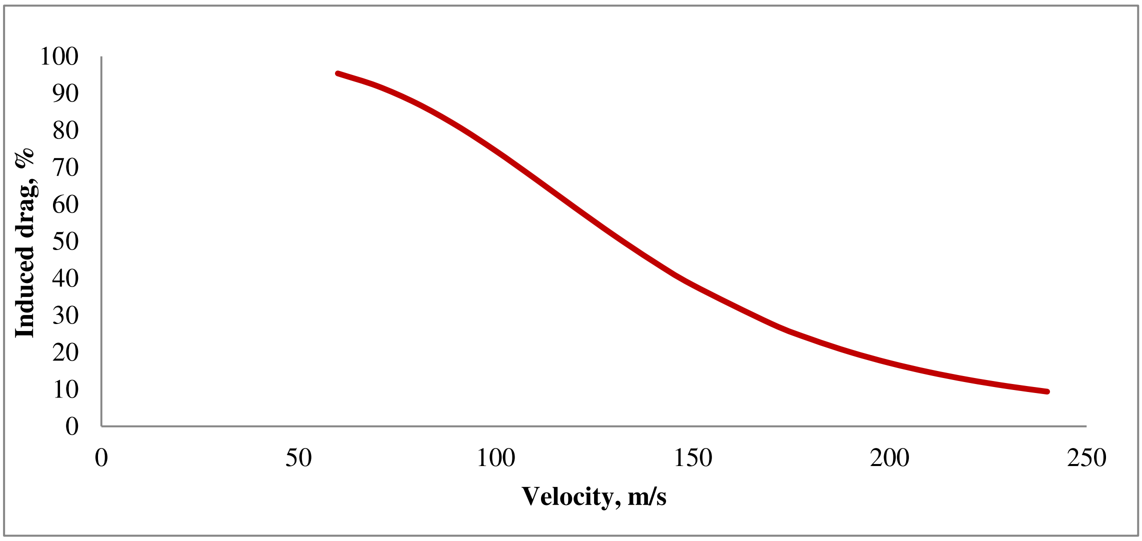


Figure 20 Induced drag percentage of the aircraft against the velocity, m/s

4.2. Performance analysis on Aircraft Take-off

The total drag of A320 aircraft's wing which comprises of parasite drag and the lift induced drag is calculated with and without winglets. The induced drag for both wing with and without winglets are plotted against the velocity. It could be seen from the figure 21, decrement in drag for the aircraft with winglets is found to be less but of considerable amount. The total drag reduction observed from the calculation is about 8-10% during the high lift coefficient conditions where the role of induced drag is very high. The aspect ratio plays a significant role in induced drag as it is the main factor which is responsible for the span efficiency. In this case, the wing plan form without winglet has its aspect ratio of 9 whereas the other with winglet has 11, which is one of the main causes for the induced drag reduction.

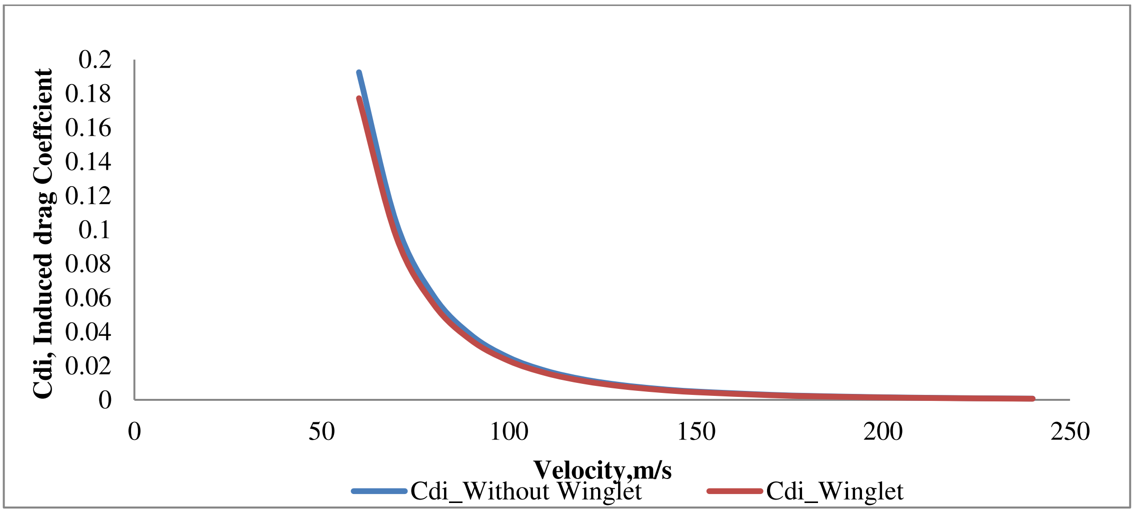


Figure 21 Induced drag coefficient, C_{di} for wing with winglet and without winglet against the velocity of the aircraft, V m/s

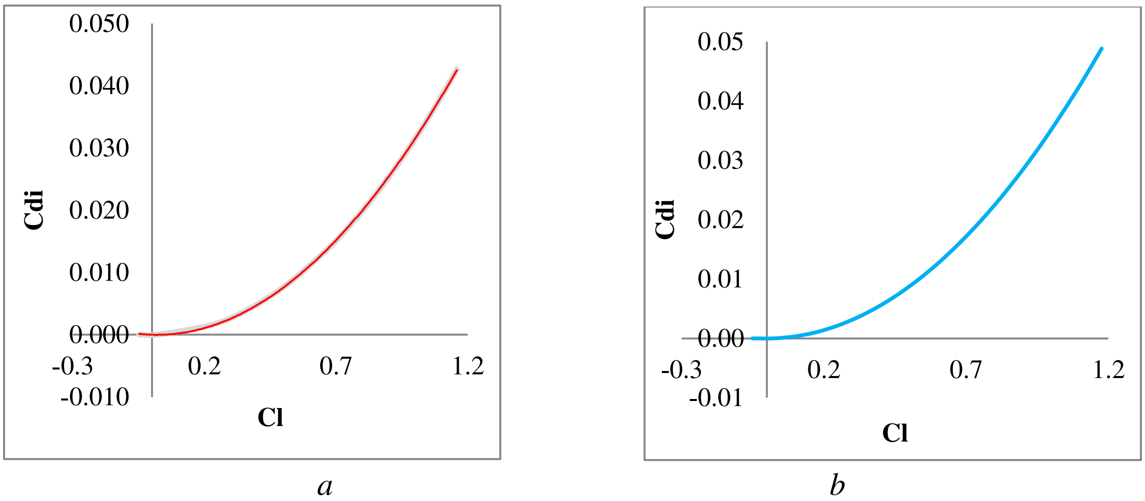


Figure 22Induced drag coefficient against coefficient of lift: (a) with winglets; (b) without winglets

The result from the Tornado analysis is observed and the induced drag values are plotted against the lift coefficient. From the figure 22 (a&b), the induced drag is very less at $C_l = 0.2$ and increases proportionally with C_l. As discussed in section 4.1, the risk of induced drag is high at high lift coefficient and has less effect at low C_l. Also, the induced drag polar of both wing configurations seems to be similar, but the values shows that the induced drag with winglet configuration was 5-6% lesser when compared with without winglet configuration.

The total drag is also calculated at different altitude and the results are plotted against the velocity. The total drag coefficient C_d, at an altitude of 12000 meters is about less than 0.05 at cruise velocity (figure 23). This is due to the fact that only parasite drag counts more than 80% during this flight condition and the induced drag contribution would be very less. But during the ground level operation it could be seen that the drag counts higher at initial take off velocities where the induced drag plays a significant role as majority of the drag comes from it (figure 23).

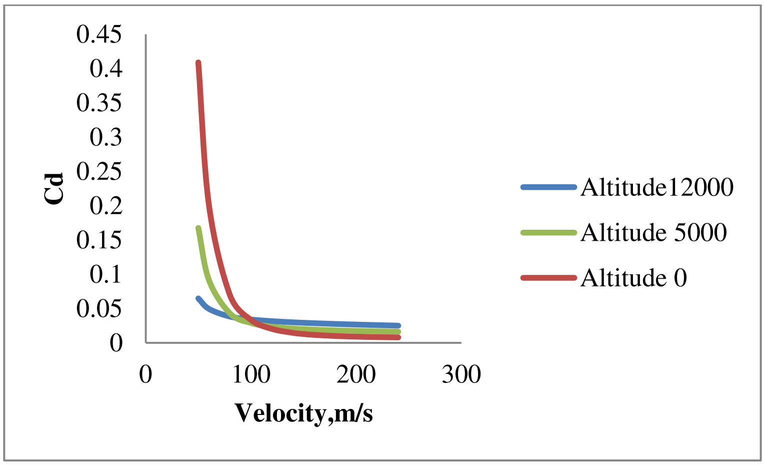


Figure 23Total drag coefficients C_d for different altitudes against the velocity of the aircraft V m/s.

As the amount of drag reduced due to the installation of winglet has been calculated, a performance advantage of this induced drag reduction is identified. Every drag reduction counts for the reduction in the amount of fuel burned for the total mission. Also, this drag reduction is responsible for the take-off field length for an aircraft. If the total drag is reduced then the aircraft could take off within the designed field length. But in this case the total amount of payload that could be increased for the drag reduction is calculated provided with the same take off field length. The results shows that for every percent reduction in drag the airlines could have more additional weight proportionally (figure 24). The aircraft could carry an additional weight of about 600-800 kg for 5-6% of total drag reduction.

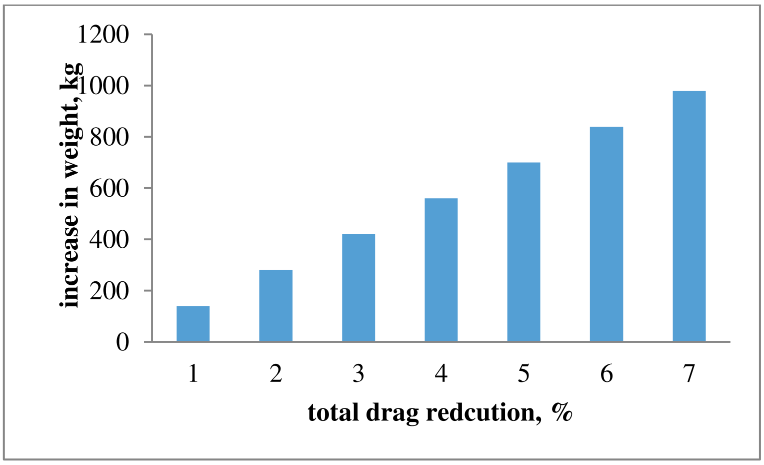


Figure 24 Increased Payload for the corresponding percentage reduction in drag

The additional take-off weight achieved due to winglets will be beneficial for the airline operators as they could carry extra passengers/payload.

4.3. Parameterized Winglet models

KP has been applied to the winglet templates for the automation process and the results have been discussed in this section.

As the input parameter has been given as “no winglet” in the global parameter set, there is no tip device attached to the end section of main wing as shown in figure 25.

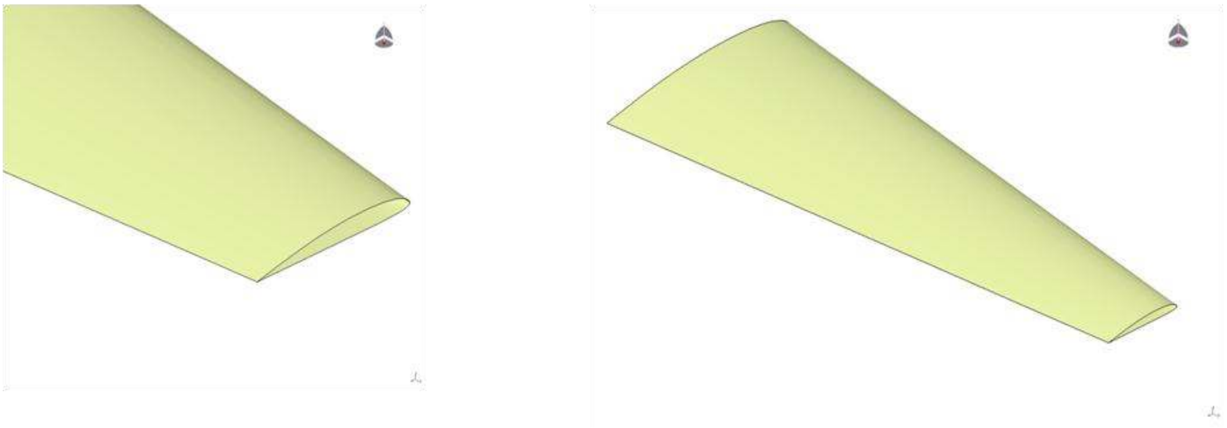


Figure 25 No winglet or tip devices at end section

Now, the input has changed to “blended” and so the surface model of blended base is instantiated. Moreover if the user needs to be placed upon the base, the number of panels needed should be given in the input field “no of blend panel”. The instantiated model is shown in figure 26b as there are two panels placed on the base section. On the other hand, figure 27 shows the blended model without a panel could be another option for the user.

