1-1-2006

Parametric representation of turbine blades and vanes internal cooling geometry

Jamal Zeinalov
Ryerson University

PARAMETRIC REPRESENTATION OF TURBINE BLADES AND VANES INTERNAL COOLING GEOMETRY

By

Jamal Zeinalov

Bachelor of Engineering
Aerospace Engineering
Ryerson University, 2004

A thesis presented to

Ryerson University

in partial fulfillment of the

requirements for the degree of

Master of Applied Science

in the Program of

Mechanical Engineering

Toronto, Ontario, Canada, 2006

Author's Declaration

I hereby declare that I am the sole author of this thesis or dissertation.
I authorize Ryerson University to lend this thesis or dissertation to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis or dissertation by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Abstract

PARAMETRIC REPRESENTATION OF TURBINE BLADES AND VANES INTERNAL COOLING GEOMETRY

© Jamal Zeinalov
Master of Applied Science
Mechanical Engineering
Ryerson University
Toronto, Ontario, Canada, 2006

Modern turbine blades and vanes possess very complex internal geometry. As a result of this complexity the design through conventional means, such as a Computer Aided Design (CAD) package, can take a large portion of the allocated time, preventing thorough testing and optimization. The aim of this thesis is to produce a parametric design methodology that can be used to create turbine blade and vane geometry from designer-specified parameters. This work includes a thorough study of the current blade and vane design process, an in-depth analysis of the most commonly used geometry representation methodologies, as well as a review of existing works on the subject. The research code, based on the created design methodology, was tested on the existing blade and vane design schemes. The result of this comparison is presented. The developed methodology can be used to shorten the design time necessary to produce the blade and vane geometry, thereby increasing the time available for analysis and optimization.

Acknowledgements

I would like to thank my supervisors, Dr. Zouheir Fawaz and Dr. Kamran Behdinan, for their support and counsel.

At Pratt & Whitney Canada I would like to thank Dr. Sri Sreekanth for his help and sound advice with methodology development and research code testing and analysis. I would also like to thank Terry Lucas for his support of the project and Mike Papple for the explanation of the vane and blade design process.

I would also like to thank Art Hammershmidt for his invaluable advice and help with geometry generation algorithms, mathematical representation methodologies and Nlib library routines.

TABLE OF CONTENTS

List Of Figures

List Of Tables

Nomenclature

β_1	Bias parameter in β-spline
β_2	Tension parameter in β-spline
ν	Parametric unit in 3-D geometry
υ	Parametric unit in 3-D geometry
P	Control point
N	B-spline/NURBS basis function
t	Parametric unit in 2-D geometry
B	Bernstein polynomial
U	Knot vector
B_r	β-spline basis function
w	Control point weight

Abbreviations

CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
FEA	Finite Element Analysis
NURBS	Non-Uniform Rational B-Spline

CHAPTER 1 Introduction and Overview

This chapter provides an introduction into the topic of the thesis as well as an overview of the research and analysis process conducted to achieve the objective.

1.1 Research Summary

The topic of this thesis was to develop a parametric modeling scheme for internally cooled turbine blades and vanes and demonstrate the effectiveness of such scheme for a robust design generation. This has been achieved by an investigation into the existing design practices, as well as an in-depth study of parametric representation techniques in the context of Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM) systems. The result of this study is a parametric design methodology capable of constructing internal cooling geometry of turbine blades and vanes from the initial parametric information supplied by the designer. This scheme was tested for design accuracy and robustness by recreating existing designs.

Parametric representation is an important element in geometry creation. It plays an important role itself, by allowing a fast and efficient way of creating geometry. In addition it is also an important part of an optimization algorithms, where a robust parametric scheme is required for fast geometry regeneration. This is especially true for turbine blades and vanes, where the geometry can be quite complex and the geometry generation by usual design means (i.e. through conventional CAD interface) can take prohibitively long time, shortening the time available for design deliberation and optimization routines. As such, the parametric representation scheme described in this work was created with the following objectives in mind:

- General parametric scheme encompassing the existing design methodologies as well as anticipated future developments.
- Robust algorithm that allows for fast geometry generation (ideally the designer should be able to generate complete components within several hours).
- Compatibility with existing CAD packages.

1.2 Thesis Outline

This thesis is arranged in such an order as to reflect the natural path taken by the investigation. In Chapter 2 a brief general turbine engine overview is given as well as an in-depth analysis of the turbine blade and vane operating environment. The focus of the chapter is on the thermal, aerodynamic and structural considerations as well as manufacturing constraints imposed on the turbine components that dictate the need for the existing design and geometric complexity. Current design procedures are covered as well and the need for the parametric representation methodology is examined.

In Chapter 3 a study of the most widespread forms of mathematical representation (particular to CAD/CAM environment) is presented. The aim of this chapter is to identify the optimal underlying form of geometric representation that would accommodate the design scope as well as guarantee compatibility with existing CAD packages. The forms of representation covered in this chapter include B-splines, β-splines, Bezier curves and Non Uniform Rational B-Splines (NURBS).

The parametric methodology developed during this work is presented in Chapter 4. The development process includes the study of the existing parametric schemes developed over last decade, the existing design methodologies as well as some speculative analysis of new design approaches. The development process also incorporates a testing stage as well, where the parametric design methodology is tested against some of the existing components. Finally, Chapter 5 provides an analysis of the results and an overview of future work necessary in this area.

As can be seen from the topics covered here, the resulting methodology incorporates much more than just a set of parametric variables. In fact it also encompasses all of the considerations regarding optimal mathematical representation, robustness, compatibility and an ability to expand the system to include additional geometric capabilities. The resulting product is a methodology that allows for a robust creation of the turbine blade and vane geometry that has been demonstrated to work in the context of established as well as recently developed design schemes.

The research code that has been developed on these principles and used to validate the model, has been developed using Microsoft Visual C++ V 6.0 © Microsoft Corporation with the aid of the following libraries:

- **Nlib™ © 1998-2002 Solid Modeling Solutions.**
  The NURBS Library provides an extensive set of routines for constructing and manipulating NURBS curves and surfaces. The algorithms of NLib are based on *The NURBS Book* by (Piegl and Tiller, 1997).

- **GSNlib™ © 1998-2002 Solid Modeling Solutions.**
  The General Surface NURBS Library is an object-oriented software toolkit, based on NLib, with methods to create, edit, query and intersect NURBS curves and surfaces. GSNLib is designed to support the development of a wide variety of NURBS based surface applications.

A pseudo-code of the developed algorithm is included in Appendix A.

CHAPTER 2 Turbine Engine Operation Overview

This chapter gives a brief overview of the turbine engine operations. The focus is then given to the turbine operating environment, including thermal and aerodynamic conditions, as well physical characteristics of the turbine blade and vane geometry. The main purpose of this chapter is to show the effect of the operating environment and manufacturing constraint on the turbine blade and vane geometry.

2.1 Overall Principles of Operation

A general gas turbine engine core set up is described in Figure 2.1. The air enters the engine core through the inlet and then passes through the compressor, which slows the air down and increases the pressure. The air then enters the combustor where it is mixed with fuel and burned, adding energy. Air is then released into the turbine, where part of the energy is transformed into mechanical energy through turbine disk rotation. This energy is then used in turn to power the compressor. The propulsive power is achieved through either inlet fan (for turbofan/turboprop engines) or through the core energy expanded through the nacelle (turbojet engines).

