Conceptual Design Modeling in the RDS-Professional Aircraft Design Software

AIAA Paper #2011- 161 (updated 2023 by the Author)
AIAA Aerospace Sciences Meeting, Orlando, FL, 2011

Daniel P. Raymer, Ph.D., Fellow AIAA
CONCEPTUAL RESEARCH CORPORATION
PO BOX 5429, Playa del Rey, CA, USA, 90296-5429
draymer@aircraftdesign.com http://www.aircraftdesign.com

ABSTRACT

The geometry mathematics of the Design Layout Module of the RDS-Professional aircraft design software is described. This includes the modeling mathematics employed for surfaces and local coordinate systems, the use of an expanded version of the RP8A category numbers as a way to couple design and analysis, and the use of this modeling framework to implement automatic geometric redesign from sizing and MDO results.

NOMENCLATURE

A	= Aspect Ratio (span^2/reference area, applied to wings and tails)
CDS	= Rockwell Configuration Development System (CAD program)
L/D	= Lift-to-Drag Ratio
NURB	= Non-Uniform Rational B-Spline
M	= Mach Number (velocity relative to speed of sound)
MDO	=Multidisciplinary Design Optimization
RDS	= Aircraft design software package *("Raymer's Design System")*
T/W	= Thrust-to-weight ratio
W_e	= Aircraft Empty Weight
W_o	= Aircraft Takeoff Gross Weight
W/S	= Wing loading (weight/area)

INTRODUCTION

The RDS-Professional[1] aircraft design software, developed by this author and marketed through Conceptual Research Corporation, is an integrated design environment which includes a design layout module for concept development, analysis modules for aerodynamics, weights, propulsion, stability, cost, performance, range, and sizing, and an optimization module including classical carpet plots as well as Multidisciplinary Design Optimization. RDS has been developed over a 30-year period, and is used in industry, government, and academia. The RDS-Student version is widely used in universities. Not a spreadsheet or similar "soft" implementation, RDS is a "hard-coded" program of 110,000+ source lines that performs a full range of aircraft conceptual design tasks. The technical methods employed in RDS are largely based on those described in this author's textbook *Aircraft Design: A Conceptual Approach*[2].

The Design Layout Module of RDS is an all-original code which permits rapid aircraft configuration layout using mouse-driven interactive computer graphics, allowing the designer to develop a new concept or modify the previous baseline design in a methodology custom-tailored for the advanced aircraft design environment. The design capabilities of this module include wings, tails, fuselages, nacelles, seats, canopies, and other required components. RDS-DLM allows interactive assembly of the aircraft using top, front, or side view, or even in an isometric, orthographic, or perspective view.

RDS-DLM is uniquely suited to Aircraft Conceptual Design because it knows what an airplane is. RDS-DLM has dozens of airplane-specific design capabilities to rapidly create, modify, and analyze an aircraft concept. RDS-DLM uses the aircraft industry SAWE RP8A (Mil-Std-1374a) Group Weight Statement component categories to identify the nature of the design's various parts and to simplify the analysis interface.

When a design is completed, RDS-DLM provides a simple data interface to the Aerodynamics, Weights, and Propulsion Modules and hence to all RDS performance analysis and optimization. RDS-DLM also produces the tabular geometric information required for reports and further analysis, and produces export files to transfer the design to other CAD and analysis programs.

This paper describes the key equations and methods used for geometric modeling in RDS, including surface definition, component local axis implementation, and RP8A implementation during and after aircraft component creation.

AIRCRAFT COMPONENTS

Perhaps the most fundamental feature of the RDS design layout module that facilitates its use in aircraft design is its use of “components” as the basic unit of data storage. This author recalls frustration dealing with commercial CAD systems that treat each piece of geometry as a separate entity, to be separately developed, manipulated, moved, saved, and deleted. While it is possible to “aggregate” regions into a single entity, it isn’t the default and it isn’t always easy.

In RDS, the basic unit of data storage is the component, and those are based on standard aircraft terminology - wing, tail, fuselage, tire, etc… Normally each component represents a single closed object.

Each component in RDS has its own local axis system (see below) and its location and orientation within the aircraft global axis system is readily changed. Each component has a header file with information used in RDS to set display options, define component symmetries, record creation date, and other information such as the original reference geometry for wings and tails. Then, of course, each component file includes the actual component geometry which is stored as YZ cross sections stacked in the X direction, using either point or Bezier representation (see below). Non-planar cross sections are also possible, with numerous tools to define them. Components can be acquired from other aircraft files or from stored “component banks” and can be either copied or instanced (a copy that uses the geometry stored in another component but has its own location data).