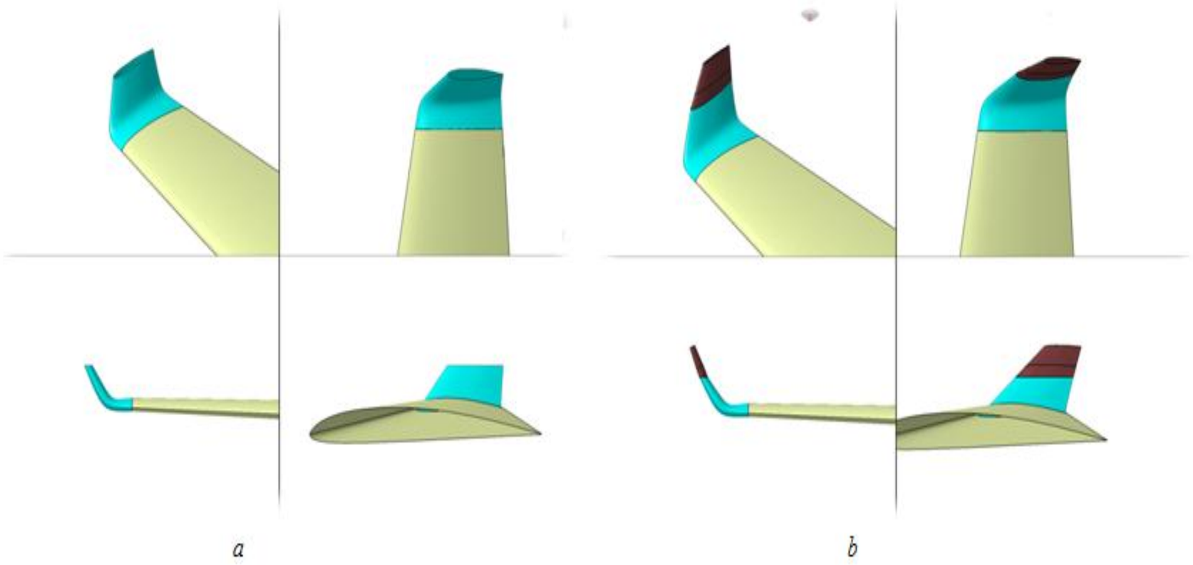


Figure 26 Blended Winglet: (a) Blended base; the base Panels placed upon base

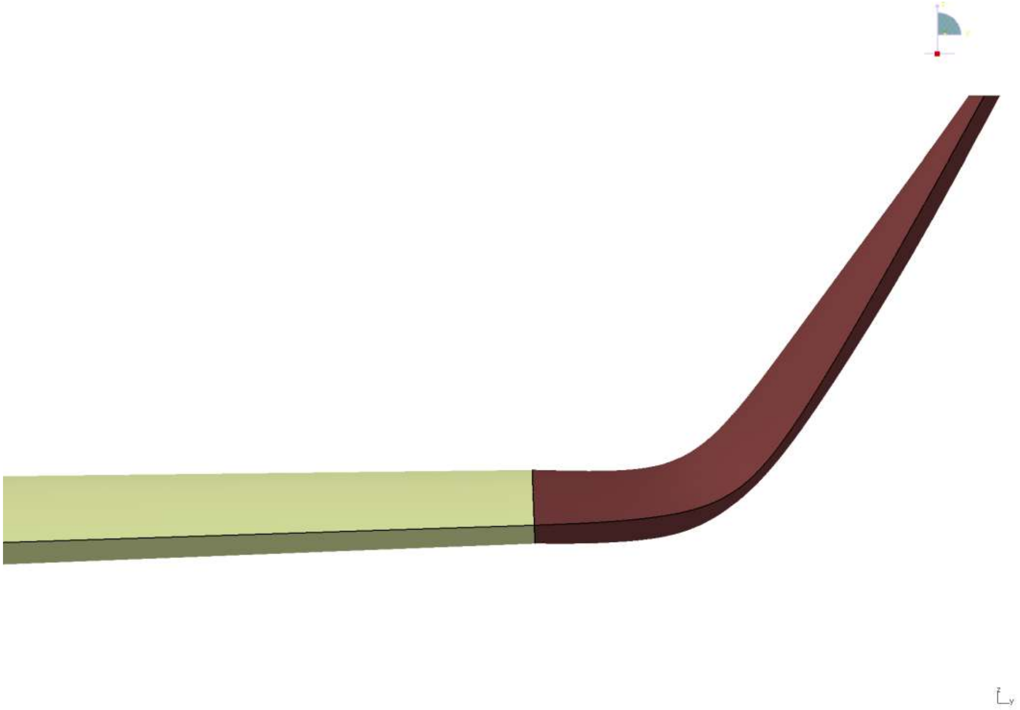


Figure 27 Blended winglet without panel

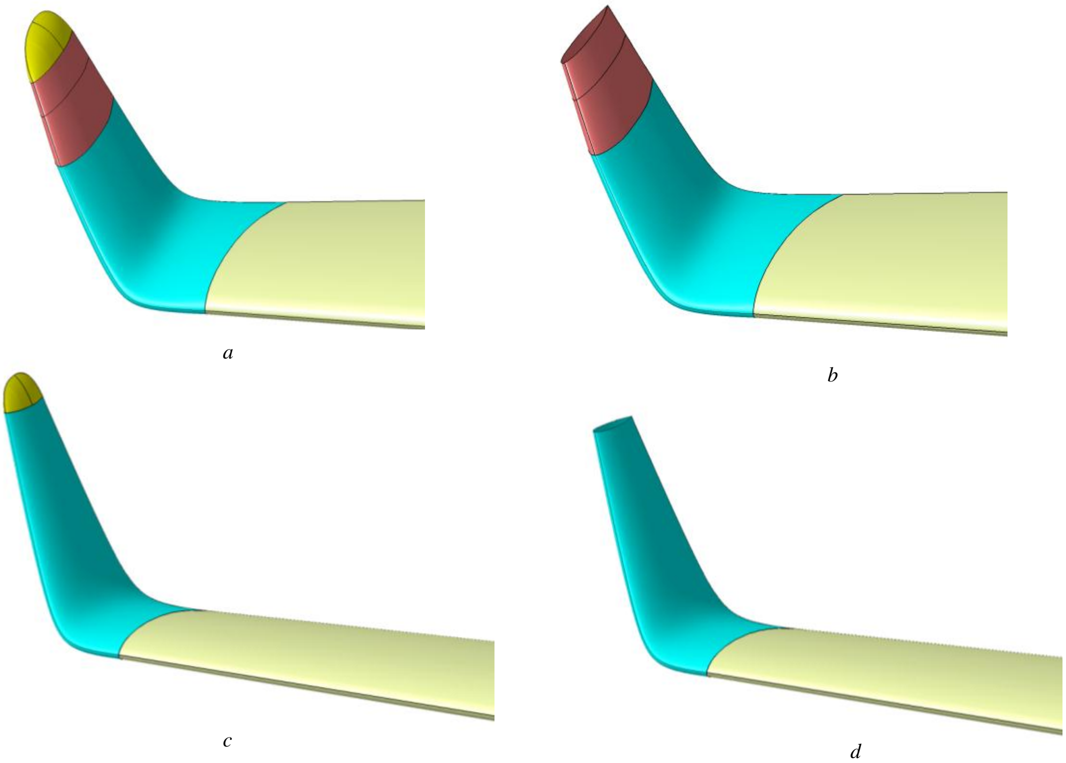


Figure 28 Blended Winglet: (a) With tip extension and two panels; (b) No tip extension; (c) With tip extension no panel; (d) No tip extension and panel

Upon changing the type of winglet parameter to wing fence, the skeleton model is generated. Figure 29a represents that horizontal panels have been instantiated. When the number of fence panel vertical for upper section 1 was chosen to be 2, green panels are instantiated as shown in figure 29b. But for the lower panel to be instantiated, the number of upper panels should be greater than 1 as lower panel depends on the upper panel's airfoil.

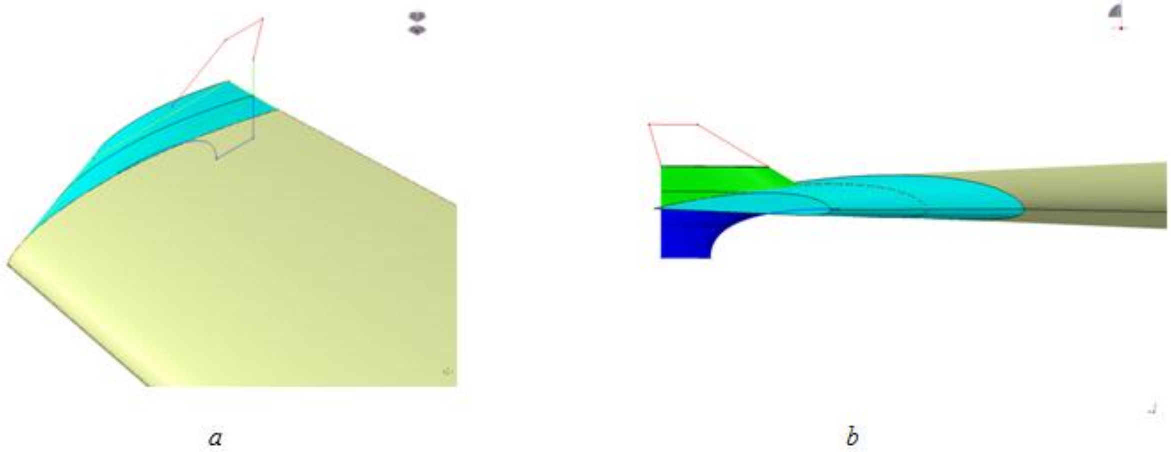


Figure 29 Wing Fence: (a) Skeleton model with horizontal panels; (b) With Lower and Upper Panels

When all the parameter associated with panel instantiation for wing fence are given, the model generated with all panel surface is shown in figure 30.

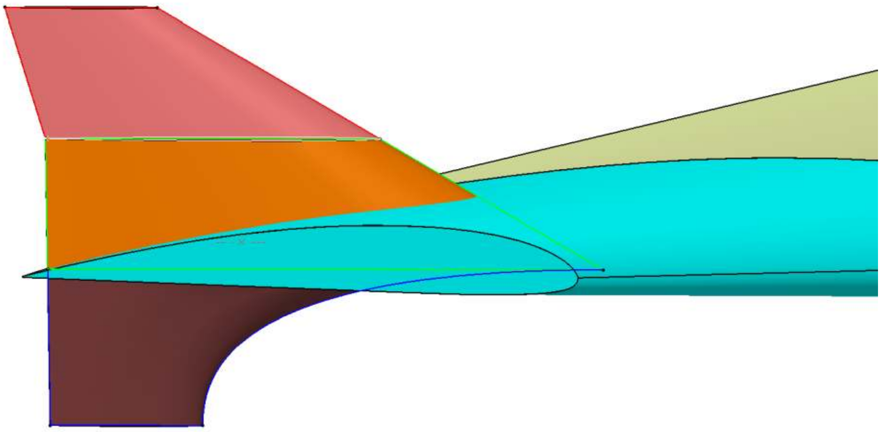


Figure 30 Wing Fence after KP

Similarly, the raked wing tip model shown below (figure 31) has three panels, where the leading edge of the raked wing tip made tangent to the leading edge of the main wing.

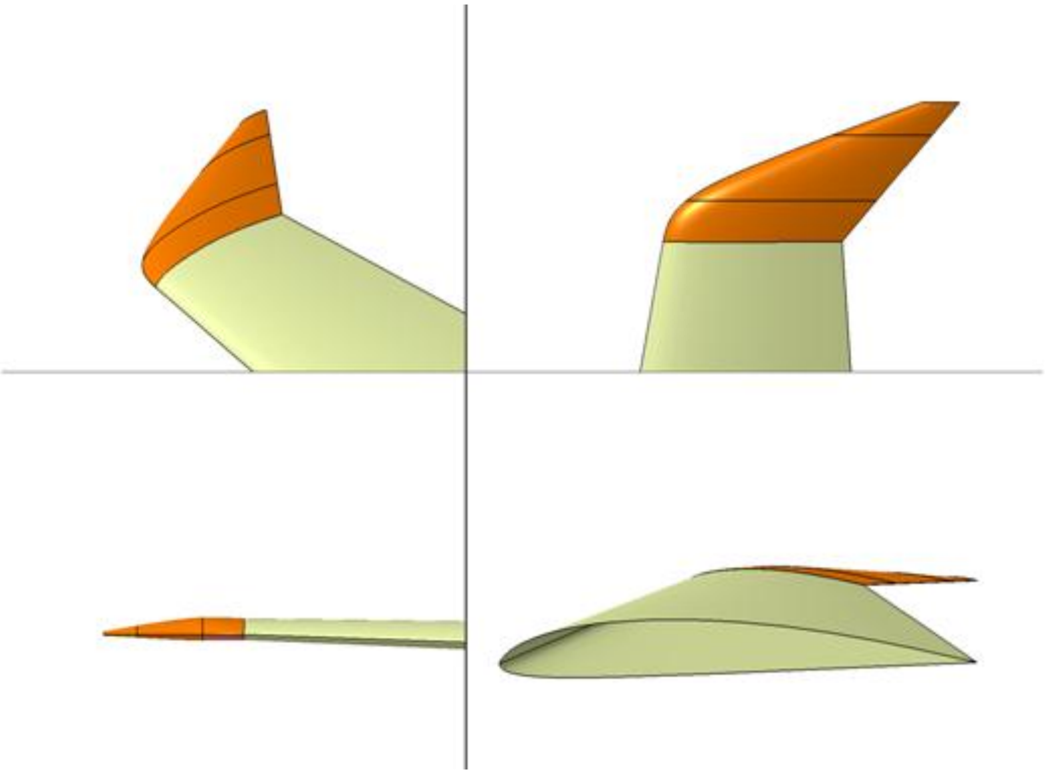


Figure 31 Raked wing tip with three panels

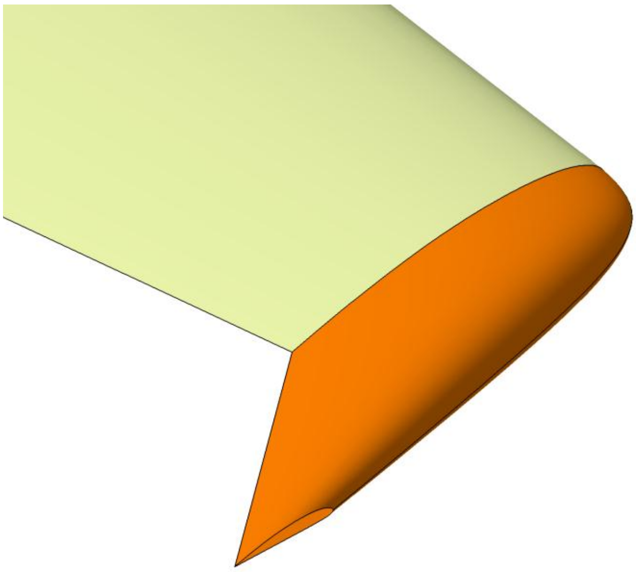


Figure 32 Raked wingtip with one panel

As mentioned in automation section, when the type of winglet becomes wing grid, only the grid root is instantiated (figure 33 a). Then the number of panels that should be placed on the grid root is provided in the global parameter. In figure 33b, two panels are instantiated where the distance between each panel was decided by position parameter which was discussed in earlier sections. Additionally, the sweep angle and dihedral angle is also changed which could be seen in the following figure (figure 33b).