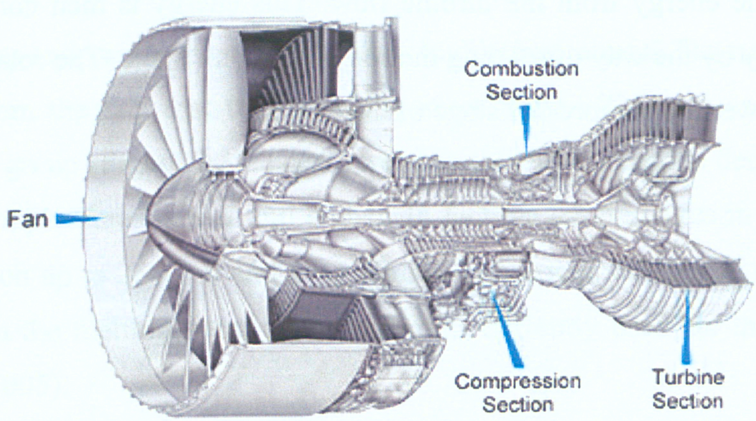

Figure 2.1 A schematic view of the typical turbine engine. Courtesy of Pratt & Whitney

2.2 Turbine Blade & Vane Description and Design Challenges

Turbine blades and vanes facilitate the transformation from chemical to mechanical energy in the turbine engines. The design process for these components can be very challenging because of the multitude of factors that must be taken into account. The operating environment, performance considerations and manufacturing constraints must all be taken into account to produce an optimal design.

2.2.1 Operating Environment

The operating environment of the turbine stage in the typical jet engine provides the majority of the constraint and requirements on the blade and vane components. The environment variables can be broadly separated into aerodynamic and thermodynamic variables, with the latter being more important.

Aerodynamics

The typical set of the turbine stage in gas turbine engines consists of rows of stators (or vanes) and rotors (blades), as seen in Figure 2.2. Stators are fixed in place; their main function is to deflect the airflow for the subsequent rotor stage. The rotor stages in the turbine remove the energy from the turbine flow. This energy is then converted into mechanical energy by the way of rotating motion of the rotor stage. The rotating energy is then used to power the compressor stage of the engine.

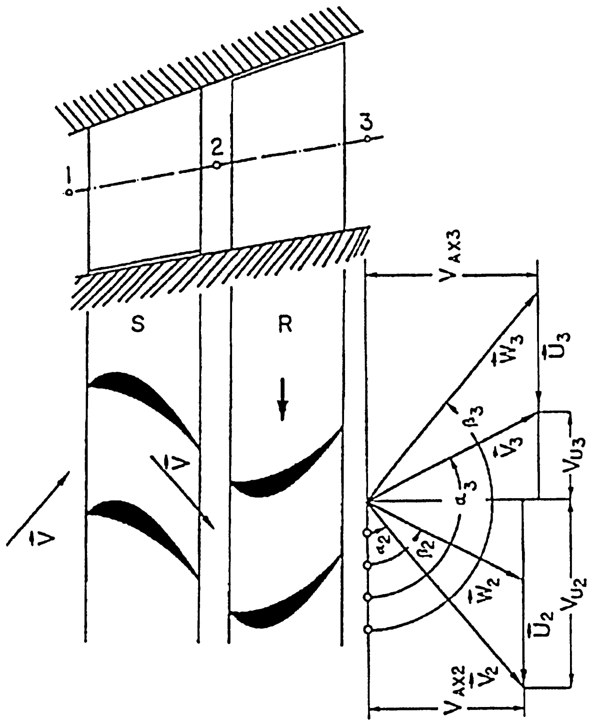

Figure 2.2 Typical turbine stage comprised of stator (vane) and rotor (blade) rows. Adapted from (Han et al., 2000)

Thermodynamics

Turbine blades and vanes operate under very high temperatures. The need for the hotter air comes from the fact that the engine operates more efficiently as the combustion temperatures go up (Figure 2.3). Thus a trend in the turbine engine designs has been to push those temperatures higher and higher. In fact the temperatures in modern engines can often reach up to 2500° F and even higher. These temperatures can be nearly 700° F higher than the melting point of the materials currently used for the turbine blades (Sreekanth, 2005).

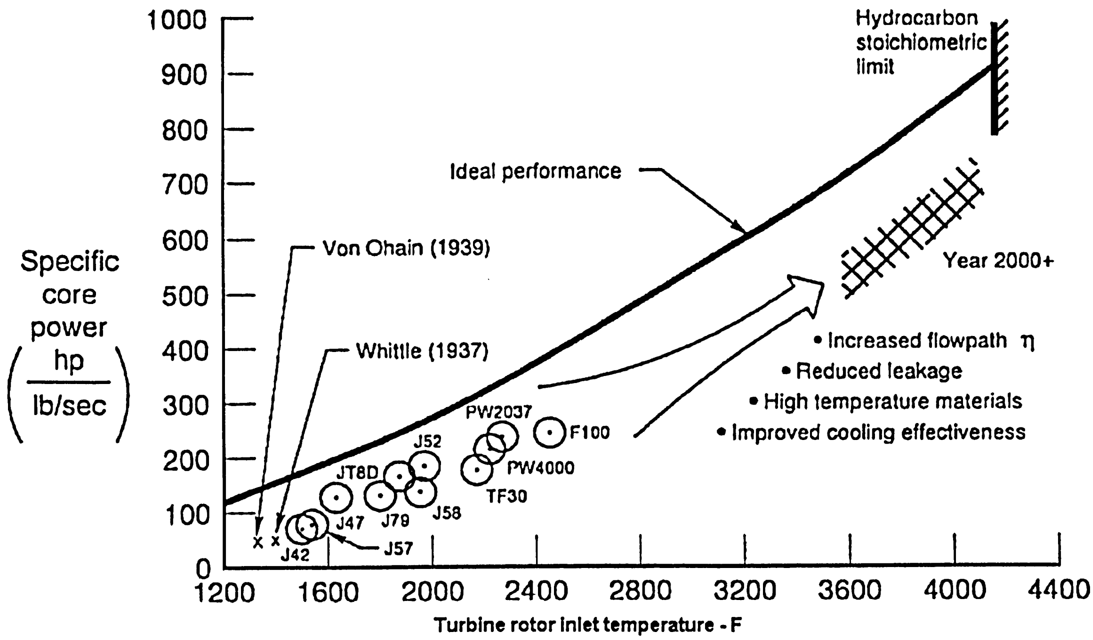

Figure 2.3 Increase in turbine inlet temperature vs. Specific horse power. Adapted from (Sautner et al., 1992). Courtesy of Pratt & Whitney

Thus, in order to be able to operate at such high temperatures, an internal cooling scheme was incorporated into the turbine blade and vane design. The cooling air is siphoned from the compressor stage, incidentally decreasing the engine efficiency at that stage. The evolution of the cooling schemes design has progressed considerably over last several decades (Figure 2.4), allowing for higher turbine inlet temperatures and consequently higher engine efficiency. Figure 2.5 shows an example of the cooled turbine blade section. As can be seen from the figure, the inside of the blade contains several hollowed passages, which allow the access of cold air from specially built channels. Some of this cold air is allowed to leak onto the surface of the blade through cooling holes (the rows on the top of the blade) and mix with the hot outer flow to create a cooled film of air around the blade.