To simplify assembly and modification of related components, “groups” can be defined but these do not affect the basic component data storage scheme. Groups are basically nothing more than shortcuts for component pick menus.

SURFACE REPRESENTATION - POINTS

For wings and tails, RDS sticks to basics. When first being created, such surfaces are always defined by airfoil coordinate points, appropriately scaled and stretched to the desired chord length and location. Rather than attempting a mathematical surface fit and incurring potential losses in accuracy as the airfoils are warped into position, RDS simply leaves them as actual XYZ points. At a later point in the design process, of course, these points will be fit to surfaces and used to create a true solid model, but this level of accuracy is not required during conceptual design studies and in fact, slows the process down.

This simple representation can also be used for other types of components. However, most other components such as the fuselage are represented with a parametric equation as described in the next section.

SURFACE REPRESENTATION - QUARTICS

While at Rockwell North American Aviation in the late 1970’s, this author had the privilege of heading the project that developed the “CDS” aircraft design system later used to design the B-1B and X-31, among others. After reviewing a number of alternative approaches for geometric surface representation, a variation of the parametric 4th degree polynomial Bezier Curve was selected[3]. The mathematics for this so-called “Quartic” curve[4,5,6] and surface had been developed by Mr. Robert Maier, head of Master Dimensions at Rockwell NAA, largely as a side hobby at the time but later implemented into a production lofting program that was extensively used prior to the eventual switch to commercial CAD systems.

A Bezier curve is a polynomial subset of the generic Non-Uniform Rational Basis-Spline (NURB) widely used in modern high-end CAD systems. These control the shape of a curve or surface by defining the location of “control points” which are actual XYZ points somewhere in space. This makes Beziers and NURBs perfect for visual design applications because the designer can simply move the points around to obtain the desired shape.

The NURB, as the superset, offers greater shape control and the ability to exactly match more complicated shapes, at the expense of additional computation and the need for additional inputs by the user. It is common in today’s CAD systems to use NURBs, but to simplify the input requirements by “hiding” much of the NURBs full capabilities from the users until absolutely needed for some complicated task.

A Bezier or NURB curve has two endpoints which are exactly on the resulting curve. The control points next to the endpoints mathematically define the end tangents of the curve, and also set the curvature at the ends by their distance from the endpoints.

In a true Bezier or NURB, the rest of the control points are "floating" out in space and do not actually lie on the curve. They are usually described as "magnets" which "pull" the curve towards themselves[*]. Thus, the user must attain the desired curve shape by a sort of trail and error process, moving around the cloud of points until the curve looks right. With practice, this becomes second nature.

For a 4th degree Bezier there are exactly five control points, which are equivalent in the equations to the five coefficients of a normal 4th degree polynomial. Since the endpoints and tangent control points number four, there is one and only one point "floating" in the middle.

The great change in Maier's unique implementation was that he mathematically moved this middle floating point to lie exactly on the resulting curve, and furthermore to lie exactly at the parametric midpoint of the curve. This turns design into child's play - simply locate the endpoints, set the tangent angles, and place the middle point exactly where you want the curve to go. Then if desired you can "play" with the curvature at the ends by sliding the tangent control points in and out, along the desired tangent angles.

This implementation also makes the Quartic[†] look remarkably like a classical conic to the designers, despite its far-greater power. Both curves have two endpoints and an on-the-curve center "shoulder" point. Both have lines from the endpoints that control the tangent angles. The only difference is that in the conic, these tangent directions are controlled by a single point representing the tangent intersection. In the Quartic, each endpoint has its own point controlling tangent direction and they can be placed independently, even on opposite sides of the desired curve. Furthermore, those tangent control points also define the curvature at the endpoints by their distance away from their endpoint – if they are moved farther away, the curve is pulled more and hence the curvature is reduced.

For CAD work a parametric version of the equation is desired so that there is no problem with double-valued points. The generic parametric 4th degree polynomial is:

$$X(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4$$
$$Y(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4$$
$$where \quad 0 \to t \to 1$$

The Maiers Quartic calculation starts by defining the five control points P_1 to P_5 as shown in figure 1. P_1 & P_5 are End Points, P_2 and P_4 are Slope Control Points, and P_3 is a Shoulder Point. To trace out the curve, the parametric variable *t* is increased from 0 to 1, applying it to weighting functions as follows:

$$f_1(t) = (1-t)^2(1-2t) \qquad f_4(t) = -4t^2(1-t)(1-2t)$$
$$f_2(t) = 4t(1-t)^2(1-2t) \qquad f_5(t) = -t^2(1-2t)$$
$$f_3(t) = 16t^2(1-t)^2$$

Then the X, Y, and Z values of points on the curve can be found by calculating *W (t)* below with the X, Y, and Z values of the five control points.

$$W(t) = f_1(t)P_1 + f_2(t)P_2 + f_3(t)P_3 + f_4(t)P_4 + f_5(t)P_5$$

It is surprising how simple this equation is, how easy to program, and how powerful in application. This can be seen in figure 2, showing a stealth aircraft fuselage defined by just two quartics. By moving just a few points in the upper quartic, substantially different shapes are created.