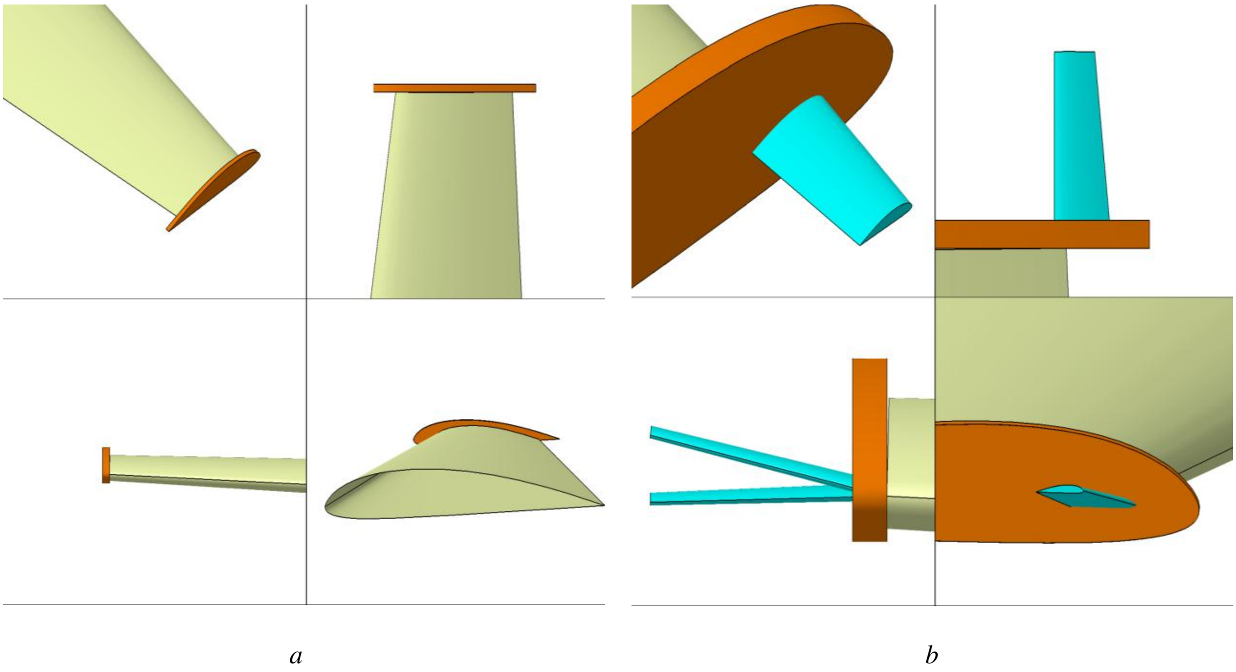


Figure 33 Wing grid: (a) Only Grid root; (b) Panels at different dihedral and Sweep angle

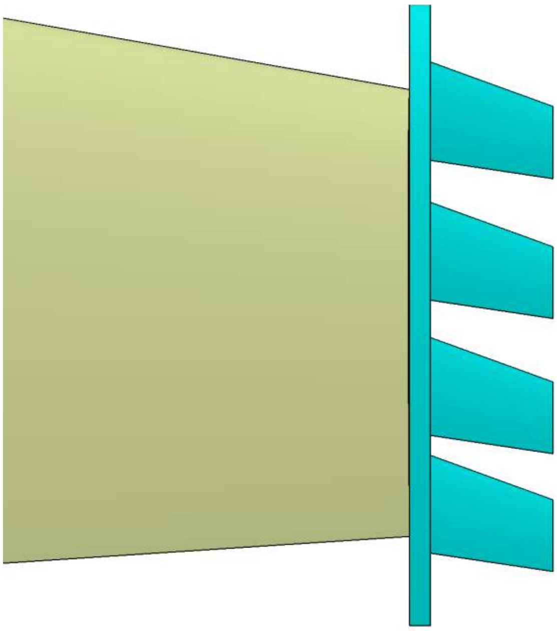


Figure 34 Wing Grid with 4 panels

Finally, the tip devices that are instantiated upon deciding the type of winglet are shown in following figure.

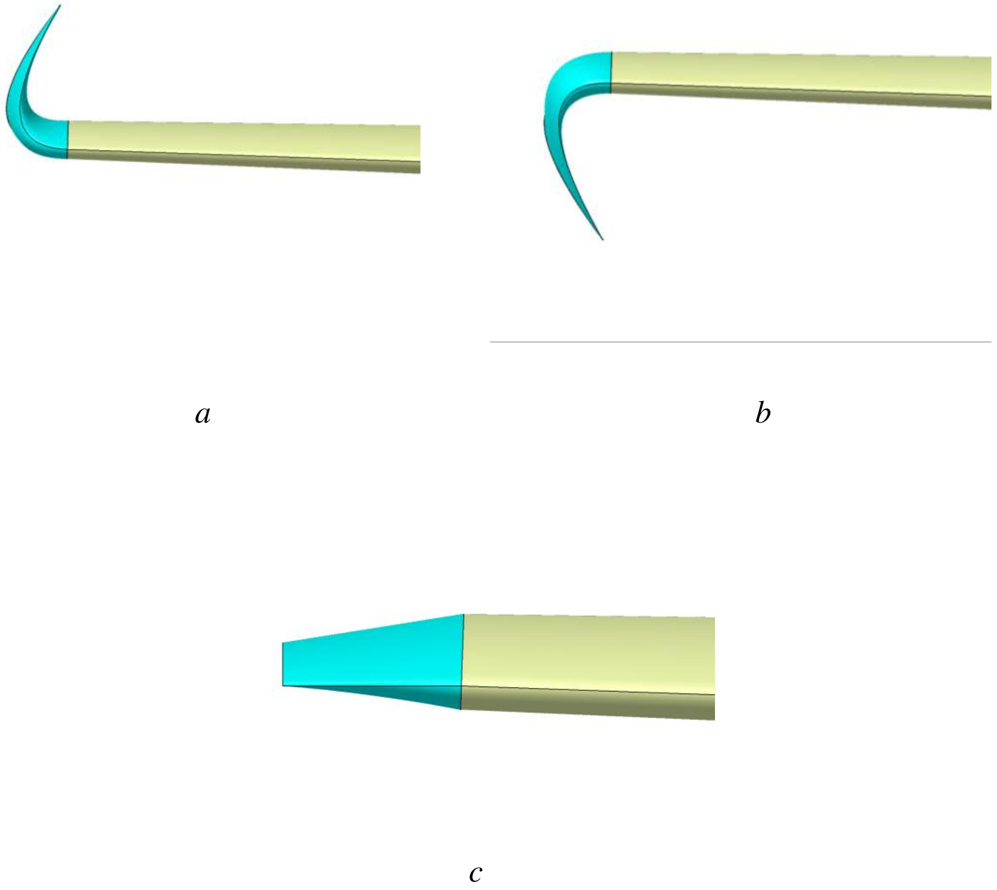


Figure 35 Instantiated tip devices: (a) Upswept; (b) Drooped; (c) Hoerner

4.4. Comparison of CAD models

The parametric model created for different types of winglets could adapt various geometry shapes accordingly with the given parameters. Below are some of the winglet models compared with the existing winglet of aircraft.

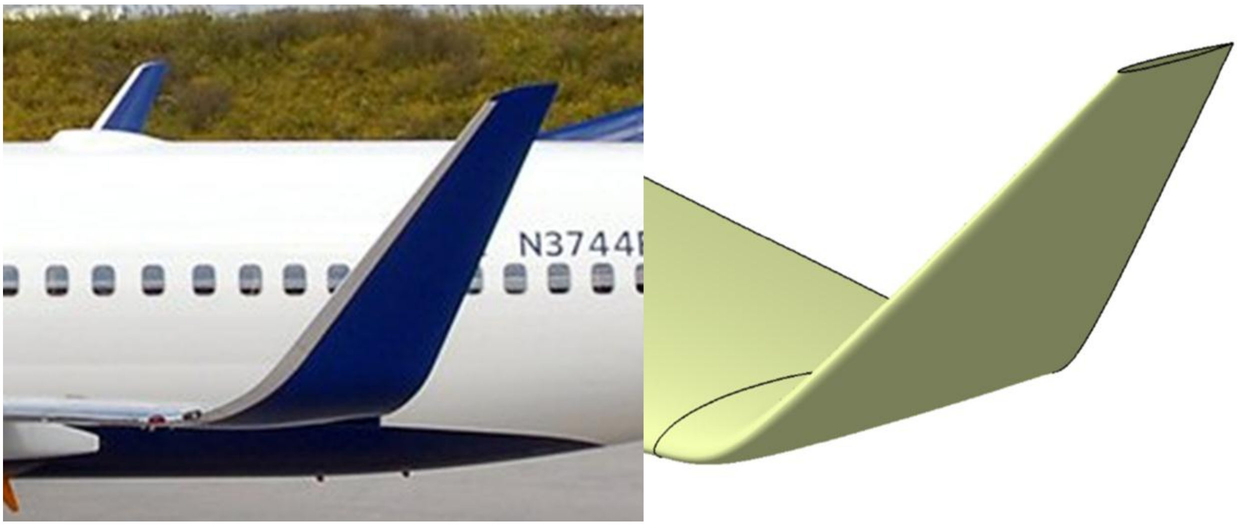


Figure 36 Comparison Boeing 737 winglet with parametric blended winglet

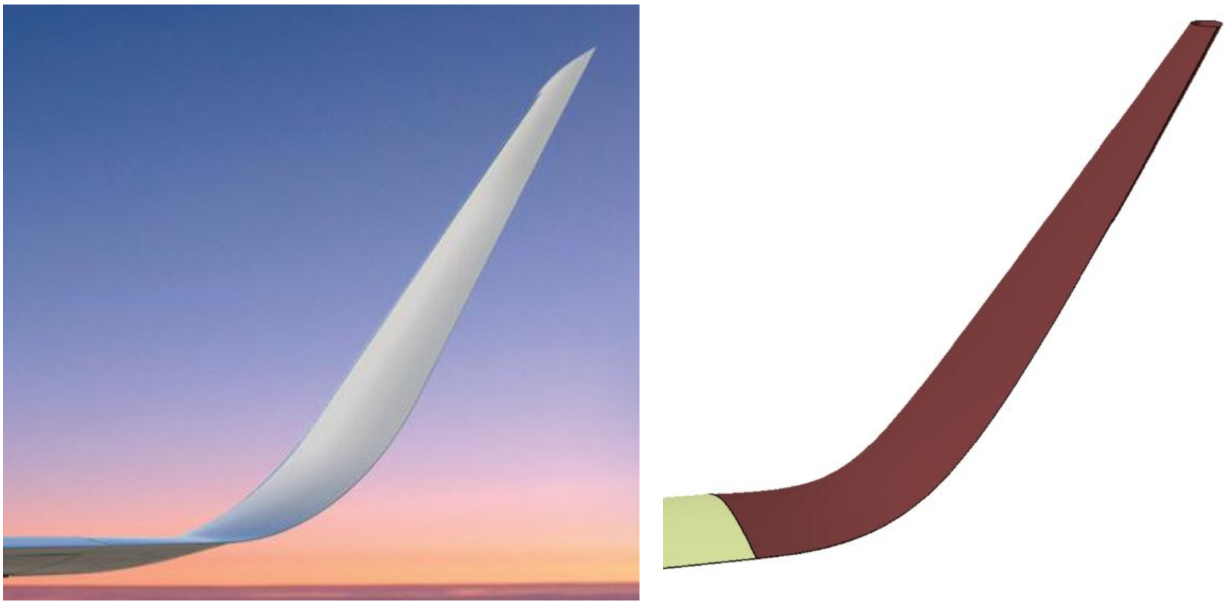


Figure 37 Comparison Aviation Partners Inc. blended winglet with parametric blended winglet