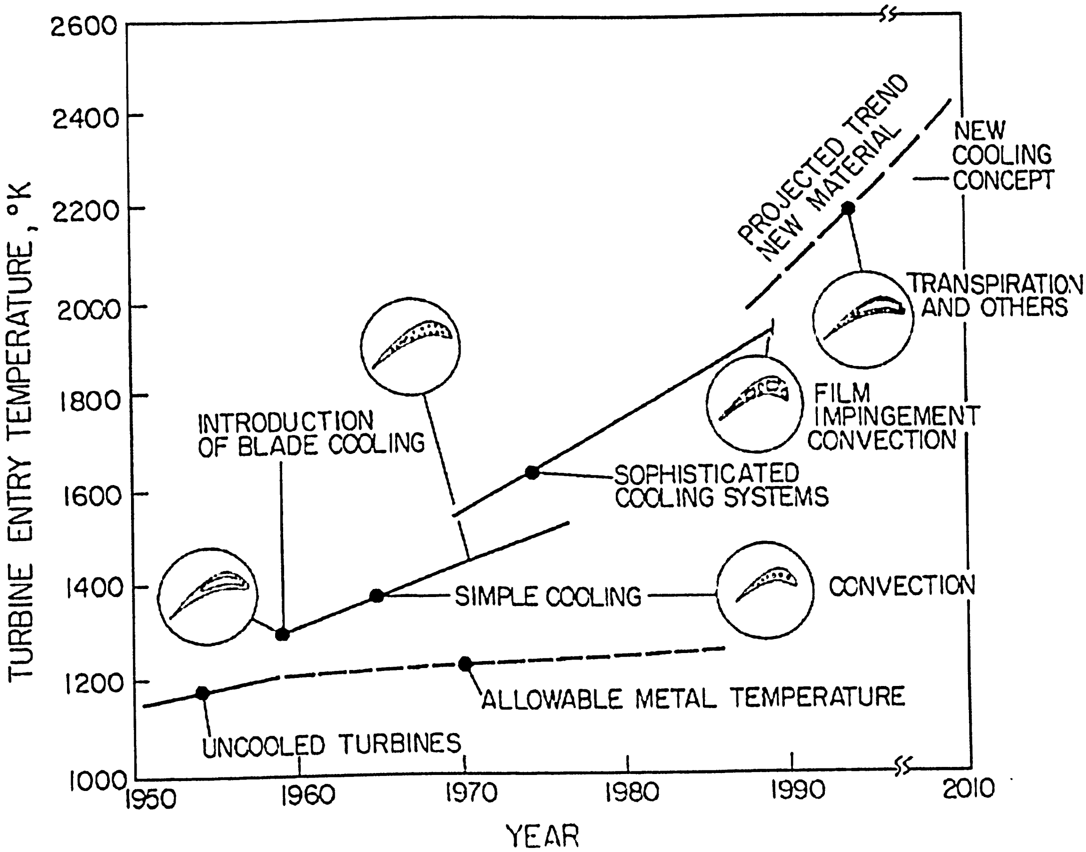

Figure 2.4 Increase in turbine engine temperature over the years. Adapted from (Lakshminarayana, 1996)

The internal cooling geometry can be very complex, with the cooling airflow fed in through multiple channels into the blade, where it can pass through a serpentine like passages along the length of the blade. Figure 2.6 shows an example of one such design. Here the air is fed in through three passages that are then passed through the internal blade passages. Some air is then ejected through the tip, trailing edge, and a row of cooling holes on the pressure side of the blade

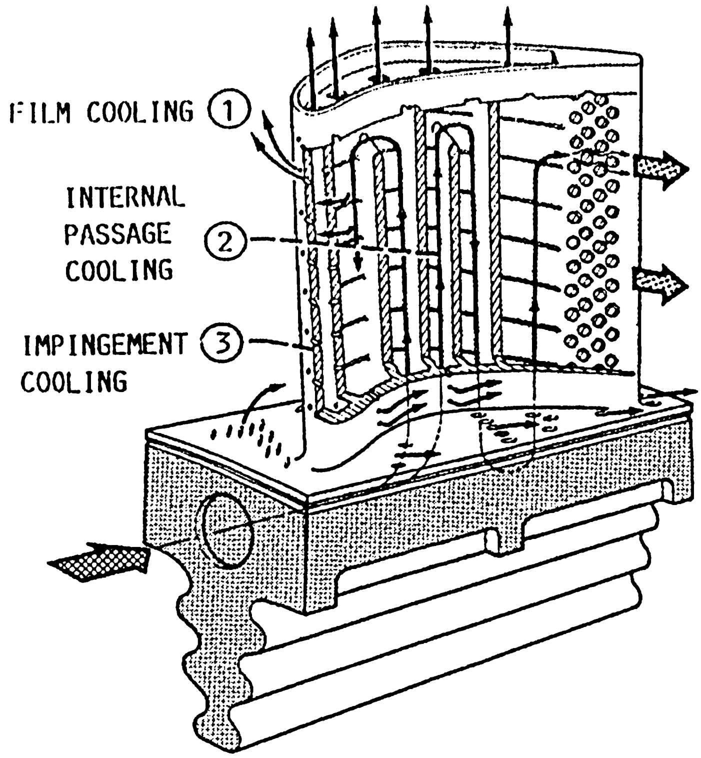

Figure 2.5 An example of a internally cooled turbine blade. Adapted from (Han et al., 2000)

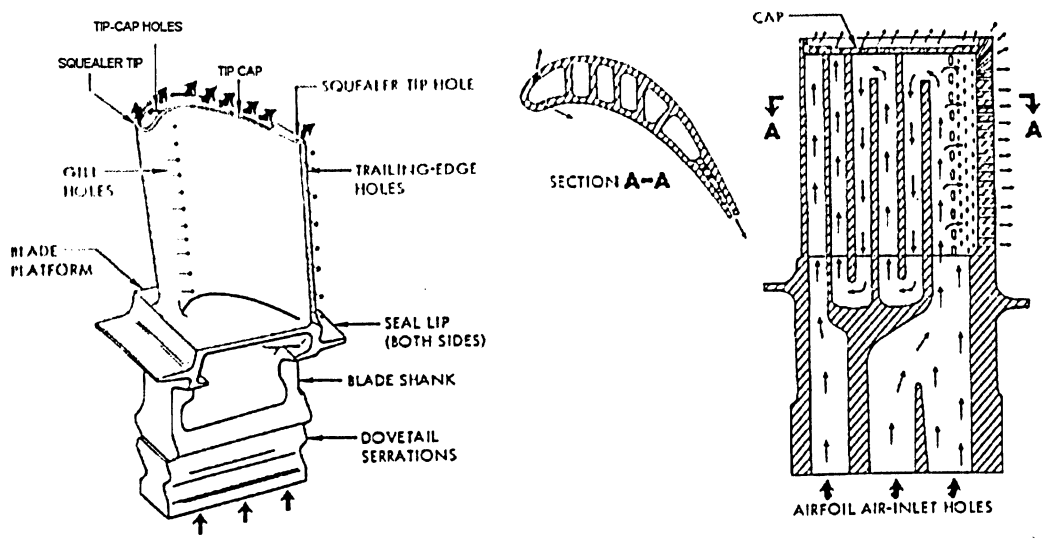

Figure 2.6 High pressure rotor blade geometry and internal cooling scheme. Adapted from (Treager, 1979)