In both CDS and RDS, the 4th degree polynomial curve is used exclusively. Even a straight line is modeled as a 4th degree polynomial. While in some cases this seems overkill, keeping to a single curve expression results in simplified programming and greater design flexibility. For example, what if a surface you'd hoped would be flat needs to be bulged out later during the design development?

At the other extreme, a 4th degree polynomial can only represent a double-reflexed curve. If more reflexes are required, the user or the computer will have to break

[*] This is a rather flawed analogy. A magnet produces a greater pull when it is closer to the object it is attracting, whereas control points pull harder when farther away. A better analogy might be to imagine stretchy rubber bands running from the control points to the curve.

[†] In RDS this is renamed a "SuperConic" to better make the connection to classic conic lofting.

the curve into several polynomials. However, for most design applications two reflexes should be sufficient. Classical conic lofting as used up to the 1970's allowed no reflexes in a single curve, so permitting two seemed the height of luxury.

Extending this Quartic curve to a 3-D surface turns out to be simple as well. Just as five points define a Quartic line, so five lines define a quartic surface. Thus, 25 points uniquely define a bounded surface (figure 3) using parametric variables *s* and *t*, each varying from 0 to 1. The math for calculation of points on the surface is straightforward matrix multiplication. In a method similar to that used for the curve, the X, Y, and Z values of points can be found by calculating *W(s,t)* below with the X, Y, and Z values of the 25 control points shown in the figure, as follows:

$$W(s,t) = \left[f_1(s) + f_2(s) + f_3(s) + f_4(s) + f_5(s)\right] \times \begin{bmatrix} P_1 & P_2 & P_3 & P_4 & P_5 \\ P_6 & P_7 & P_8 & P_9 & P_{10} \\ P_{11} & P_{12} & P_{13} & P_{14} & P_{15} \\ P_{16} & P_{17} & P_{18} & P_{19} & P_{20} \\ P_{21} & P_{22} & P_{23} & P_{24} & P_{25} \end{bmatrix} \begin{bmatrix} f_1(t) \\ f_2(t) \\ f_3(t) \\ f_4(t) \\ f_5(t) \end{bmatrix}$$

The weighting functions *f ()* are the same as those for the line, using the appropriate value of *s* or *t*.

When applied to the design of typical aircraft components such as a fuselage it is convenient to store the 25 control points as five stacked cross sections. These are normally parallel and planar but the mathematics isn't so restricted. Just as the second and fourth points of a quartic line control the end tangencies and curvatures, so the second and fourth stored cross sections control the surface tangencies and curvatures. As a result, the even numbered cross-sections from nose to tail are not actually on the surface, instead forming "collars" to the edges of the surface patches. This gives great power and flexibility for the designer.

This simple method, using control points which resemble those of the time-honored conic loft methods, allows the designer to produce design surfaces of incredible complexity. These methods were used exclusively to design the X-31 and the B-1B, and are now implemented in RDS.

LOCAL AXIS SYSTEM IMPLEMENTATION

Once the component surfaces have been defined as described above, the next issue is the use of local coordinate axis systems. This has a long and honored history, going back at least to the P-51 and probably to the Wright Flyer. The designers will design the various components in an axis system that makes it easy to design that component, then will use some coordinate transformation scheme to rotate that component into the global axis system.

For example, engine nacelles for commercial jet aircraft are slightly tipped with respect to the fuselage. It would be an unnecessary complication to design the nacelle in the fuselage axis system, since the obvious perpendicular nacelle cross sections would not line up with the fuselage cross sections.

CAD systems usually permit a local axis system for different geometric entities. In RDS this is accomplished by defining individual components, each of which has its own local axis system which is potentially offset and/or rotated with respect to the global axis system.

There are two methods of coordinate transformation in wide usage - Quaternions, and Homogenous Coordinate Transformations (of which Direction Cosines are a subset). Quaternions are very popular in modern CAD systems for display rotations, and can be recognized when the mouse produces rotations by up-down and side-to-side motions that seem to flip the display image over almost at random. The experienced user learns how to combine mouse motions to get the desired display direction, even if several non-obvious intermediate rotations are required.