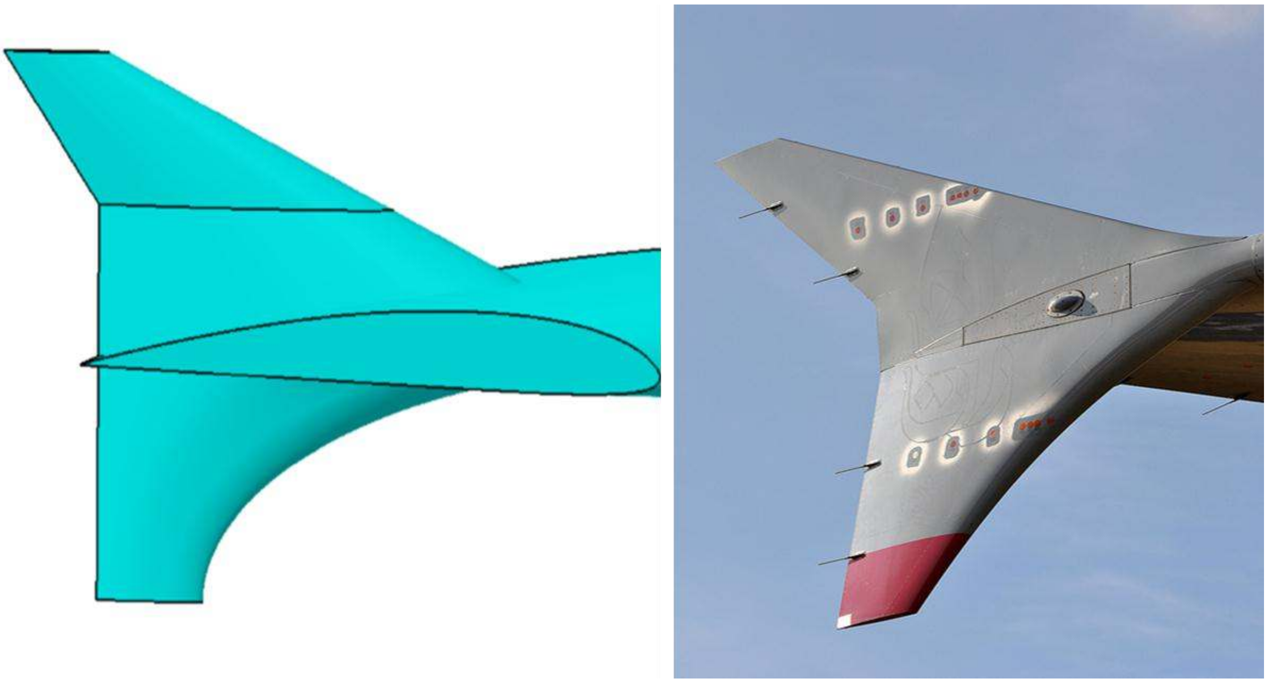


Figure 38 Comparison parametric wingtip fence with A380 wingtip fence

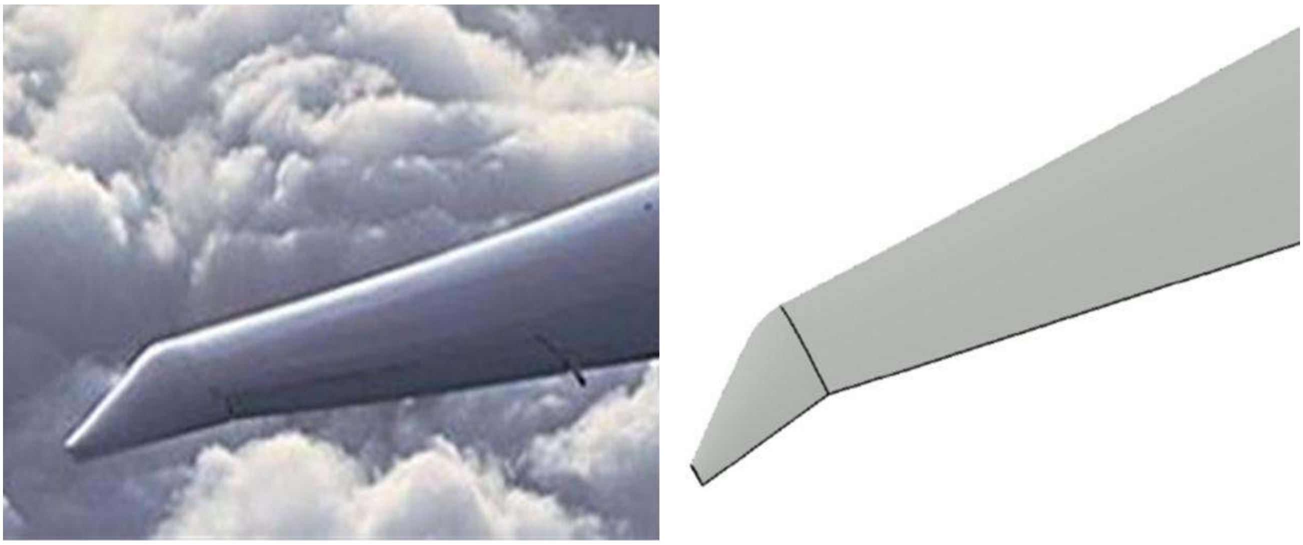


Figure 39 Comparison Boeing 777 winglet with parametric raked winglet

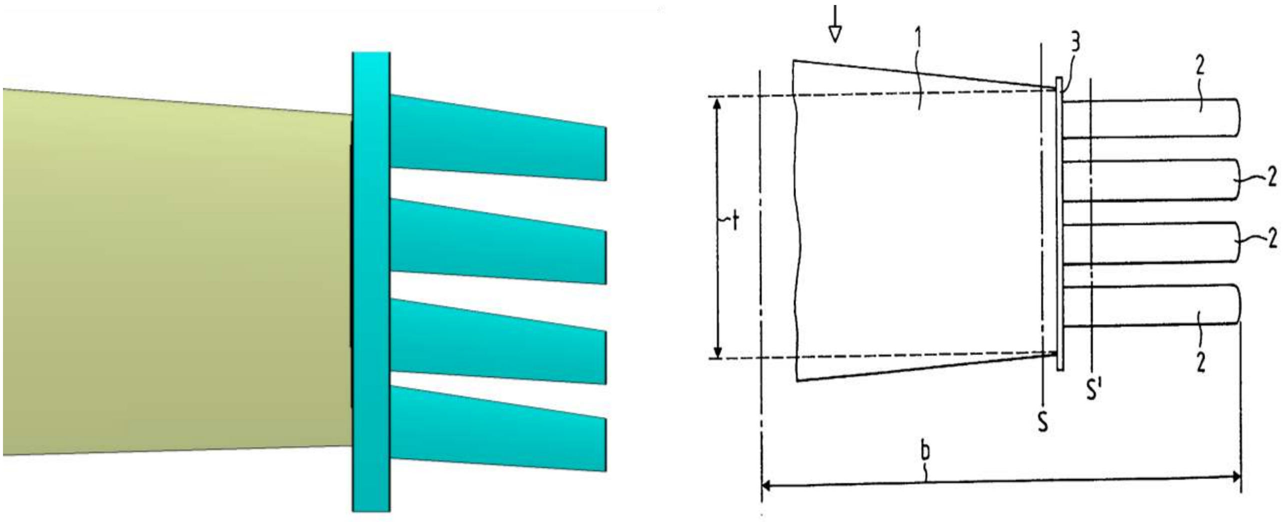


Figure 40 Comparison parametric wing grid with La Roche’s wing grid

5. Conclusion

Different types of winglets and tip devices have been studied and also their role in the aircraft industry is discussed in this work. Reducing overall drag in an aircraft is possible with the help of winglets. Adding winglets to aircraft wing provide an increase in aspect ratio which is one of the main factors for reducing induced drag. Moreover, the concept of induced drag is studied and its contribution in the overall aircraft performance is calculated. All the calculations made are through a conceptual approach i.e. by applying simple formula and no sophisticated techniques are used. Based on the results of induced drag calculation and take-off performance calculation, an additional weight for each drag count reduction is identified. Airline operators could have an extra passenger / payload for the same take-off field length, which is one of the bottom line benefits for them. Also, each passenger accounts for extra income for the operator and this is one of the main reasons behind the installation of the winglet. From the aerodynamic perspective, winglets would reduce the induced drag despite increasing the parasite drag. Optimizing the size of the winglets could avoid the drastic increase of frictional drag, thereby reducing overall drag at compromising level.

On the other hand, by understanding the significance of winglets to an aircraft, automated generic models of winglet are designed mainly focusing on reducing the design time. Furthermore, Knowledge Pattern approach and its features are used for the instantiations. The model of main wing where the parametric model to be instantiated should be named as mentioned in UDFs. Also, the airfoil to be used in the winglet would also be dependent on the tip airfoil of the main wing. Pre-instantiation work has to be done before choosing the number of blend panels in case of blended winglet type. The skeleton of the blended vertical should be modified before instantiating the panels in order to get the desired shape and this applies to all the models except for tip devices. The automation work is done in this work based on Knowledge Pattern only and no VBA approach is used. Hence, no time comparison analysis is performed and so time for each instantiation is not observed.

Generic model developed could take different shapes and sizes with the help associated parameters and could be used in the pre-design stage of winglets, where spending more time in the design process can be minimized. All the winglets developed in this work are designed according with the design criteria provided by the respective patent holders and so there will be no need for designing a specific type of winglet from the base, when this model has been used.

6. Future work

Since the results presented in this work for induced drag calculation are only through basic formulas which only depend on area of the winglet, a flow analysis could be done in order to determine the lift-off performance, vorticity reduction and also to find the parasite drag associated with size of winglet.

Based on the type of analysis, user interface can be created. If the analysis is to be done on Tornado, an Excel user interface should be linked with MATLAB and CATIA. Upon changing the values in excel the CAD model should change its geometry and the flow results should be obtained automatically from Tornado.

Similarly, for CFD method analysis, software like ANSYS with CFX should be linked with CATIA V5. So an actual drag reduction, vortex effects for the respective winglet can be calculated by varying the parameters. Optimization of the geometry for each type of winglet could be done, by targeting drag as minimizing factor.

Moreover, applying a composite material to each winglet and optimizing the drag could be performed. Ribs and spars would be designed for each winglets and a finite element analysis could be performed to calculate structural integrity using FEA tools like ANSYS or CATIA.

This work has been done with only following Knowledge Pattern approach for automation, so VB approach by making power copies of each template could be done in order to compare the efficiency of both approaches.

7. References

[1] R. T. Whitcomb, "A Design Approach and Selected Wing-Tunnel Result at High Subsonic Speed for Wing-Tip Mounted Winglets," *NASA TN D-8260,* 1976.

[2] R. T. Jones and T. A. Lasinski, "Effect of Winglets on the Induced Drag of Ideal Wing Shapes," *NASA TM-81230,* 1980.

[3] C. D. Cone, "Theory of Induced Lift and Minimum Induced Drag of Nonplanar Lifting Systems," *NASA TR-RI39,* 1962.

[4] J. Mohammed*, et al.*, "Applications of Computer Aided Design (CAD) In Knowledge Based Engineering," *Proceedings of the 2008 IAJC-IJME International Conference,* 2008.

[5] M. V. Raghu Chaitanya*, et al.*, "Model Based Aircraft Control System Design and Simulation," *International Congress of the Aeronautical Sciences,* 2010.

[6] C. Ledermann*, et al.*, "Associative parametric CAE methods in the aircraft pre-design," *Aerospace Science and Technology,* vol. 9, pp. 641-651, 2005.

[7] X. F. Zha*, et al.*, "Knowledge-based approach and system for assembly oriented design, Part I: the approach," *Engineering Applications of Artificial Intelligence,* vol. 14, pp. 61-75, 2001.

[8] C. Ledermann*, et al.*, "Dynamic CAD objects for structural optimization in preliminary aircraft design," *Aerospace Science and Technology,* vol. 10, pp. 601-610, 2006.

[9] N. S. Aakash and G. Vijayakumar, "Automated generic parameterized design of aircraft fairing and windshield," Master Thesis, Department of Management and Engineering, Linkoping University, Sweden, 2012.

[10] M. Sohaib, "Parameterized Automated Generic Model for Aircraft Wing Structural Design and Mesh Generation for Finite Element Analysis," Masters in Mechanical Engineering Master Thesis, Department of Management and Engineering, Linkoping University, Linkoping, Sweden, 2011.

[11] L. B. Gratzer, "Blended winglet," ed: Google Patents, 1994.

[12] L. B. Gratzer, "Spiroid-tipped wing," ed: Google Patents, 1992.

[13] U. La Roche, "Wing with a wing grid as the end section," ed: Google Patents, 1998.

[14] L. L. Herrick*, et al.*, "Blunt-leading-edge raked wingtips," ed: Google Patents, 2000.

[15] D. P. Raymer*, et al.*, *Aircraft design: a conceptual approach*: American Institute of Aeronautics and Astronautics, 1999.

[16] H. H. Heyson*, et al.*, "Theoretical Parametric Study of the Relative Advantages of Winglets and Wing-tip Extension," NASA Langely Research Centre, Virginia1977.

[17] J. E. Guerrero*, et al.*, "Biomimetic spiroid winglets for lift and drag control," *Comptes Rendus Mécanique,* vol. 340, pp. 67-80, 2012.

[18] L. R. Jenkinson*, et al.*, *Civil jet aircraft design* vol. 7: Arnold London, 1999.

[19] P. Thiede, *Aerodynamic Drag Reduction Technologies: Proceedings of the CEAS/DragNet European Drag Reduction Conference, 19-21 June 2000, Potsdam, Germany* vol. 76: Springer Verlag, 2001.

[20] D. McLean, "Wingtip Devices: What they Do and How They Do it," presented at the Boeing Performance and Flight Operations Engineering Conference 2005.

Appendix 1

Aircraft specification table (source: www.airbus.com)

Airbus A320 specifications	
MTOW	7800kg
Wing span	34
Wing span (with winglets)	36
Root Chord	7m
Range	6150 km
Payload	16.6 tons
Take off distance	1500 m
Engine Specification	
Type	CFM56 –B
Number of Engines	2
Take-off Thrust	23,500 lbs.