In addition to the passages, the internal cooling scheme usually contains other features that aid in the heat transfer between the cooling flow to metal. Figure 2.7 shows a blade that contains turbulence promoters, pin fins and impingement holes. Turbulent promoters or trip strips induce turbulence in the internal cooling passages thereby increasing the heat transfer rate between the cooling flow and the internal metal walls. Pin fins increase the heat transfer rate as well thereby aiding in bringing down the overall blade temperature. As the name suggests, the impingement holes force the cooling air to impinge onto the surface of the metal, rather than flow over it (as is the case with normal serpentine passage flow), thus increasing the heat transfer rate between the cooling flow and the blade.

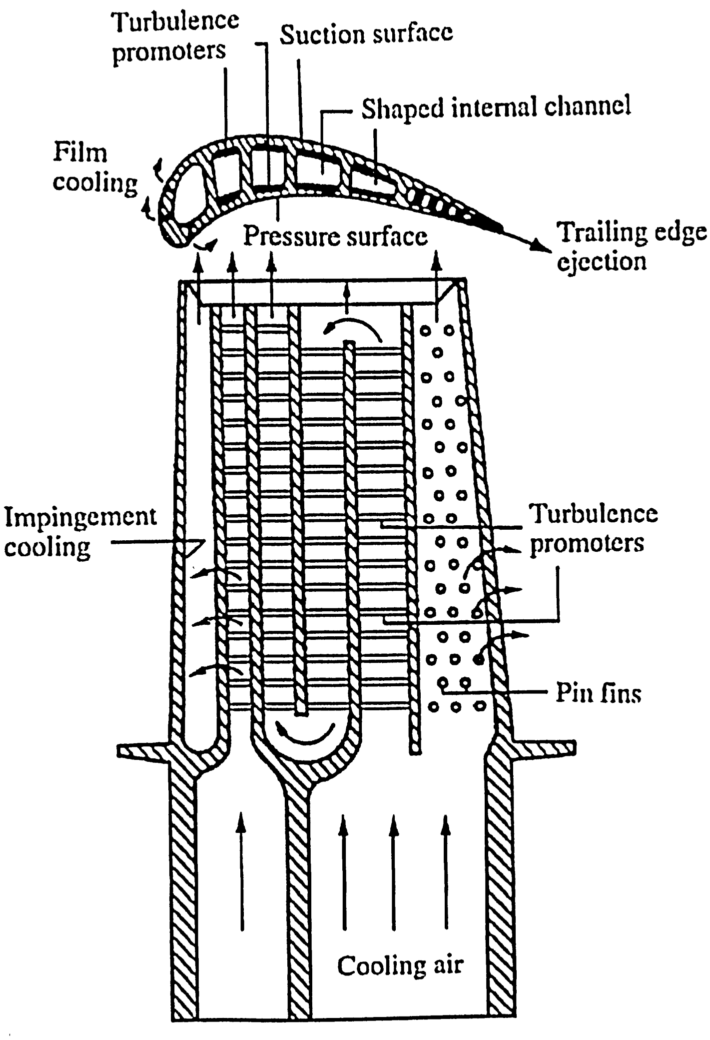

Figure 2.7 Cooling concept of a multi-pass turbine blade. Adapted from (Han et al., 1984)

2.2.2 Physical Characteristics

In addition to the operating environment, several other characteristics play an important role in the blade and vane design. The structural integrity of the components has a strong influence on the blade geometry (largely through wall thickness). The constraints that arise from the nature of the manufacturing process affect both blades and vanes, and actually influence the design methodology.

Structural Integrity

Because of the addition of the cooling passages in the turbine blade core, special attention must be paid to structural integrity of the part. For cooling purposes it may be beneficial to make the metal thickness between the external airfoil and the internal cooling passages (wall thickness) as small as possible, thereby increasing the thermal efficiency of the cooling scheme. This must be balanced against the stress loads acting on the turbine components during the operation.

The main source of stress for the blades comes from the centrifugal forces. The blades can rotate at very high RPM, thereby appropriate metal wall thickness distribution must be incorporated into the design to accommodate these loads. In vanes (or stators) the main source of stress comes from thermal expansion. Since it is fixed at the root and the tip to the passage walls, it does not have room for natural thermal expansion (although the tip and root walls are designed to expand as well to try to match the vane expansion rate it is not usually possible to match it exactly). Because of the thermal nature of the expansion in the vanes, more attention is paid to the cooling geometry to minimize the expansion effect.

Manufacturing Considerations

Because of the thermal, aerodynamic and structural considerations described above, the final shape of the turbine blade or vane can be extremely complex. The manufacturing of such a complex design is currently done through investment casting. Investment casting is the process that allows for manufacture of high-quality parts with complex geometry.

The process begins with a manufacture of metal injection die. This part of the process is very expensive and can take considerable amount of time. With the turbine blade geometry, the die is usually made up of several sections. The exact number of sections depends on the shape of the blade. The completed die is injected with wax. Once the wax has solidified, the die is carefully removed, producing an exact replica of the blade or vane. The component is then dipped repeatedly into ceramic slurry and covered in sand stucco. The dipping process is repeated until the coating is about six to eight millimetres (Figure 2.8). The component is then left to dry. The dried assembly is placed in a steam autoclave, removing most of the wax, while any residual wax in the assembly is burned away. The resulting component is then an empty ceramic shell for the actual part (Figure 2.9). Next, the mould is preheated to sufficient temperature and filled with metal (various alloys are used for blade and vane fabrication). The cooling is carefully monitored to create the desired alloy. Once cooled, the moulding is chipped from the assembly and final sandblasting and machining is done to polish off the part (Figure 2.10).