Homogenous Coordinate Transformations use sine and cosine matrix relationships based upon input angles of roll, pitch, and yaw, plus translation distances in three directions. A single 4x4 "R" matrix comprising all required rotations and translations is created and used, with an additional dummy dimension, in a multiplication/division operation. For display purposes, perspective distortions can also be obtained in the same calculation. All required manipulations including local axis system, global display rotations, and even perspective manipulations are concatenated into a single "R" matrix prior to their application to actual points. The mathematics is widely available[7] and will not be repeated here.

Homogenous Coordinate Transformations require a three-axis input scheme rather than the two-axis mouse motions of Quaternions. When used as component local coordinate system transformations, these allow precise input of rotations and translations in a manner expected by designers, and are therefore used in RDS. Each component has its own axis location XYZ and axis rotation (roll, pitch, and yaw). Roll and pitch can be modified interactively using the mouse or arrow keys, with yaw being changed by holding down the Alt key while moving mouse or arrow keys side to side. Axis orientations can also be entered numerically.

One problem with the use of homogenous coordinate transformations for local axis systems is the gyro lock problem. If components are defined as YZ cross sections stacked in the X direction (as in RDS), a component such as a landing gear strut might be pitched by 90 degrees, then rolled for outward splay, and then yawed for alignment. If an additional minor adjustment in orientation is required, it may no longer be obvious how to obtain it with changes to the existing angles[‡]. While there is always some combination of roll, pitch, and yaw that will attain the desired final orientation, it is sometimes the case that you "can't get there from here" and have to start over again.

In RDS this is handled by allowing components to be "pre-rotated." Wings, tails, landing gear, and other components can have yaw and pitch rotations applied prior to the user's desired alignment rotations. These pre-rotations are concatenated into the component's "R" matrix so they add only a minuscule amount to the calculation time.

Pre-rotations are shown in figure 4. A wing (left side only) is shown in its actual coordinate system, namely airfoils stacked in the X direction. Next it is shown pre-rotated by 90 degrees in yaw, like most wings. The third image shows an additional rotation of 90 degrees in pitch, typical for a vertical tail. The final illustration shows how a ventral fin is pre-rotated by 90 degrees in yaw followed by negative 90 degrees in pitch.

COMPONENT SYMMETRIES

Aircraft have obvious and routine symmetries which an aircraft design CAD system should recognize and easily implement. The most prevalent is the global left-right symmetry seen in most of the airplanes ever designed. Wings, tails, the fuselage, and most other components are mirrored across the global X-Z centerline plane.

In addition, many components have symmetries of their own. For aircraft such symmetry is almost always across the component's XZ plane, i.e., left-right. Both global and local left-right symmetry are supported in RDS and easily defined. In figure 5 an engine nacelle illustrates these option. First is the actual stored data with no symmetry enabled. Next the stored geometry is reflected across the global aircraft centerline plane of symmetry. Third, the stored geometry is reflected across the nacelle's axis system. Finally, both symmetry options are enabled creating two complete nacelles.

In some rare cases there is symmetry about some other plane such as top-bottom, or even radial symmetry. RDS supports one more symmetry, that across the root airfoil (YZ plane at X=0). This is used for wings and tails that are left-right symmetric, but are positioned off the aircraft's global centerline (see figure 6). Another usage is for a skewed wing in which the left and right wing panels are mirror imaged but reflected across a skewed plane.

SAWE RP8A FOR ANALYSIS AUTOMATION

As mentioned above, RDS "knows" what an aircraft is. RDS has numerous features just for aircraft design, such as "canned" routines for initial creation of many typical aircraft components including wings, tails, fuselages, nacelles, engines, landing gear, and more. Furthermore, when transitioning from the design layout geometry to the analysis modules, RDS automatically "knows" which sort of analysis is appropriate for which components.

This is accomplished with a detailed component type code scheme. Since one of the analysis outputs of RDS is a group weight statement in the format specified by the SAWE specification RP8A (previously Mil-Std-1374) it was decided to use that as the basis of the RDS component type code.

RP8A defines component weights by type using a three-digit scheme. However, RP8A doesn't go far enough to distinguish between different components and analysis methods so an additional three-digit string was appended. Be advised that this extended scheme is not considered industry standard practice, but this author has found it very useful. A small sampling of these codes can be seen below: For the full set please contact the author.

[‡] A major advantage of quaternions is that they can't experience mathematical gyro lock, but unfortunately they don't allow component alignment by the input of simple three-axis rotation angles.