Appendix 2

```
let blended (UserFeature)
let i (Integer)
let j (Integer)
let k (Integer)
let x (Integer)
let verticalPanel (UserFeature)
let oldblend (UserFeature)
let wingFence (UserFeature)
let airfoilV (UserFeature)
let airfoilV1 (UserFeature)
let oldairfoilV1(UserFeature)
let oldfence (UserFeature)
let fencePanelV (UserFeature)
let fencePanelV1(UserFeature)
let oldairfoilV (UserFeature)
let oldpanelV (UserFeature)
let oldpanelV1 (UserFeature)
let airfoil (String)
let fenceL (UserFeature)
let oldfenceL (UserFeature)
let fencePanelH (UserFeature)
let oldpanelH (UserFeature)
let raked (UserFeature)
let oldraked (UserFeature)
let rakedwing (UserFeature)
let oldrakedwing (UserFeature)
let WingGrid (UserFeature)
let winggrid (UserFeature)
let oldudf (UserFeature)
let oldgrid (UserFeature)
let airfoilH (UserFeature)
let oldairfoil (UserFeature)
let gridpanel (UserFeature)
let oldgridpanel(UserFeature)
let upswept (UserFeature)
let drooped (UserFeature)
let hoerner (UserFeature)
let tipextension (UserFeature)
let oldtipextension (UserFeature)

/* ******************************************************** */
/* Blended Winglet */
```

```
if `typeOfWinglet` == "blended"
{
        blended=CreateOrModifyTemplate("WingletCatalog|blended1",`Instantiated`
,`Relations\Knowledge Pattern.1\blended` ,1)
        blended->SetAttributeObject("ZXPlane",`External References\ZXPlane`   )
        blended->SetAttributeObject("rootChord",`External
References\tipChordNewZeroTwistNew` )
        blended->SetAttributeObject("TEPointAirfoilUpper",`External
References\TEPointAirfoilUpper`  )
        blended->SetAttributeObject("TEPointAirfoilLower",`External
References\TEPointAirfoilLower` )
        blended->SetAttributeObject("airfoilWithThicknessTERoot",`External
References\airfoilWithThicknessTE` )
        blended->SetAttributeObject("LEPointAirfoil",`External
References\tipChordLEPointNew` )
        blended->SetAttributeObject("airfoilWithOutThicknessTERoot",`External
References\airfoilWithOutThicknessTE` )
        blended->SetAttributeObject("YZPlane",`External References\YZPlane` )
        blended->SetAttributeObject("XYPlane",`External References\XYPlane` )
        blended->SetAttributeObject("apex",`External
References\tipChordLEPointNew` )
        blended->SetAttributeObject("tangentLE",`External References\LE` )
        blended->SetAttributeObject("tangentTE",`External References\TE` )
        EndModifyTemplate(blended)
        blended.Name = "Blended." + ToString(1)
        /* Tip Extension */

        if `noOfBlendPanel` ==0
        {
                if `tipExtension` == "yes"
                {

        tipextension=CreateOrModifyTemplate("WingletCatalog|tipExtensionVertical",
`Instantiated` ,`Relations\Knowledge Pattern.1\tipdevices` ,1)
                        oldblend=`Relations\Knowledge Pattern.1\blended` -
>GetItem(1)
                        tipextension-
>SetAttributeObject("rootChord",oldblend->GetAttributeObject("rootChordNewZeroTwist"))
                        tipextension-
>SetAttributeObject("rootAirfoilWithThicknessTE",oldblend-
>GetAttributeObject("rootAirfoilWithThicknessTE") )
                        tipextension-
>SetAttributeObject("tangentLE",oldblend->GetAttributeObject("sweepCantLine") )
```

```
            tipextension->SetAttributeObject("rootAirfoilWithOutThicknessTE",oldblend->GetAttributeObject("rootAirfoilWithOutThicknessTE") )
            tipextension->SetAttributeObject("apex",oldblend->GetAttributeObject("rootChordLEPoint") )
            tipextension->SetAttributeObject("tangentTE",oldblend->GetAttributeObject("TEWithOutThickness") )
            tipextension->SetAttributeObject("rootLEPointAirfoil",oldblend->GetAttributeObject("rootLEPointAirfoil")  )
            tipextension->SetAttributeObject("TEPointAirfoilUpper",oldblend->GetAttributeObject("TEPointAirfoilUpperOut"))
            tipextension->SetAttributeObject("TEPointAirfoilLower",oldblend->GetAttributeObject("TEPointAirfoilLowerOut")  )
            EndModifyTemplate(tipextension)
            tipextension.Name = "TipExtension." +ToString(1)

            oldtipextension=`Relations\Knowledge Pattern.1\tipdevices`->GetItem(1)
            oldtipextension->SetAttributeString("airfoilType",`typeOfAirfoil` )
        }

    }

    oldblend=`Relations\Knowledge Pattern.1\blended`->GetItem(1)
    oldblend->SetAttributeString("airfoilType",`typeOfAirfoil` )
    /* Blended Vertical Panel */

    if `noOfBlendPanel` > 0
    {
        i =1
        for i while i<= `noOfBlendPanel`
        {
            if i ==1
            {

    verticalPanel=CreateOrModifyTemplate("WingletCatalog|partitionVT",`Instantiated` ,`Relations\Knowledge Pattern.1\panelBlend` ,i)
                oldblend=`Relations\Knowledge Pattern.1\blended` ->GetItem(1)
```

```
                                        verticalPanel-
>SetAttributeObject("verticalTailPoint",oldblend->GetAttributeObject("rootChordLEPoint")
)
                                        verticalPanel-
>SetAttributeObject("leadingEdge",oldblend->GetAttributeObject("leadingEdgeVertical") )
                                        verticalPanel-
>SetAttributeObject("rootChordNewZeroTwist",oldblend-
>GetAttributeObject("rootChordNewZeroTwist") )
                                        verticalPanel-
>SetAttributeObject("trailingEdge",oldblend->GetAttributeObject("trailingEdgeVertical") )
                                        verticalPanel-
>SetAttributeObject("rootLEPointAirfoil",oldblend-
>GetAttributeObject("rootLEPointAirfoil") )
                                        verticalPanel-
>SetAttributeObject("increasedRootChordTEPoint",oldblend-
>GetAttributeObject("rootChordTEPoint") )
                                        verticalPanel-
>SetAttributeObject("TEPointAirfoilUpper",oldblend-
>GetAttributeObject("TEPointAirfoilUpperOut") )
                                        verticalPanel-
>SetAttributeObject("TEPointAirfoilLower",oldblend-
>GetAttributeObject("TEPointAirfoilLowerOut") )
                                        verticalPanel-
>SetAttributeObject("TEPointAirfoilWithoutThickness",oldblend-
>GetAttributeObject("TEPointAirfoilWithoutThickness") )
                                        verticalPanel-
>SetAttributeObject("rootChordTEPoint",oldblend-
>GetAttributeObject("rootChordTEPoint") )
                                        verticalPanel-
>SetAttributeObject("rootChordLEPoint",oldblend-
>GetAttributeObject("rootChordLEPoint") )
                                        verticalPanel-
>SetAttributeObject("rootAirfoilWithThicknessTE",oldblend-
>GetAttributeObject("rootAirfoilWithThicknessTE") )
                                        verticalPanel-
>SetAttributeObject("rootAirfoilWithOutThicknessTE",oldblend-
>GetAttributeObject("rootAirfoilWithOutThicknessTE") )
                                        verticalPanel-
>SetAttributeObject("XYPlane",oldblend->GetAttributeObject("XYPlaneOut") )
                                        EndModifyTemplate(verticalPanel)

                              }
                              if i > 1
                              {
```

```
		verticalPanel=CreateOrModifyTemplate("WingletCatalog|partitionVT",`Instantiated` ,`Relations\Knowledge Pattern.1\panelBlend` ,i)
					oldblend = `Relations\Knowledge Pattern.1\blended` ->GetItem(1)
					oldpanelV = `Relations\Knowledge Pattern.1\panelBlend` ->GetItem(i-1)
					verticalPanel->SetAttributeObject("verticalTailPoint",oldpanelV->GetAttributeObject("LEPoint") )
					verticalPanel->SetAttributeObject("leadingEdge",oldblend->GetAttributeObject("leadingEdgeVertical") )
					verticalPanel->SetAttributeObject("rootChordNewZeroTwist",oldpanelV->GetAttributeObject("tipChordNewZeroTwist") )
					verticalPanel->SetAttributeObject("trailingEdge",oldblend->GetAttributeObject("trailingEdgeVertical") )
					verticalPanel->SetAttributeObject("rootLEPointAirfoil",oldpanelV->GetAttributeObject("tipLEPointAirfoil") )
					verticalPanel->SetAttributeObject("increasedRootChordTEPoint",oldpanelV->GetAttributeObject("increasedTipChordTEPoint") )
					verticalPanel->SetAttributeObject("TEPointAirfoilUpper",oldpanelV->GetAttributeObject("TEPointAirfoilUpperOut") )
					verticalPanel->SetAttributeObject("TEPointAirfoilLower",oldpanelV->GetAttributeObject("TEPointAirfoilLowerOut") )
					verticalPanel->SetAttributeObject("TEPointAirfoilWithoutThickness",oldpanelV->GetAttributeObject("TEPointAirfoilWithoutThicknessOut") )
					verticalPanel->SetAttributeObject("rootChordTEPoint",oldpanelV->GetAttributeObject("tipChordTEPoint") )
					verticalPanel->SetAttributeObject("rootChordLEPoint",oldpanelV->GetAttributeObject("tipChordLEPoint") )
					verticalPanel->SetAttributeObject("rootAirfoilWithThicknessTE",oldpanelV->GetAttributeObject("tipAirfoilWithThicknessTE") )
					verticalPanel->SetAttributeObject("rootAirfoilWithOutThicknessTE",oldpanelV->GetAttributeObject("tipAirfoilWithOutThicknessTE") )
```

```
                                                            verticalPanel-
>SetAttributeObject("XYPlane",oldpanelV->GetAttributeObject("XYPlaneOut") )
                                                            EndModifyTemplate(verticalPanel)
                                            }
                                            verticalPanel.Name="Panel."+ToString(i)
                                            oldblend=`Relations\Knowledge Pattern.1\blended` -
>GetItem(1)
                                            oldpanelV=`Relations\Knowledge
Pattern.1\panelBlend` ->GetItem(i)
                                            oldpanelV-
>SetAttributeString("airfoilType",`typeOfAirfoil` )
                                            oldpanelV->SetAttributeReal("dihedral",oldblend-
>GetAttributeReal("cantAngle") )
                                            if `tipExtension` == "yes"
                                            {
                                                            j=`noOfBlendPanel`


                tipextension=CreateOrModifyTemplate("WingletCatalog|tipExtensionVertical",
`Instantiated` ,`Relations\Knowledge Pattern.1\tipdevices` ,1)
                                                            oldblend = `Relations\Knowledge
Pattern.1\blended` ->GetItem(1)
                                                            oldpanelV =`Relations\Knowledge
Pattern.1\panelBlend` ->GetItem(j)
                                                            tipextension-
>SetAttributeObject("rootChord",oldpanelV-
>GetAttributeObject("tipChordNewZeroTwist"))
                                                            tipextension-
>SetAttributeObject("rootAirfoilWithThicknessTE",oldpanelV-
>GetAttributeObject("tipAirfoilWithThicknessTE") )
                                                            tipextension-
>SetAttributeObject("tangentLE",oldblend->GetAttributeObject("sweepCantLine") )
                                                            tipextension-
>SetAttributeObject("rootAirfoilWithOutThicknessTE",oldpanelV-
>GetAttributeObject("tipAirfoilWithOutThicknessTE") )
                                                            tipextension-
>SetAttributeObject("apex",oldpanelV->GetAttributeObject("tipChordLEPoint") )
                                                            tipextension-
>SetAttributeObject("tangentTE",oldblend->GetAttributeObject("trailingEdgeVertical") )
                                                            tipextension-
>SetAttributeObject("rootLEPointAirfoil",oldpanelV->GetAttributeObject("LEPoint") )
                                                            tipextension-
>SetAttributeObject("TEPointAirfoilUpper",oldpanelV-
>GetAttributeObject("TEPointAirfoilUpperOut") )
```

```
                tipextension-
>SetAttributeObject("TEPointAirfoilLower",oldpanelV-
>GetAttributeObject("TEPointAirfoilLowerOut") )
                EndModifyTemplate(tipextension)
                tipextension.Name = "TipExtension."
+ToString(1)

                oldtipextension=`Relations\Knowledge
Pattern.1\tipdevices` ->GetItem(1)
                oldtipextension-
>SetAttributeString("airfoilType",`typeOfAirfoil` )
            }
        }

    }

}

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

/*  Wing Tip Fence  */

if `typeOfWinglet` =="wingFence"
{
    wingFence=CreateOrModifyTemplate("WingletCatalog|wingFence",`Instantiate
d` ,`Relations\Knowledge Pattern.1\wingFence` ,1)
    wingFence->SetAttributeObject("apexTipExtension",`External
References\tipChordLEPointNew` )
    wingFence->SetAttributeObject("rootChord",`External
References\tipChordNewZeroTwistNew` )
    wingFence->SetAttributeObject("xy plane",`External References\XYPlane` )
    wingFence->SetAttributeObject("yz plane",`External References\YZPlane` )
    wingFence->SetAttributeObject("zx plane",`External References\ZXPlane` )
    EndModifyTemplate(wingFence)
    wingFence.Name="WingFence."+ToString(1)
    if `noOfFencePanelH` > 0
    {
        k = 1
        for k while k<= `noOfFencePanelH`
        {
            if k == 1
            {
```

```
            fencePanelH=CreateOrModifyTemplate("WingletCatalog|partitionHT",`Instanti
ated` ,`Relations\Knowledge Pattern.1\fencePanelH` ,k)
                                        oldfence = `Relations\Knowledge