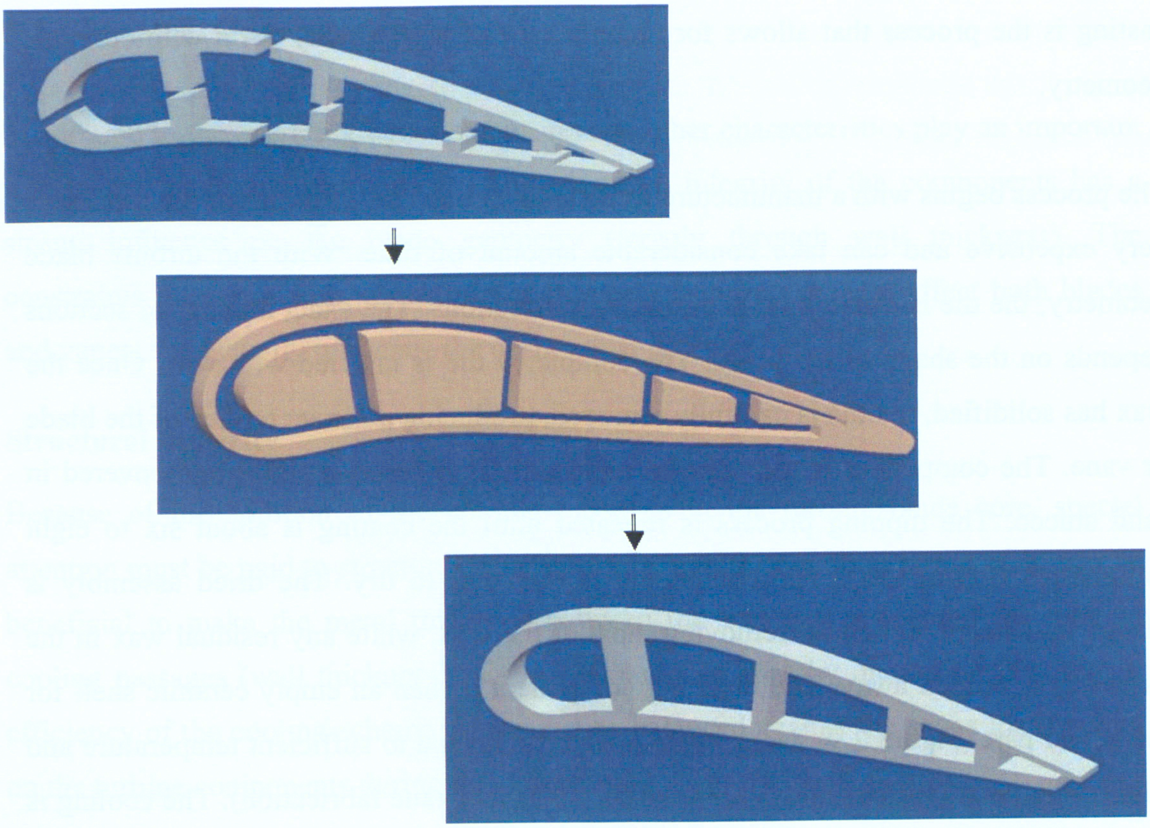

Figure 2.8 Investment casting process

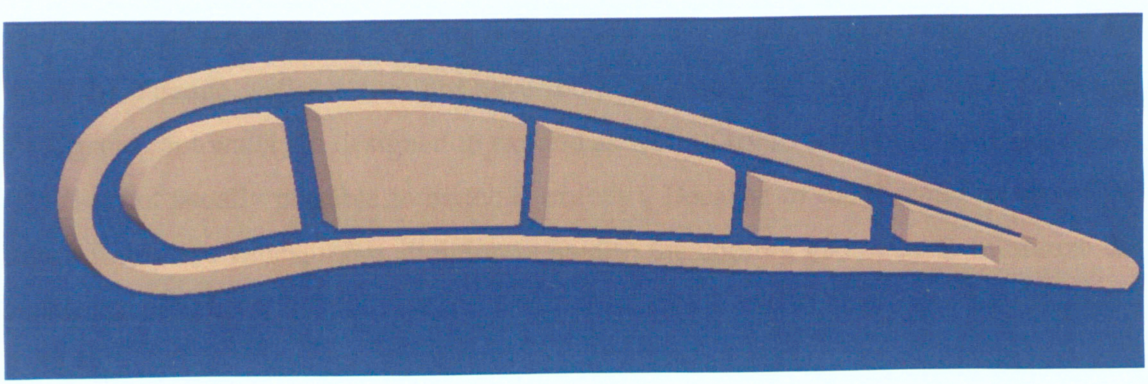

Figure 2.9 Investment Casting Process. Ceramic Shell.

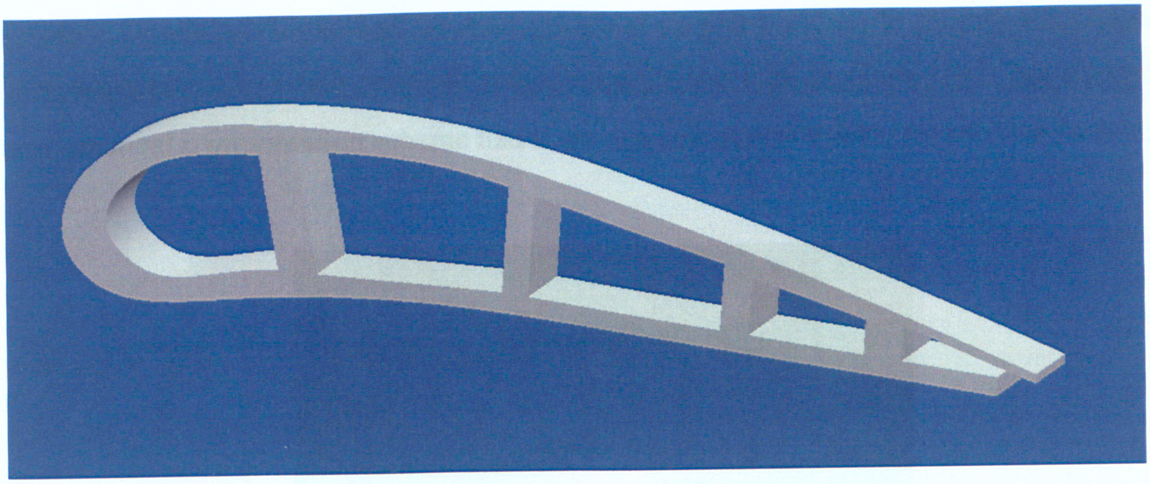

Figure 2.10 Investment casting process. Metal part.

2.2.3 Design Challenges

In light of all considerations described so far in this chapter, it is apparent that the design of the cooled turbine blade or vane is an extremely challenging process. Thermal, aerodynamic and structural considerations must be taken into account. In addition to these, the investment casting process imposes some manufacturing constraints that must be incorporated in the design.

The designer starts with a solid blade or vane. Some analytical work is then done to determine the optimum cooling flow rate and likely outline of cooling geometry is determined. The wall thickness distribution along the part is specified. Once the core thickness is determined additional features are defined. They include the aforementioned cooling passages, flow turbulators, pin fins, trailing edge ejection lands, and other more advanced components (Papple, 2005). The entire design process often requires weeks to complete while the designer makes necessary changes. Figure 2.11 shows an example of turbine blade, complete with cooling passages, and film cooling holes.

Significant amount of time is spent on manually creating and modifying the geometry through the CAD package interface, thus reducing the time that could otherwise be spent on analysis and optimization.