(sample extended RP8A codes)

- 002-000:Ref Wing
- 002-003:LEX
- 002-004:Winglet
- 002-005:Wing Strut
- 002-999:Wing-Other
- 008-000:Aileron
- 008-001:Elevon
- 009-000:Spoiler
- 010-000:Flaps(TE)
- 011-000:Flaps(LE)
- 012-000:Slats
- 031-000:Fuselage
- 031-001:Canopy
- 085-000:Instruments
- 086-000:Hydraulics
- 087-000:Pneumatics
- 088-000:Electrical
- 090-000:Avionics

As an example of their usage, when a wing is created the user is prompted to select the type of wing. If *002:000:Ref Wing* is selected, that code is stored with the wing component. When the design is completed and the geometric information is collected for aerodynamic analysis, RDS will recognize from this code that the area of this wing is to be used as the reference area for the calculated aerodynamic coefficients. Furthermore, the *002:000* code tells the Weights Module to use a certain wing weight equation, and the code knows which geometric information to extract from the component to populate the weight analysis input fields.

This is all simple and obvious, but when you are designing and analyzing a new aircraft concept it is nice to have this "dumb" stuff done automatically.

AUTOMATIC REDESIGN FROM SIZING AND OPTIMIZATION RESULTS

One unique capability of the RDS CAD module is its ability to automatically modify a design based on the results of sizing analysis and design optimization. This allows the designer to instantly return to the design layout, see the affect of the changes, and fix or modify them as desired. The automatic redesign is "smart" as will be described below, and was detailed in an oral-only presentation[8] at the 2009 AIAA Aerospace Sciences Meeting.

The configuration changes needed to match a calculated change in sized takeoff gross weight affect almost every aspect of the design, from tails to tires, from inlet to engine, from wing to wheels. RDS effects automatic changes based of takeoff gross weight as follows:

- Fuselage scaled by cube root of weight ratio, unless length constrained by user input
- Wing area scaled proportional to weight ratio
- Tails scaled by 3/2 power then adjusted based on change in fuselage length, to hold constant the tail volume coefficient
- Engine scaled assuming T/W constant, using empirical exponents for diameter and length vs. thrust ratio (unless disabled)
- Nacelle and inlet duct scaled in diameter by square root of thrust ratio
- Wheels and tires scaled based on statistical tire diameter and width equations
- Gear shock-strut diameters scaled by square root of weight ratio
- Ground plane and tail-down angle components scaled proportional to fuselage scaling

RDS can also automatically modify the entire design layout to match optimization results. The RDS MDO routine optimizes for a total of eight key design parameters, namely T/W, W/S, wing planform parameters (aspect ratio, taper ratio, and sweep), wing thickness, wing design lift coefficient (surrogate for camber), and fuselage fineness ratio.

The optimal results are stored in a suitably-formatted file and, if the user requests, applied to the CAD file. This involves the automatic application of various scaling and stretching laws to revise each aircraft component to reflect the new optimum design, including recalculated takeoff gross weight, thrust, wing area, wing geometry, tail areas, fuselage length and diameter, plus changes to various other components such as engines and landing gear. The extended SAWE8 codes described above are used to select the appropriate scaling laws to apply to each component.

The most difficult part of this CAD file revision is the modification of a wing or tail by changes to its trapezoidal reference wing parameters (area, aspect ratio, taper ratio, and sweep) while preserving any non-trapezoidal modifications such as planform breaks, strakes, wingtip shaping, and the like. This is done using geometric transformation equations derived by this author, resulting in appropriate scalings of airfoils in Y and Z, and moving of airfoils in X. This is also applied to any components that have been created from the original wing or tail such as fuel tanks, flaps, ailerons, spars, and the like.

An example is shown in figure 7, where aspect ratio and sweep of a wing have been changed. The non-trapezoidal features of the original wing design are preserved, scaled proportionally to the changes in the planform parameters. This is done in addition to the scaling for the optimized takeoff weight as described above.

Note that this requires no up-front establishment of parametric design relationships and takes no additional time during initial design or after optimization. It is all automatic and instantaneous.

Properly done, such a procedure can significantly reduce the time to complete a design iteration ("Dash-1" to "Dash-2"). While these operations cannot be expected to produce perfect "buildable" geometry, they can do most of the "grunt work" associated with revising a design layout to match improved design parameters. A typical example is shown in figure 8.

SUMMARY & CONCLUSIONS

The geometric mathematics and methods of the RDS-Professional Design Layout Module have been described, including Quartic surface mathematics, local axis system transformations, use of an augmented SAWE RP8A component type code, and automatic redesign from optimization results. These methods have resulted in a powerful set of design tools for the initial conceptual design of an aircraft and for the critical design modifications as the configuration is iterated towards a final baseline.