Pattern.1\wingFence` ->GetItem(1)
                                        fencePanelH-
>SetAttributeObject("rootChordLEPointIC",`External References\tipChordLEPointNew` )
                                        fencePanelH-
>SetAttributeObject("leadingEdge",oldfence->GetAttributeObject("leadingEdge") )
                                        fencePanelH-
>SetAttributeObject("rootChordNewZeroIncidenceIC",`External
References\tipChordNewZeroTwistNew` )
                                        fencePanelH-
>SetAttributeObject("trailingEdge",oldfence->GetAttributeObject("trailingEdge") )
                                        fencePanelH-
>SetAttributeObject("ZXPlane",`External References\ZXPlane` )
                                        fencePanelH-
>SetAttributeObject("LEPointAirfoil",`External References\LEPointAirfoilNew` )
                                        fencePanelH-
>SetAttributeObject("increasedRootChordTEPoint",`External
References\increasedTipChordTEPointNew` )
                                        fencePanelH-
>SetAttributeObject("TEPointAirfoilUpper",`External References\TEPointAirfoilUpper` )
                                        fencePanelH-
>SetAttributeObject("TEPointAirfoilLower",`External References\TEPointAirfoilLower` )
                                        fencePanelH-
>SetAttributeObject("airfoilWithOutThicknessTERoot",`External
References\airfoilWithOutThicknessTE` )
                                        fencePanelH-
>SetAttributeObject("rootChordTEPointIC",`External References\tipChordTEPointNew` )
                                        fencePanelH-
>SetAttributeObject("airfoilWithThicknessTERoot",`External
References\airfoilWithThicknessTE` )
                                        fencePanelH-
>SetAttributeObject("increasedRootChord",`External References\increasedTipChord` )
                                        fencePanelH-
>SetAttributeObject("TEPoint",`External References\tipChordTEPointNew` )
                                        EndModifyTemplate(fencePanelH)
                                        fencePanelH.Name = "PanelH." +
ToString(k)
                              }
                              if k >1
                              {
```

```
        fencePanelH=CreateOrModifyTemplate("WingletCatalog|partitionHT",`Instanti
ated` ,`Relations\Knowledge Pattern.1\fencePanelH` ,k)
                              oldfence=`Relations\Knowledge
Pattern.1\wingFence` ->GetItem(1)
                              oldpanelH=`Relations\Knowledge
Pattern.1\fencePanelH` ->GetItem(k-1)
                              fencePanelH-
>SetAttributeObject("rootChordLEPointIC",oldpanelH-
>GetAttributeObject("tipChordLEPointNew") )
                              fencePanelH-
>SetAttributeObject("leadingEdge",oldfence->GetAttributeObject("leadingEdge") )
                              fencePanelH-
>SetAttributeObject("rootChordNewZeroIncidenceIC",oldpanelH-
>GetAttributeObject("tipChordNewZeroTwistNew") )
                              fencePanelH-
>SetAttributeObject("trailingEdge",oldfence->GetAttributeObject("trailingEdge") )
                              fencePanelH-
>SetAttributeObject("ZXPlane",oldpanelH->GetAttributeObject("ZXPlane") )
                              fencePanelH-
>SetAttributeObject("LEPointAirfoil",oldpanelH->GetAttributeObject("LEPointAirfoilNew")
)
                              fencePanelH-
>SetAttributeObject("increasedRootChordTEPoint",oldpanelH-
>GetAttributeObject("increasedTipChordTEPointNew") )
                              fencePanelH-
>SetAttributeObject("TEPointAirfoilUpper",oldpanelH-
>GetAttributeObject("TEPointAirfoilUpperout") )
                              fencePanelH-
>SetAttributeObject("TEPointAirfoilLower",oldpanelH-
>GetAttributeObject("TEPointAirfoilLowerOut") )
                              fencePanelH-
>SetAttributeObject("airfoilWithOutThicknessTERoot",oldpanelH-
>GetAttributeObject("airfoilWithOutThicknessTE") )
                              fencePanelH-
>SetAttributeObject("rootChordTEPointIC",oldpanelH-
>GetAttributeObject("increasedTipChordTEPointNew") )
                              fencePanelH-
>SetAttributeObject("airfoilWithThicknessTERoot",oldpanelH-
>GetAttributeObject("airfoilWithThicknessTE") )
                              fencePanelH-
>SetAttributeObject("increasedRootChord",oldpanelH-
>GetAttributeObject("increasedTipChord") )
```

```
                fencePanelH-
>SetAttributeObject("TEPoint",oldpanelH->GetAttributeObject("TEPoint") )
                EndModifyTemplate(fencePanelH)
                fencePanelH.Name = "PanelH." +
ToString(k)
            }
            oldpanelH=`Relations\Knowledge
Pattern.1\fencePanelH` ->GetItem(k)
            oldpanelH-
>SetAttributeString("airfoilType",`typeOfAirfoil` )
        }


    }

    if `noOfFencePanelV1` > 0
    {

    airfoilV=CreateOrModifyTemplate("WingletCatalog|airfoilVT",`Instantiated`
,`Relations\Knowledge Pattern.1\airfoilV` ,1)
        oldfence=`Relations\Knowledge Pattern.1\wingFence` -
>GetItem(1)
        airfoilV->SetAttributeObject("verticalTailPoint",oldfence-
>GetAttributeObject("apexVertical") )
        airfoilV->SetAttributeObject("rootChord",oldfence-
>GetAttributeObject("rootChordVertical") )
        EndModifyTemplate(airfoilV)
        airfoilV->SetAttributeString("airfoilType",`typeOfAirfoil` )
        airfoilV.Name = "AirfoilV." + ToString(1)
        if `noOfFencePanelV1` >0
        {

            i=1
            for i while i <=`noOfFencePanelV1`
            {
                if i == 1
                {

    fencePanelV=CreateOrModifyTemplate("WingletCatalog|partitionVT",`Instanti
ated` ,`Relations\Knowledge Pattern.1\panelFenceV1`  ,i)
                    oldairfoilV=
`Relations\Knowledge Pattern.1\airfoilV` ->GetItem(1)
                    oldfence
=`Relations\Knowledge Pattern.1\wingFence` ->GetItem(1)
```

```
                                                            fencePanelV-
>SetAttributeObject("verticalTailPoint",oldfence->GetAttributeObject("apexVertical") )
                                                            fencePanelV-
>SetAttributeObject("leadingEdge",oldfence->GetAttributeObject("leadingEdgeUP1") )
                                                            fencePanelV-
>SetAttributeObject("rootChordNewZeroTwist",oldairfoilV-
>GetAttributeObject("rootChordNewZeroTwist") )
                                                            fencePanelV-
>SetAttributeObject("trailingEdge",oldfence->GetAttributeObject("trailingEdgeUP1") )
                                                            fencePanelV-
>SetAttributeObject("rootLEPointAirfoil",oldairfoilV-
>GetAttributeObject("rootLEPointAirfoil") )
                                                            fencePanelV-
>SetAttributeObject("increasedRootChordTEPoint",oldairfoilV-
>GetAttributeObject("TEPointAirfoilWithoutThickness") )
                                                            fencePanelV-
>SetAttributeObject("TEPointAirfoilUpper",oldairfoilV-
>GetAttributeObject("TEPointAirfoilUpper") )
                                                            fencePanelV-
>SetAttributeObject("TEPointAirfoilLower",oldairfoilV-
>GetAttributeObject("TEPointAirfoilLower") )
                                                            fencePanelV-
>SetAttributeObject("TEPointAirfoilWithoutThickness",oldairfoilV-
>GetAttributeObject("TEPointAirfoilWithoutThickness") )
                                                            fencePanelV-
>SetAttributeObject("rootChordLEPoint",oldairfoilV-
>GetAttributeObject("rootLEPointAirfoil") )
                                                            fencePanelV-
>SetAttributeObject("rootChordTEPoint",oldairfoilV-
>GetAttributeObject("TEPointAirfoilWithoutThickness") )
                                                            fencePanelV-
>SetAttributeObject("rootAirfoilWithThicknessTE",oldairfoilV-
>GetAttributeObject("rootAirfoilWithThicknessTE") )
                                                            fencePanelV-
>SetAttributeObject("rootAirfoilWithOutThicknessTE",oldairfoilV-
>GetAttributeObject("rootAirfoilWithOutThicknessTE") )
                                                            fencePanelV-
>SetAttributeObject("XYPlane",oldairfoilV->GetAttributeObject("XYPlane") )

          EndModifyTemplate(fencePanelV)

                                                  }
                                                  if i>1
                                                  {
```

```
                fencePanelV=CreateOrModifyTemplate("WingletCatalog|partitionVT",`Instanti
ated`   ,`Relations\Knowledge Pattern.1\panelFenceV1`  ,i)
                                                            oldpanelV =
`Relations\Knowledge Pattern.1\panelFenceV1` ->GetItem(i-1)
                                                            oldfence =
`Relations\Knowledge Pattern.1\wingFence` ->GetItem(1)
                                                            fencePanelV-
>SetAttributeObject("verticalTailPoint",oldpanelV->GetAttributeObject("LEPoint") )
                                                            fencePanelV-
>SetAttributeObject("leadingEdge",oldfence->GetAttributeObject("leadingEdgeUP1") )
                                                            fencePanelV-
>SetAttributeObject("rootChordNewZeroTwist",oldpanelV-
>GetAttributeObject("tipChordNewZeroTwist") )
                                                            fencePanelV-
>SetAttributeObject("trailingEdge",oldfence->GetAttributeObject("trailingEdgeUP1") )
                                                            fencePanelV-
>SetAttributeObject("rootLEPointAirfoil",oldpanelV-
>GetAttributeObject("tipChordLEPoint") )
                                                            fencePanelV-
>SetAttributeObject("increasedRootChordTEPoint",oldpanelV-
>GetAttributeObject("increasedTipChordTEPoint") )
                                                            fencePanelV-
>SetAttributeObject("TEPointAirfoilUpper",oldpanelV-
>GetAttributeObject("TEPointAirfoilUpperOut") )
                                                            fencePanelV-
>SetAttributeObject("TEPointAirfoilLower",oldpanelV-
>GetAttributeObject("TEPointAirfoilLowerOut") )
                                                            fencePanelV-
>SetAttributeObject("TEPointAirfoilWithoutThickness",oldpanelV-
>GetAttributeObject("TEPointAirfoilWithoutThicknessOut") )
                                                            fencePanelV-
>SetAttributeObject("rootChordLEPoint",oldpanelV-
>GetAttributeObject("tipChordLEPoint") )
                                                            fencePanelV-
>SetAttributeObject("rootChordTEPoint",oldpanelV-
>GetAttributeObject("tipChordTEPoint") )
                                                            fencePanelV-
>SetAttributeObject("rootAirfoilWithThicknessTE",oldpanelV-
>GetAttributeObject("tipAirfoilWithThicknessTE") )
                                                            fencePanelV-
>SetAttributeObject("rootAirfoilWithOutThicknessTE",oldpanelV-
>GetAttributeObject("tipAirfoilWithOutThicknessTE") )
```

```
                    fencePanelV-
>SetAttributeObject("XYPlane",oldpanelV->GetAttributeObject("XYPlaneOut") )

    EndModifyTemplate(fencePanelV)

                }
                fencePanelV.Name = "PanelV1." +
ToString(i)
                oldpanelV=`Relations\Knowledge
Pattern.1\panelFenceV1` ->GetItem(i)
                oldpanelV-
>SetAttributeString("airfoilType",`typeOfAirfoil` )
            }

        }
        if `noOfFencePanelV2` > 0
        {

    airfoilV1=CreateOrModifyTemplate("WingletCatalog|airfoilVT",`Instantiated`
,`Relations\Knowledge Pattern.1\airfoilV1` ,1)
            oldfence=`Relations\Knowledge
Pattern.1\wingFence` ->GetItem(1)
            airfoilV1-
>SetAttributeObject("verticalTailPoint",oldfence->GetAttributeObject("apexUP2") )
            airfoilV1->SetAttributeObject("rootChord",oldfence-
>GetAttributeObject("tipChord") )
            EndModifyTemplate(airfoilV1)
            airfoilV1.Name = "AirfoilUP2."+ToString(1)
            x = 1
            for x while x <= `noOfFencePanelV2`
            {
                if x == 1
                {

    fencePanelV1=CreateOrModifyTemplate("WingletCatalog|partitionVT",`Instant
iated` ,`Relations\Knowledge Pattern.1\panelFenceV2` ,x)
                    oldairfoilV1 =
`Relations\Knowledge Pattern.1\airfoilV1` ->GetItem(1)
                    oldfence =
`Relations\Knowledge Pattern.1\wingFence` ->GetItem(1)
                    fencePanelV1-
>SetAttributeObject("verticalTailPoint",oldfence->GetAttributeObject("apexUP2") )
                    fencePanelV1-
>SetAttributeObject("leadingEdge",oldfence->GetAttributeObject("leadingEdgeUP2") )
```

```
                                                                        fencePanelV1-
>SetAttributeObject("rootChordNewZeroTwist",oldairfoilV1-
>GetAttributeObject("rootChordNewZeroTwist") )
                                                                        fencePanelV1-
>SetAttributeObject("trailingEdge",oldfence->GetAttributeObject("trailingEdgeUP2") )
                                                                        fencePanelV1-