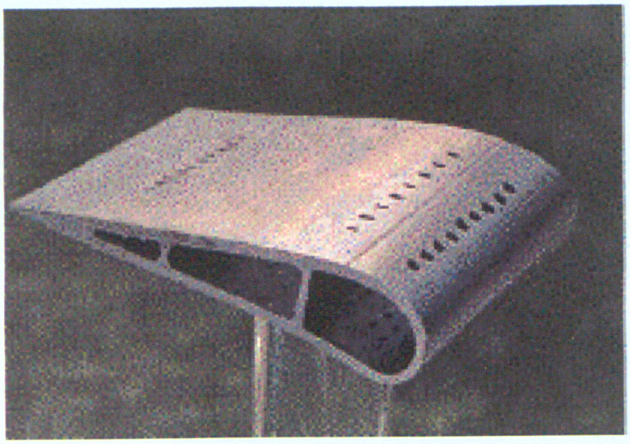

Figure 2.11 An example of cooled turbine blade. Courtesy of Pratt & Whitney

2.3 Parametric Design

A parametric design package for the turbine blade and vane creation would greatly reduce the design lead time and in particular the time spent tediously creating each detail through CAD interface. Over the years, several parametric design packages have been developed for the turbine blade and vane external geometry. They allow the user to specify several parameters that would define the airfoil shape and then use the internal algorithm to determine the final shape. This final shape can then be passed into the CAD software for analysis. Some of these design systems are covered in more detail in Chapter 4. However, to the best of the author's knowledge, there has only been one published in-depth study on the parametric design of internal cooling passages. The summary of this study and the some samples of the resulting geometry are provided in Chapter 4 as well. This particular design scheme has been studied carefully to identify its benefits and shortcoming.

The final product of this work is the parametric design system that can be used to create some of the more predominant elements in the turbine blade and vane internal cooling geometry. However, before the actual parametric work could begin (i.e., identifying the underlying parameters of the different geometries found in the design), the underlying mathematical representation scheme had to be determined.

CHAPTER 3 Research Into Mathematical Representation Methods

This chapter covers the research into various mathematical representation methods. Curve, surface and solid modelling options are discussed focusing on the practical properties of the various types of representation. As part of curve/surface representation, particular focus is placed on spline representation including β-splines, B-splines, NURBS as well as Bezier curves. Various solid modelling techniques are discussed as well.

3.1 Mathematical Representation Types

There are three predominant forms of mathematical representation: implicit, explicit and parametric. An implicit equation for a plane curve can be expressed as:

$$f(x,y)=0 \qquad (3.1)$$

For example an implicit curve of the second degree can be expressed as $ax^2+bxy+cy^2=0$. A parametric representation involves representing variables (e.g. x and y) as a function of parameter t, within closed interval $t_1 \le t \le t_2$. The general form therefore is:

$$x=x(t), y=y(t) \qquad (3.2)$$

The explicit form can be considered as a special case of parametric and implicit forms. If t can be expressed as a function of x and y, t can then be eliminated to yield an explicit form:

$$y = F(x), x = G(y) \tag{3.3}$$

Each of these representation forms has its advantages and disadvantages, but as will be shown, the parametric representation is best suited for robust geometry representation and generation, as well as for optimization algorithms. The following sections describe in detail the properties of each type of mathematical representation. They focus on curve representation only, but similar conclusions may be inferred for surface representations as well.

3.1.1 Implicit Representation Overview

The implicit representation is one of the most widely used forms in geometric modelling. This method describes an implicit relationship between x and y coordinates of the points lying on the curve. It has several important advantages. One of the most important advantages of this representation form is that closed curves and infinite slopes can be represented. An example of that is a circle of the form:

$$x^2 + y^2 - r^2 = 0 \tag{3.4}$$

It is also easy to check if a given point is lying on the curve. This factor is very important when evaluation of intersections is needed.

The implicit representation also has several crucial drawbacks. Firstly and most importantly, it is difficult to fit or manipulate free form shapes, a feature which is very important in most design and optimization applications. It is also axis dependent, thereby making the geometric transformations rather difficult. For example, taking a hyperbolic equation of the form (Patrikalakis, Meakawa, 2002):

$$\frac{x^2}{4} - \frac{y^2}{2} = 1 \quad (3.5)$$

Translating it by (1,2) and rotating the coordinate axis about the origin by $\tan^{-1}(3/4)$ will transform the equation (3.5) into:

$$2x^2 - 72xy + 23y^2 + 140x - 20y + 50 = 0 \quad (3.6)$$

This is clearly cumbersome for a system where frequent geometric manipulations may be needed.

3.1.2 Explicit Representation Overview

The explicit method is another popular method of analytical representation. It describes one of the coordinates in terms of another coordinate (or coordinates if more than two are involved). This method is even less useful from the point of view of robust manipulation. There is the same problem as with implicit method when it comes to manipulation as this representation method is axis dependent as well. Moreover, the scope of geometries that can be readily represented is even more limited, since closed geometries may not be easily represented. As can be seen from the figure 3.1, the circle must be represented by two equations instead of one.

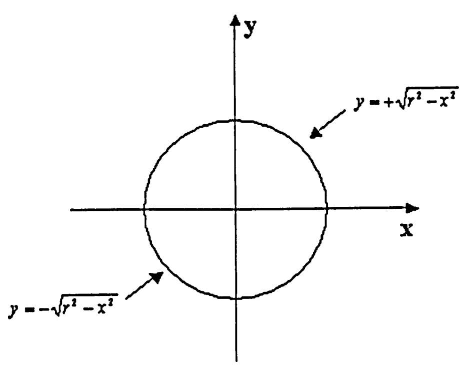

Figure 3.1 Circle representation using implicit method. Adapted from (Patrikalakis, 2003)

The only important advantage of this form of representation is that it is easy to trace the curve and determine if a given point lies on it.

3.1.3 Parametric Representation Overview

As described, the parametric representation is defined by expressing each coordinate as a function of a parameter:

$$r(t) = (x(t), y(t)) \tag{3.7}$$

Here *r(t)* is vector valued function of the independent variable *t*. This representation has several important characteristics. One of the first deductions that can be made is that, unlike for implicit or explicit representation, parametric representation is not necessarily unique. In other words, there may be more than one different expression for the exact same curve. For example, a first quadrant of the circle in figure 3.1 can be parametrically described as:

$$\begin{aligned} x(u) &= \cos(u) \\ y(u) &= \sin(u) \end{aligned} \qquad 0 \le u \le \pi/2 \tag{3.8}$$

Then setting $t = \tan(u/2)$, an alternative representation can be obtained:

$$\begin{aligned} x(t) &= \frac{1-t^2}{1+t^2} \\ y(t) &= \frac{2t}{1+t^2} \end{aligned} \qquad 0 \le t \le 1 \tag{3.9}$$

Both equations (3.8) and (3.9) describe exactly the same geometry, yet they look quite different. The drawback of such flexibility is that it complicates the intersection and point classification.

There are however many advantageous characteristics in using parametric representation. It is easy to generate closed curves and infinite loops, as well as fit and manipulate free form shapes (which is not easy at all with other forms of representation). This representation is also axis-independent, which allows for easy transformation of geometry. All these points make the parametric representation the best possible method of expressing geometry. As a result parametric representation has been used extensively in CAD/CAM applications.