FIGURES:

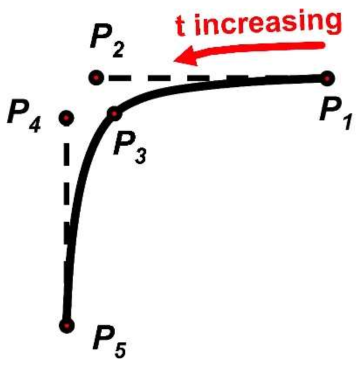

figure 1. Quartic Curve Control Points

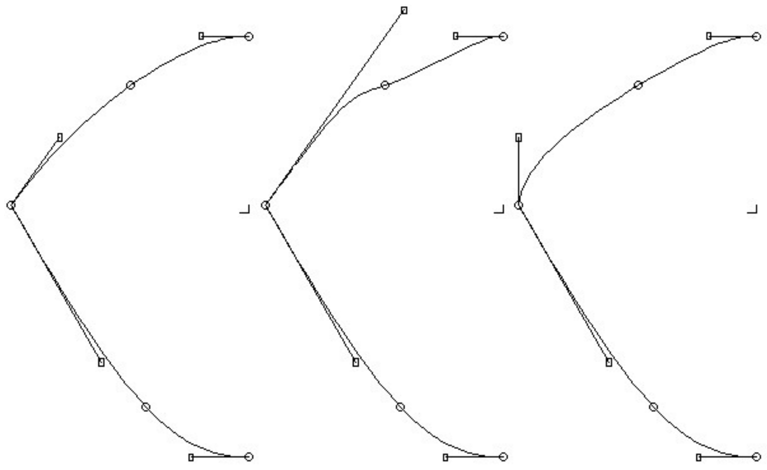

figure 2. Quartic Curve Examples

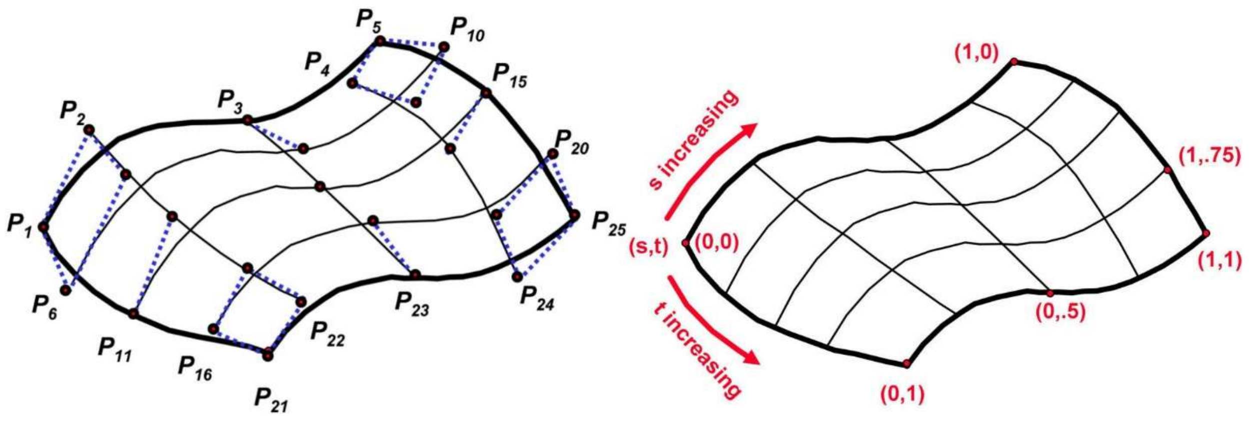

figure 3. Quartic Surface Control Points

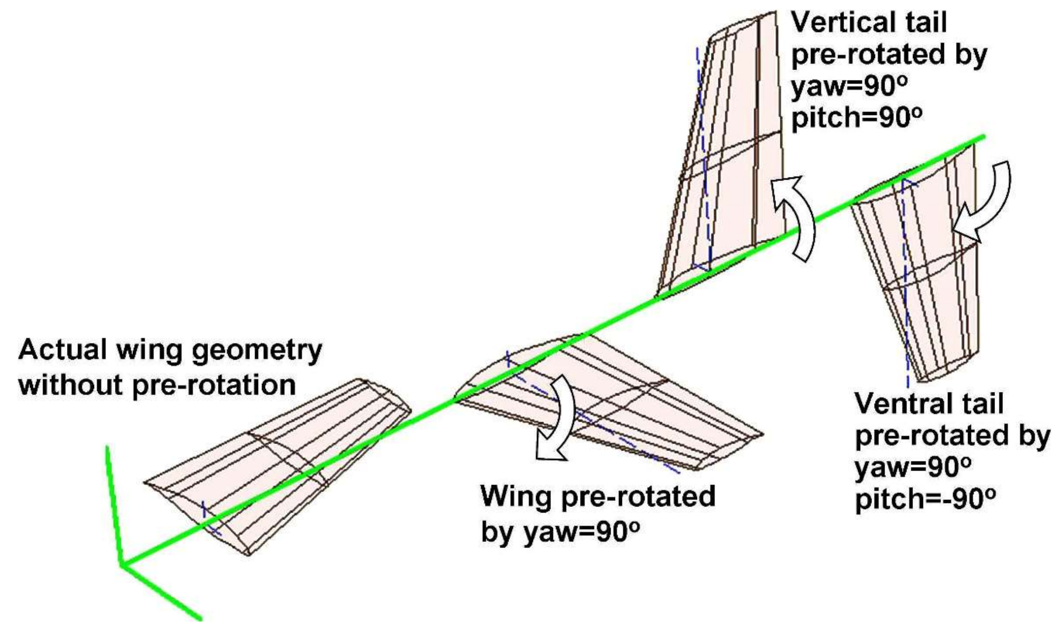

figure 4. Prerotation of Wing-like Components

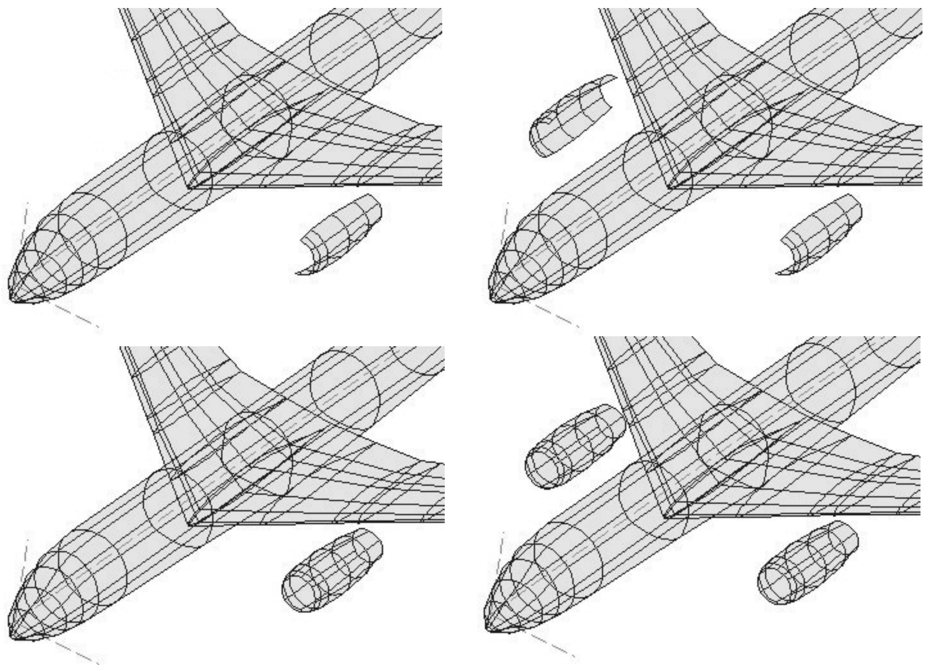

figure 5. Symmetry Options

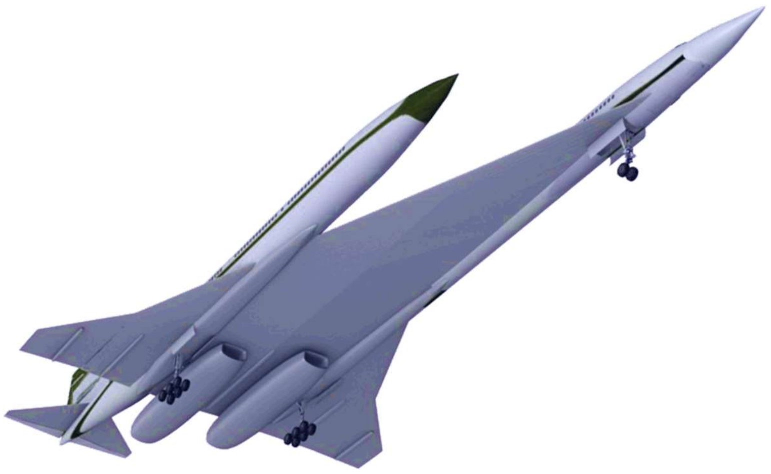

figure 6. Unusual Symmetry - Horizontal Tail

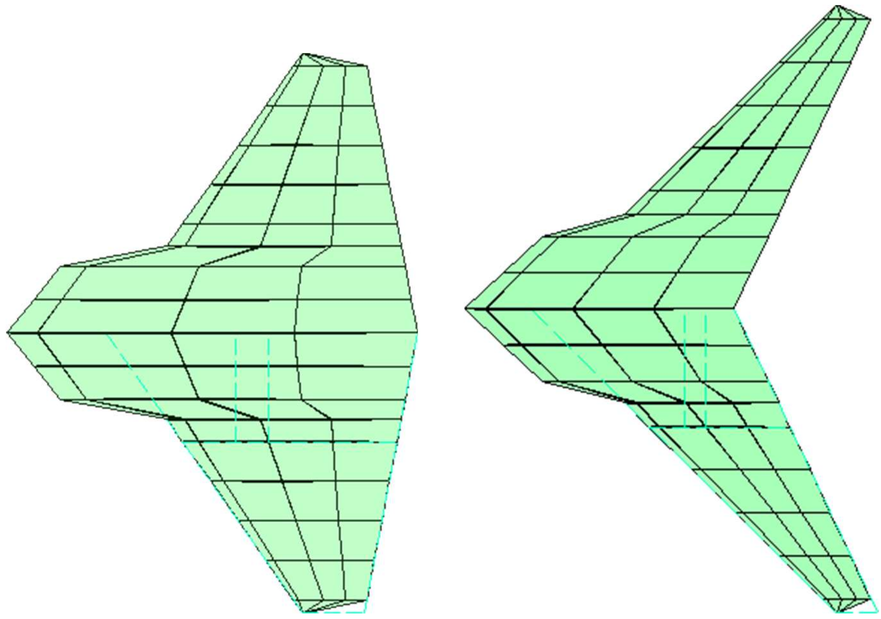

figure 7. Wing Redesign by Trapezoidal Parameter Revision

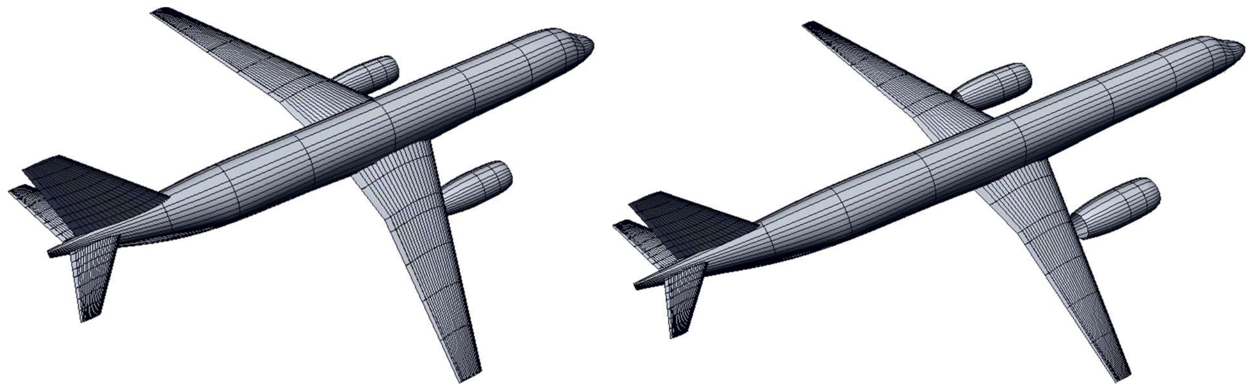