>SetAttributeObject("rootLEPointAirfoil",oldairfoilV1-
>GetAttributeObject("rootLEPointAirfoil") )
                                                                        fencePanelV1-
>SetAttributeObject("increasedRootChordTEPoint",oldairfoilV1-
>GetAttributeObject("increasedRootChordTEPoint") )
                                                                        fencePanelV1-
>SetAttributeObject("TEPointAirfoilUpper",oldairfoilV1-
>GetAttributeObject("TEPointAirfoilUpper") )
                                                                        fencePanelV1-
>SetAttributeObject("TEPointAirfoilLower",oldairfoilV1-
>GetAttributeObject("TEPointAirfoilLower") )
                                                                        fencePanelV1-
>SetAttributeObject("TEPointAirfoilWithoutThickness",oldairfoilV1-
>GetAttributeObject("TEPointAirfoilWithoutThickness") )
                                                                        fencePanelV1-
>SetAttributeObject("rootChordLEPoint",oldairfoilV1-
>GetAttributeObject("rootChordLEPoint") )
                                                                        fencePanelV1-
>SetAttributeObject("rootChordTEPoint",oldairfoilV1-
>GetAttributeObject("rootChordTEPoint") )
                                                                        fencePanelV1-
>SetAttributeObject("rootAirfoilWithThicknessTE",oldairfoilV1-
>GetAttributeObject("rootAirfoilWithThicknessTE") )
                                                                        fencePanelV1-
>SetAttributeObject("rootAirfoilWithOutThicknessTE",oldairfoilV1-
>GetAttributeObject("rootAirfoilWithOutThicknessTE") )
                                                                        fencePanelV1-
>SetAttributeObject("XYPlane",oldairfoilV1->GetAttributeObject("XYPlane") )

            EndModifyTemplate(fencePanelV1)

                                                        }
                                                        if x > 1
                                                        {

            fencePanelV1=CreateOrModifyTemplate("WingletCatalog|partitionVT",`Instant
iated` ,`Relations\Knowledge Pattern.1\panelFenceV2` ,x)
```

```
                                        oldfence =
`Relations\Knowledge Pattern.1\wingFence` ->GetItem(1)
                                        oldpanelV1 =
`Relations\Knowledge Pattern.1\panelFenceV2`->GetItem(x-1)
                                        fencePanelV1-
>SetAttributeObject("verticalTailPoint",oldpanelV1->GetAttributeObject("LEPoint") )
                                        fencePanelV1-
>SetAttributeObject("leadingEdge",oldfence->GetAttributeObject("leadingEdgeUP2") )
                                        fencePanelV1-
>SetAttributeObject("rootChordNewZeroTwist",oldpanelV1-
>GetAttributeObject("tipChordNewZeroTwist") )
                                        fencePanelV1-
>SetAttributeObject("trailingEdge",oldfence->GetAttributeObject("trailingEdgeUP2") )
                                        fencePanelV1-
>SetAttributeObject("rootLEPointAirfoil",oldpanelV1-
>GetAttributeObject("tipLEPointAirfoil") )
                                        fencePanelV1-
>SetAttributeObject("increasedRootChordTEPoint",oldpanelV1-
>GetAttributeObject("increasedTipChordTEPoint") )
                                        fencePanelV1-
>SetAttributeObject("TEPointAirfoilUpper",oldpanelV1-
>GetAttributeObject("TEPointAirfoilUpperOut") )
                                        fencePanelV1-
>SetAttributeObject("TEPointAirfoilLower",oldpanelV1-
>GetAttributeObject("TEPointAirfoilLowerOut") )
                                        fencePanelV1-
>SetAttributeObject("TEPointAirfoilWithoutThickness",oldpanelV1-
>GetAttributeObject("TEPointAirfoilWithoutThicknessOut") )
                                        fencePanelV1-
>SetAttributeObject("rootChordLEPoint",oldpanelV1-
>GetAttributeObject("tipChordLEPoint") )
                                        fencePanelV1-
>SetAttributeObject("rootChordTEPoint",oldpanelV1-
>GetAttributeObject("tipChordTEPoint") )
                                        fencePanelV1-
>SetAttributeObject("rootAirfoilWithThicknessTE",oldpanelV1-
>GetAttributeObject("tipAirfoilWithThicknessTE") )
                                        fencePanelV1-
>SetAttributeObject("rootAirfoilWithOutThicknessTE",oldpanelV1-
>GetAttributeObject("tipAirfoilWithOutThicknessTE") )
                                        fencePanelV1-
>SetAttributeObject("XYPlane",oldpanelV1->GetAttributeObject("XYPlaneOut") )

        EndModifyTemplate(fencePanelV1)
```

```
                }
                fencePanelV1.Name = "PanelV2." +
ToString(x)
                oldpanelV1=`Relations\Knowledge
Pattern.1\panelFenceV2` ->GetItem(x)
                oldpanelV1-
>SetAttributeString("airfoilType",`typeOfAirfoil` )
            }


        }
        if `noOfLowerFencePanel` >0
        {
            j =1
            for j while j<=`noOfLowerFencePanel`
            {
                if j == 1
                {

    fenceL=CreateOrModifyTemplate("WingletCatalog|partitionVTCurved",`Instan
tiated` ,`Relations\Knowledge Pattern.1\fenceLower` ,j)
                    oldairfoilV =
`Relations\Knowledge Pattern.1\airfoilV`->GetItem(1)
                    oldfence =
`Relations\Knowledge Pattern.1\wingFence` ->GetItem(1)
                    fenceL-
>SetAttributeObject("rootChordLEPoint",oldairfoilV-
>GetAttributeObject("rootChordLEPoint") )
                    fenceL-
>SetAttributeObject("leadingEdge",oldfence->GetAttributeObject("leadinEdgeLO") )
                    fenceL-
>SetAttributeObject("rootChordNewZeroTwist",oldairfoilV-
>GetAttributeObject("rootChordNewZeroTwist") )
                    fenceL-
>SetAttributeObject("trailingEdge",oldfence->GetAttributeObject("trailingEdgeLO") )
                    fenceL-
>SetAttributeObject("XYPlane",oldairfoilV->GetAttributeObject("XYPlane") )
                    fenceL-
>SetAttributeObject("increasedRootChordTEPoint",oldairfoilV-
>GetAttributeObject("increasedRootChordTEPoint") )
                    fenceL-
>SetAttributeObject("rootLEPointAirfoil",oldairfoilV-
>GetAttributeObject("rootLEPointAirfoil") )
```

```
                                                        fenceL-
>SetAttributeObject("rootChordTEPoint",oldairfoilV-
>GetAttributeObject("rootChordTEPoint") )
                                                        fenceL-
>SetAttributeObject("TEPointAirfoilUpper",oldairfoilV-
>GetAttributeObject("TEPointAirfoilUpper") )
                                                        fenceL-
>SetAttributeObject("TEPointAirfoilLower",oldairfoilV-
>GetAttributeObject("TEPointAirfoilLower") )
                                                        fenceL-
>SetAttributeObject("TEPointAirfoilWithoutThickness",oldairfoilV-
>GetAttributeObject("TEPointAirfoilWithoutThickness") )
                                                        fenceL-
>SetAttributeObject("rootAirfoilWithThicknessTE",oldairfoilV-
>GetAttributeObject("rootAirfoilWithThicknessTE") )
                                                        fenceL-
>SetAttributeObject("rootAirfoilWithOutThicknessTE",oldairfoilV-
>GetAttributeObject("rootAirfoilWithOutThicknessTE") )

         EndModifyTemplate(fenceL)
                                                        fenceL.Name =
"PanelLower." + ToString(j)
                                             }
                                             if j>1
                                             {

         fenceL=CreateOrModifyTemplate("WingletCatalog|partitionVTCurved",`Instan
tiated` ,`Relations\Knowledge Pattern.1\fenceLower` ,j)

         oldfence=`Relations\Knowledge Pattern.1\wingFence`->GetItem(1)

         oldfenceL=`Relations\Knowledge Pattern.1\fenceLower` ->GetItem(j-1)
                                                        fenceL-
>SetAttributeObject("rootChordLEPoint",oldfenceL-
>GetAttributeObject("tipChordLEPoint") )
                                                        fenceL-
>SetAttributeObject("leadingEdge",oldfence->GetAttributeObject("leadinEdgeLO") )
                                                        fenceL-
>SetAttributeObject("rootChordNewZeroTwist",oldfenceL-
>GetAttributeObject("tipChordNewZeroTwist") )
                                                        fenceL-
>SetAttributeObject("trailingEdge",oldfence->GetAttributeObject("trailingEdgeLO") )
                                                        fenceL-
>SetAttributeObject("XYPlane",oldfenceL->GetAttributeObject("XYPlaneL") )
```

```
                                                           fenceL-
>SetAttributeObject("increasedRootChordTEPoint",oldfenceL-
>GetAttributeObject("increasedTipChordTEPoint") )
                                                           fenceL-
>SetAttributeObject("rootLEPointAirfoil",oldfenceL-
>GetAttributeObject("tipLEPointAirfoil") )
                                                           fenceL-
>SetAttributeObject("rootChordTEPoint",oldfenceL-
>GetAttributeObject("tipChordTEPoint") )
                                                           fenceL-
>SetAttributeObject("TEPointAirfoilUpper",oldfenceL-
>GetAttributeObject("TEPointAirfoilUpperL") )
                                                           fenceL-
>SetAttributeObject("TEPointAirfoilLower",oldfenceL-
>GetAttributeObject("TEPointAirfoilLowerL") )
                                                           fenceL-
>SetAttributeObject("TEPointAirfoilWithoutThickness",oldfenceL-
>GetAttributeObject("TEPointAirfoilWithoutThickness") )
                                                           fenceL-
>SetAttributeObject("rootAirfoilWithThicknessTE",oldfenceL-
>GetAttributeObject("tipAirfoilWithThicknessTE") )
                                                           fenceL-
>SetAttributeObject("rootAirfoilWithOutThicknessTE",oldfenceL-
>GetAttributeObject("tipAirfoilWithOutThicknessTE") )

            EndModifyTemplate(fenceL)
                                                           fenceL.Name
="PanelLower." + ToString(j)
                                               }
                                               oldfenceL=`Relations\Knowledge
Pattern.1\fenceLower` ->GetItem(j)
                                               oldfenceL-
>SetAttributeString("airfoilType",`typeOfAirfoil` )
                                    }

                        }

            }

}

if `typeOfWinglet` == "raked"
{
            raked=CreateOrModifyTemplate("WingletCatalog|rakedWingtip",`Instantiated` ,`Relations\Knowledge Pattern.1\raked` ,1)
            raked->SetAttributeObject("apex",`External References\tipChordLEPointNew` )
            raked->SetAttributeObject("rootChord",`External References\tipChordNewZeroTwistNew` )
            raked->SetAttributeObject("tangentLE",`External References\LE` )
            raked->SetAttributeObject("zx plane",`External References\ZXPlane` )
            raked->SetAttributeObject("xy plane",`External References\XYPlane` )
            raked->SetAttributeObject("yz plane",`External References\YZPlane` )
            EndModifyTemplate(raked)
            raked.Name = "Raked." + ToString(1)
            if `noOfRakedPanel` > 0
            {
                i = 1
                for i while i<= `noOfRakedPanel`
                        {
                            if i == 1
                            {

            rakedwing=CreateOrModifyTemplate("WingletCatalog|wingPartitionCurved",`Instantiated` ,`Relations\Knowledge Pattern.1\rakedPartition` ,i)
                        oldraked = `Relations\Knowledge Pattern.1\raked` ->GetItem(1)
                        rakedwing->SetAttributeObject("apex",`External References\tipChordLEPointNew` )
                        rakedwing->SetAttributeObject("LE",oldraked->GetAttributeObject("leadingEdge") )
                        rakedwing->SetAttributeObject("rootChord",`External References\tipChordNewZeroTwistNew` )
                        rakedwing->SetAttributeObject("trailingEdge",oldraked->GetAttributeObject("trailingEdge") )
                        rakedwing->SetAttributeObject("ZXPlane",`External References\ZXPlane` )
                        rakedwing->SetAttributeObject("LEPointAirfoil",`External References\LEPointAirfoilNew` )
                        rakedwing->SetAttributeObject("increasedRootChordTEPoint",`External References\increasedTipChordTEPointNew` )
                        rakedwing->SetAttributeObject("TEPointAirfoilUpper",`External References\TEPointAirfoilUpper` )
                        rakedwing->SetAttributeObject("TEPointAirfoilLower",`External References\TEPointAirfoilLower` )
```

```
                                                rakedwing-
>SetAttributeObject("airfoilWithOutThicknessTERoot",`External
References\airfoilWithOutThicknessTE` )
                                                rakedwing-
>SetAttributeObject("rootChordTEPointIC",`External
References\increasedTipChordTEPointNew` )
                                                rakedwing-
>SetAttributeObject("airfoilWithThicknessTERoot",`External
References\airfoilWithThicknessTE` )
                                                rakedwing-
>SetAttributeObject("increasedRootChord",`External References\increasedTipChord` )