3.2 Parametric Representation

As mentioned before parametric representation of geometric shape is very useful for CAD modelling. However, even with such representation alone it may be challenging to describe any continuous curve or surface shape with one polynomial (or other) function. The splines allow the designer to describe almost any geometry through series of curves or surfaces that are interconnected. A spline curve is composed of a sequence of polynomials called spline curve segments (Figure 3.2) while spline surface is a mosaic of surface patches (Figure 3.3).

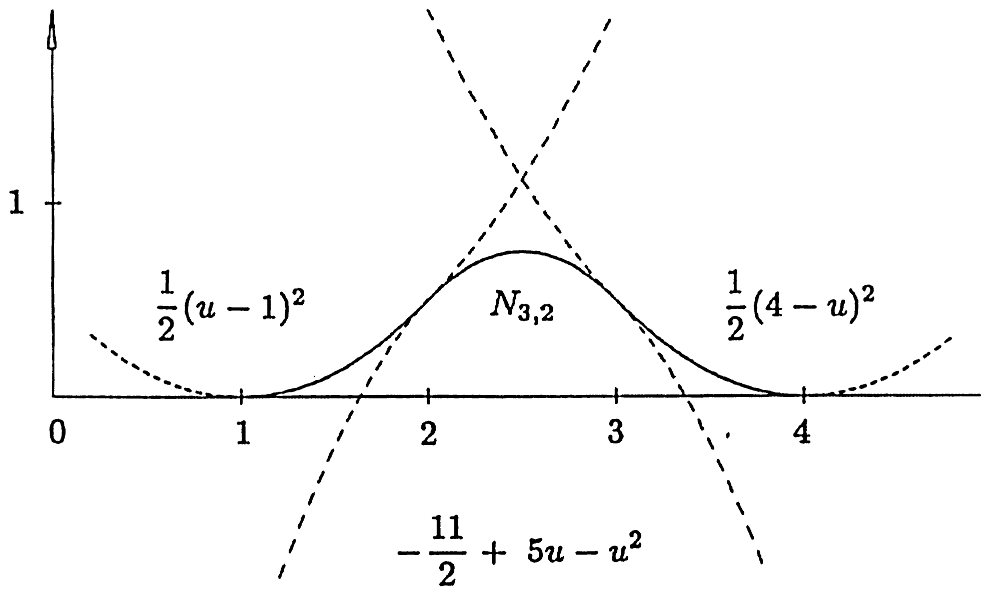

Figure 3.2 Polynomial decomposition of quadratic spline. Adapted from (Piegl et al., 1997)

There are many applications of spline curves and surfaces, primarily in the naval and aerospace industry. Some of these applications will be examined in more detail in later chapters because it has direct relation to the application considered in this study, namely the turbine blade design. The splines themselves can be broken down into several categories, that have different properties and applications. The following sections examine different spline types for both curve and surface representation. Lastly the solid definitions are examined, from the perspective of creating them from base curves and surfaces, as may be often the case with CAD design. The information given in this chapter does not go into in-depth mathematical analysis, but rather focuses on practical features of various representations.

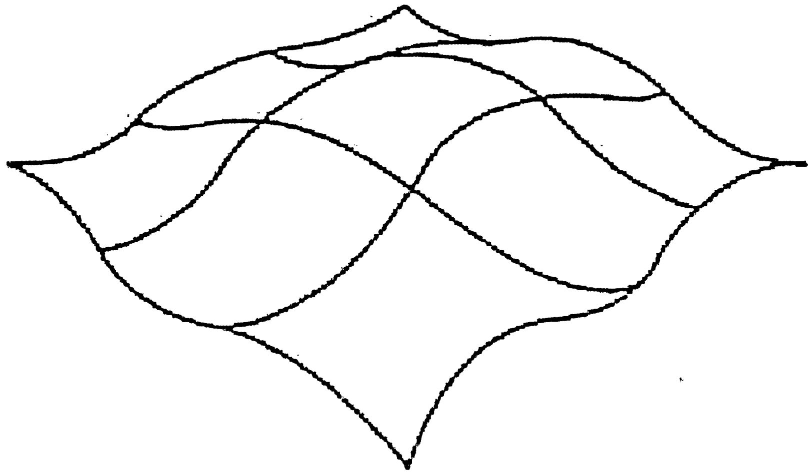

Figure 3.3 Surface spline as a combination patches. Adapted from (Barsky, 1988)

3.2.1 Parametric Curves

Bezier Curves

One of the earliest types of parametric polynomial curves to be developed were Bezier curves. The general form of a Bezier curve is:

$$C(u) = \sum_{i=0}^{n} B_{i,n}(u) P_i \qquad 0 \le u \le 1 \tag{3.10}$$

Where $B_{i,n}(u)$ denotes a basis blending function based on the classical n^{th} degree Bernstein polynomial of the form:

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i} \tag{3.11}$$

The P_i variable denotes geometric coefficients of the control points. Figure 3.4 shows a third order (cubic) Bezier curve with control points.

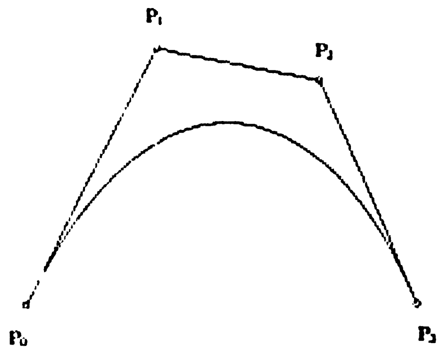

Figure 3.4 Cubic Bezier curve with control polygon. Adapted from (Piegl et al., 1997)

Although Bezier curves can be very flexible, they do have several serious shortcomings when it comes to free form design. In order to satisfy a large number of constraints, a very high degree is required (e.g. n-1 degree is required to pass a polynomial Bezier curve through n data points). However high degree curves can be numerically unstable and are inefficient to process. In addition there is very little local control over the geometry. Although the Bezier curve can be controlled via the control points, the resulting change is not local. A change in the position of one of the control points will affect the geometry of the entire curve. The reason for this can be seen in the general form of the Bezier curve in equations 3.10 and 3.11. The basis blending function is structured in such a way as to incorporate all of the control point's positions into each segment of the curve, no matter how far removed the given segment is from a particular control point. This can pose a problem where only local change is required to be made to the geometry.

B-Splines

The parametric polynomial function, even one as flexible as a Bezier curve, may not be enough to accurately and efficiently model complicated geometries in the CAD environment. The solution to this is to use piecewise polynomials or splines. The use of B-splines curves for CAD based geometric design was first proposed in the seventies. This group of curves has many properties, which make them very useful in free form geometry creation. A p^{th} degree B-spline is given by:

$$C(u)=\sum_{i=0}^{n} N_{i,p}(u)P_i \qquad a \le u \le b \tag{3.12}$$

Where P_i are the control points and the $N_{i,p}(u)$ is the B-spline basis function described as:

$$N_{i,0}(u)=\begin{cases}1 & \text{if} \quad u_i \le u < u_{i+1} \\ 0 & \text{otherwise}\end{cases} \tag{3.13a}$$

$$N_{i,p}(u)=\frac{u-u_i}{u_{i+p}-u_i}N_{i,p-1}(u)+\frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1}(u) \tag{3.13b}$$

This function is defined on a non-periodic and non-uniform knot vector:

$$U=\left\{\underbrace{a,\ldots a}_{p+1},u_{p+1},\ldots,u_{m-p-1},\underbrace{b,\ldots b}_{p+1}\right\} \tag{3.14}$$

The polygon formed by P_i is termed a control polygon and roughly approximates the shape of the curve. Several observations should be made about the knot vector, U. As can be seen from the definition it has constant values of *a* and *b* at the beginning and the end, respectively, for p + 1 values (where p is the degree of piecewise curve). This guarantees an important property know as *clamping*. When this property is in effect, the curve starts at the first control point and ends at the last control point. Moreover the tangents at the start and end of the curve are parallel to the lines drawn from the first control point to the second and from the next-to-last control point to the last one. Figure 3.5 shows a B-spline with control points.