figure 8. Automatic Redesign of Airliner (before-after)

REFERENCES

[1] Raymer, D., **"RDS: A PC-Based Aircraft Design, Sizing, and Performance System,"** AIAA Paper 92-4226, Aug. 1992

[2] Raymer, D., ***AIRCRAFT DESIGN: A Conceptual Approach*,** American Institute of Aeronautics and Astronautics, Washington, D.C., Third Edition 1999

[3] Raymer, D., ***Living In The Future: The Education and Adventures of an Advanced Aircraft Designer***, Design Dimension Press, Los Angeles, CA, 2009

[4] Maier, Robert., "**Quartic Curves**", Rockwell/North American Aviation TFD-78-718, Los Angeles, CA 1978

[5] Maier, Robert., "**Master Dimensions Primer**", Rockwell/North American Aviation TFD-81-356, Los Angeles, CA 1981

[6] Liming, R., Matlock, D., and Maier, R., "**Master Dimensions Group Calculation Handbook**", North American Aviation NA-53-670, Los Angeles, CA 1953 with regular updates to ~1970

[7] Raymer, D., **Use of Computers in the (Conceptual) Design Process**, Lecture notes, University of Dayton Short Course in Aircraft Design, 1982 (available at www.aircraftdesign.com)

[8] Raymer, D., **Automatic Aircraft Configuration Redesign - the Application of MDO Results to a CAD File**, AIAA Aerospace Sciences Meeting Paper 2009-1659 (oral), Reno, NV, 2009