                                                EndModifyTemplate(rakedwing)

                                    }

                                    if i> 1
                                    {

            rakedwing=CreateOrModifyTemplate("WingletCatalog|wingPartitionCurved",`I
nstantiated` ,`Relations\Knowledge Pattern.1\rakedPartition` ,i)
                                                oldraked = `Relations\Knowledge
Pattern.1\raked` ->GetItem(1)
                                                oldrakedwing = `Relations\Knowledge
Pattern.1\rakedPartition` ->GetItem(i-1)
                                                rakedwing-
>SetAttributeObject("apex",oldrakedwing->GetAttributeObject("tipChordLEPointNew") )
                                                rakedwing-
>SetAttributeObject("LE",oldraked->GetAttributeObject("leadingEdge") )
                                                rakedwing-
>SetAttributeObject("rootChord",oldrakedwing-
>GetAttributeObject("tipChordNewZeroTwistNew") )
                                                rakedwing-
>SetAttributeObject("trailingEdge",oldraked->GetAttributeObject("trailingEdge") )
                                                rakedwing-
>SetAttributeObject("ZXPlane",oldrakedwing->GetAttributeObject("ZXPlaneOut") )
                                                rakedwing-
>SetAttributeObject("LEPointAirfoil",oldrakedwing-
>GetAttributeObject("LEPointAirfoilNew") )
                                                rakedwing-
>SetAttributeObject("increasedRootChordTEPoint",oldrakedwing-
>GetAttributeObject("increasedTipChordTEPointNew") )
                                                rakedwing-
>SetAttributeObject("TEPointAirfoilUpper",oldrakedwing-
>GetAttributeObject("TEPointAirfoilUpperOut") )
```

```
                rakedwing-
>SetAttributeObject("TEPointAirfoilLower",oldrakedwing-
>GetAttributeObject("TEPointAirfoilLowerOut") )
                rakedwing-
>SetAttributeObject("airfoilWithOutThicknessTERoot",oldrakedwing-
>GetAttributeObject("airfoilWithOutThicknessTE") )
                rakedwing-
>SetAttributeObject("rootChordTEPointIC",oldrakedwing-
>GetAttributeObject("increasedTipChordTEPointNew") )
                rakedwing-
>SetAttributeObject("airfoilWithThicknessTERoot",oldrakedwing-
>GetAttributeObject("airfoilWithThicknessTE") )
                rakedwing-
>SetAttributeObject("increasedRootChord",oldrakedwing-
>GetAttributeObject("increasedTipChord") )
                EndModifyTemplate(rakedwing)
            }
            rakedwing.Name = "Partition." + ToString(i)
            oldrakedwing=`Relations\Knowledge
Pattern.1\rakedPartition` ->GetItem(i)
            oldrakedwing-
>SetAttributeString("airfoilType",`typeOfAirfoil` )
        }

    }
}


if `typeOfWinglet` == "wingGrid"
{

    WingGrid=CreateOrModifyTemplate("WingletCatalog|wingGrid",`Instantiated`
,`Relations\Knowledge Pattern.1\GridRoot` ,1)
    WingGrid->SetAttributeObject("zx plane",`External References\ZXPlane` )
    WingGrid->SetAttributeObject("rootChordLEPoint",`External
References\tipChordLEPointNew` )
    WingGrid->SetAttributeObject("rootChord",`External
References\tipChordNewZeroTwistNew` )
    EndModifyTemplate(WingGrid)
    WingGrid.Name = "GridRoot." + ToString(1)

    if `noOfGridPanels` > 0
    {
        i = 1
        for i while i <=`noOfGridPanels`
        {
            if i == 1
            {

    winggrid=CreateOrModifyTemplate("WingletCatalog|panelWingGrid",`Instanti
ated` ,`Relations\Knowledge Pattern.1\winggrid` ,i)
                oldudf=`Relations\Knowledge
Pattern.1\GridRoot` ->GetItem(1)
                winggrid->SetAttributeObject("yz
plane",`External References\YZPlane` )
                winggrid->SetAttributeObject("zx
plane",`External References\ZXPlane` )
                winggrid->SetAttributeObject("xy
plane",`External References\XYPlane` )
                winggrid-
>SetAttributeObject("rootChordLEPointGrid",oldudf-
>GetAttributeObject("rootChordLEPointGrid")  )
                winggrid-
>SetAttributeObject("rootChordGrid",oldudf->GetAttributeObject("rootChordGridRef")  )
                EndModifyTemplate(winggrid)
                winggrid.Name= "Grid." + ToString(i)

            }

            if i>1
            {

    winggrid=CreateOrModifyTemplate("WingletCatalog|panelWingGrid",`Instanti
ated` ,`Relations\Knowledge Pattern.1\winggrid` ,i)
                oldgrid = `Relations\Knowledge
Pattern.1\winggrid` ->GetItem(i-1)
                oldudf =`Relations\Knowledge
Pattern.1\GridRoot` ->GetItem(1)
                winggrid->SetAttributeObject("yz
plane",`External References\YZPlane` )
                winggrid->SetAttributeObject("zx
plane",`External References\ZXPlane` )
                winggrid->SetAttributeObject("xy
plane",`External References\XYPlane` )
                winggrid-
>SetAttributeObject("rootChordGrid",oldudf->GetAttributeObject("rootChordGridRef") )
```

```
                                        winggrid-
>SetAttributeObject("rootChordLEPointGrid",oldgrid-
>GetAttributeObject("rootTEPointPanel") )
                                        EndModifyTemplate(winggrid)
                                        winggrid.Name= "Grid." + ToString(i)
                               }



                               airfoilH =
CreateOrModifyTemplate("WingletCatalog|wingRootAirfoil",`Instantiated`
,`Relations\Knowledge Pattern.1\gridairfoil` ,i)
                               oldgrid = `Relations\Knowledge Pattern.1\winggrid`
->GetItem(i)
                               airfoilH->SetAttributeObject("apex",oldgrid-
>GetAttributeObject("rootLEPointPanel") )
                               airfoilH->SetAttributeObject("rootChord",oldgrid-
>GetAttributeObject("rootChordPanel") )
                               EndModifyTemplate(airfoilH)
                               airfoilH.Name = "AirfoilH." + ToString(i)


        gridpanel=CreateOrModifyTemplate("WingletCatalog|partitionHT",`Instantiate
d`,`Relations\Knowledge Pattern.1\gridPanel`,i )
                               oldgrid =`Relations\Knowledge Pattern.1\winggrid` -
>GetItem(i)
                               oldairfoil = `Relations\Knowledge
Pattern.1\gridairfoil` ->GetItem(i)
                               gridpanel-
>SetAttributeObject("rootChordLEPointIC",oldairfoil-
>GetAttributeObject("rootChordLEPointIC") )
                               gridpanel-
>SetAttributeObject("leadingEdge",oldgrid->GetAttributeObject("leadingEdgePanel") )
                               gridpanel-
>SetAttributeObject("rootChordNewZeroIncidenceIC",oldairfoil-
>GetAttributeObject("rootChordNewZeroIncidenceIC") )
                               gridpanel-
>SetAttributeObject("trailingEdge",oldgrid->GetAttributeObject("trailingEdgePanel") )
                               gridpanel->SetAttributeObject("ZXPlane",oldairfoil-
>GetAttributeObject("ZXPlane") )
                               gridpanel-
>SetAttributeObject("LEPointAirfoil",oldairfoil->GetAttributeObject("LEPointAirfoil") )
```

```
                        gridpanel-
>SetAttributeObject("increasedRootChordTEPoint",oldairfoil-
>GetAttributeObject("increasedRootChordTEPoint") )
                        gridpanel-
>SetAttributeObject("TEPointAirfoilUpper",oldairfoil-
>GetAttributeObject("TEPointAirfoilUpper") )
                        gridpanel-
>SetAttributeObject("TEPointAirfoilLower",oldairfoil-
>GetAttributeObject("TEPointAirfoilLower") )
                        gridpanel-
>SetAttributeObject("airfoilWithOutThicknessTERoot",oldairfoil-
>GetAttributeObject("airfoilWithOutThicknessTERoot") )
                        gridpanel-
>SetAttributeObject("rootChordTEPointIC",oldairfoil-
>GetAttributeObject("rootChordTEPointIC") )
                        gridpanel-
>SetAttributeObject("airfoilWithThicknessTERoot",oldairfoil-
>GetAttributeObject("airfoilWithThicknessTERoot") )
                        gridpanel-
>SetAttributeObject("increasedRootChord",oldairfoil-
>GetAttributeObject("increasedRootChord") )
                        gridpanel->SetAttributeObject("TEPoint",oldairfoil-
>GetAttributeObject("TEPoint") )
                        EndModifyTemplate(gridpanel)
                        gridpanel.Name = "GridPanel." + ToString(i)
                        oldgridpanel=`Relations\Knowledge
Pattern.1\gridPanel` ->GetItem(i)
                        oldgridpanel-
>SetAttributeString("airfoilType",`typeOfAirfoil` )
                }
        }
}

if `typeOfWinglet` == "upSwept"
{
        upswept=CreateOrModifyTemplate("WingletCatalog|upSwept",`Instantiated`
,`Relations\Knowledge Pattern.1\tipdevices` ,1)
        upswept->SetAttributeObject("apex",`External
References\tipChordLEPointNew` )
        upswept->SetAttributeObject("rootChord",`External
References\tipChordNewZeroTwistNew` )
        upswept->SetAttributeObject("tangentLE",`External References\LE` )
        upswept->SetAttributeObject("tangentTE",`External References\TE` )
        upswept->SetAttributeObject("yz plane",`External References\YZPlane` )
```

```
		upswept->SetAttributeObject("TEPointAirfoilUpper",`External
References\TEPointAirfoilUpper` )
		upswept->SetAttributeObject("TEPointAirfoilLower",`External
References\TEPointAirfoilLower` )
		upswept->SetAttributeObject("airfoilWithOutThicknessTERoot",`External
References\airfoilWithOutThicknessTE` )
		upswept->SetAttributeObject("LEPointAirfoil",`External
References\LEPointAirfoilNew` )
		upswept->SetAttributeObject("airfoilWithThicknessTERoot",`External
References\airfoilWithThicknessTE` )
		EndModifyTemplate(upswept)
		upswept.Name = "UpSwept." + ToString(1)
}

if `typeOfWinglet` == "drooped"
{
		drooped=CreateOrModifyTemplate("WingletCatalog|drooped",`Instantiated`
,`Relations\Knowledge Pattern.1\tipdevices` ,1)
		drooped->SetAttributeObject("apex",`External
References\tipChordLEPointNew` )
		drooped->SetAttributeObject("rootChord",`External
References\tipChordNewZeroTwistNew` )
		drooped->SetAttributeObject("tangentLE",`External References\LE` )
		drooped->SetAttributeObject("tangentTE",`External References\TE` )
		drooped->SetAttributeObject("yz plane",`External References\YZPlane` )
		drooped->SetAttributeObject("TEPointAirfoilUpper",`External
References\TEPointAirfoilUpper` )
		drooped->SetAttributeObject("TEPointAirfoilLower",`External
References\TEPointAirfoilLower` )
		drooped->SetAttributeObject("airfoilWithOutThicknessTERoot",`External
References\airfoilWithOutThicknessTE` )
		drooped->SetAttributeObject("LEPointAirfoil",`External
References\LEPointAirfoilNew` )
		drooped->SetAttributeObject("airfoilWithThicknessTERoot",`External
References\airfoilWithThicknessTE` )
		EndModifyTemplate(drooped)
		drooped.Name = "Drooped." + ToString(1)
}
if `typeOfWinglet` == "hoerner"
{

		hoerner=CreateOrModifyTemplate("HoernerCatalog|hoerner",`Instantiated`
,`Relations\Knowledge Pattern.1\tipdevices` ,1 )
```

```
                hoerner->SetAttributeObject("apex",`External
References\tipChordLEPointNew` )
                hoerner->SetAttributeObject("rootChord",`External
References\tipChordNewZeroTwistNew` )
                hoerner->SetAttributeObject("xy plane",`External References\XYPlane` )
                hoerner->SetAttributeObject("airfoilWithThicknessTERoot",`External
References\airfoilWithThicknessTE` )
                hoerner->SetAttributeObject("LEPointAirfoil",`External
References\LEPointAirfoilNew` )
                hoerner->SetAttributeObject("TEPointAirfoilUpper",`External
References\TEPointAirfoilUpper` )
                hoerner->SetAttributeObject("TEPointAirfoilLower",`External
References\TEPointAirfoilLower` )
                hoerner->SetAttributeObject("airfoilWithOutThicknessTERoot",`External
References\airfoilWithOutThicknessTE` )
                EndModifyTemplate(hoerner)
                hoerner.Name = "Hoerner." + ToString(1)
}
if `typeOfWinglet` == "tipExtension"
{
                tipextension=CreateOrModifyTemplate("WingletCatalog|tipExtensionHorizonta
l",`Instantiated` ,`Relations\Knowledge Pattern.1\tipdevices` ,1)
                tipextension->SetAttributeObject("apex",`External
References\tipChordLEPointNew` )
                tipextension->SetAttributeObject("tangentLE",`External References\LE` )
                tipextension->SetAttributeObject("rootChord",`External
References\tipChordNewZeroTwistNew` )
                tipextension->SetAttributeObject("tangentTE",`External References\TE` )
                tipextension->SetAttributeObject("airfoilWithOutThicknessTERoot",`External
References\airfoilWithOutThicknessTE` )
                tipextension->SetAttributeObject("airfoilWithThicknessTERoot",`External
References\airfoilWithThicknessTE` )
                tipextension->SetAttributeObject("LEPoint",`External
References\LEPointAirfoilNew` )
                tipextension->SetAttributeObject("TEPointAirfoilUpper",`External
References\TEPointAirfoilUpper` )
                tipextension->SetAttributeObject("TEPointAirfoilLower",`External
References\TEPointAirfoilLower` )
                EndModifyTemplate(tipextension)
                tipextension.Name = "TipExtension." + ToString(1)
                oldtipextension=`Relations\Knowledge Pattern.1\tipdevices` ->GetItem(1)
                oldtipextension->SetAttributeString("airfoilType",`typeOfAirfoil` )
}
```