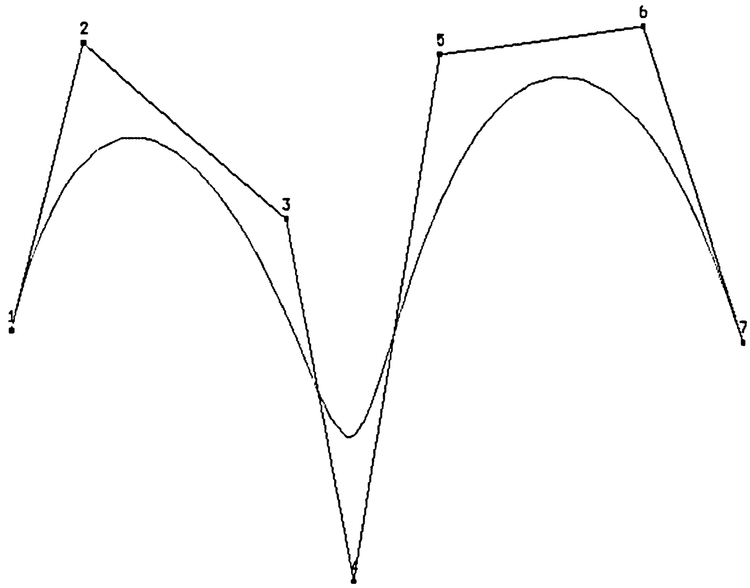

Figure 3.5 B-spline curve with control points

B-spline curves provide local control over geometry via control points. A shift in one control point would alter the shape of four curve segments (two on each side), as seen in Figure 3.6.

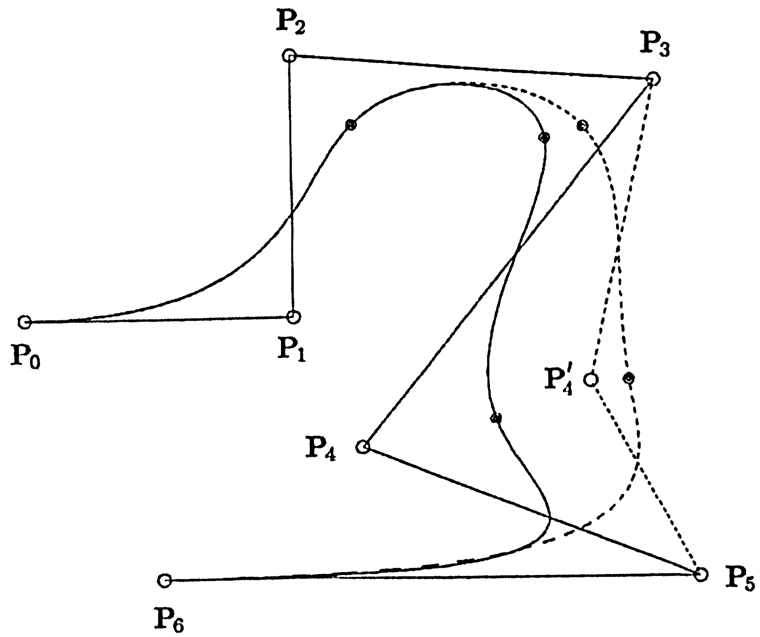

Figure 3.6 The effect of displacement of one control point on B-spline. Note local deformation only. Adapted from (Piegl et al., 1997)

B-spline curves provide a high degree of flexibility and local control due to their piecewise nature. However since they are based on a polynomial form, they can only approximate conical sections (circles, ellipses, cones, etc). The approximations do get better with higher number of control points and degrees, but this can have negative influence on robustness and accuracy of the algorithm.

NURBS

NURBS stands for **N**on **U**niform **R**ational **B**-**S**plines. The idea behind NURBS is to expand the available set of geometric shapes available for modeling, by rewriting the B-Spline form in a rational form:

$$C(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u) w_i P_i}{\sum_{i=0}^{n} N_{i,p}(u) w_i} \tag{3.15}$$

Here the $N_{i,p}(u)$ is the B-spline basis function, defined over a knot vector U, identical to equation 3.14. and P_i is a set of control points. The new variable in this case is w_i which denotes a set of *weights*. Weights can be thought of as a property of control points, and are used to indicate a bias (or pull) towards any given control point. Figure 3.7 shows the effect of changing a weight on P_3.

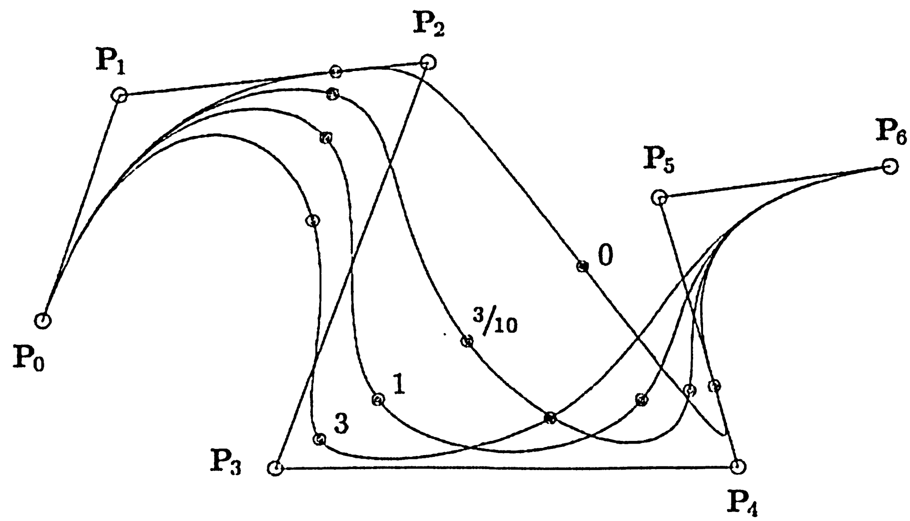

Figure 3.7 The effect of modifying a weight value at P3 on NURBS curve. Adapted from (Piegl et al., 1997)

Because of the rational polynomial form of the NURBS curve, the conical sections may be expresses exactly. This provides a very important additional feature to a design algorithm, since designers often make use of conical section shapes in their models.

β-Splines

β-Splines are another group of parametric piecewise polynomial curves. The curve function for β-splines has a format similar to that of a B-spline (equation 3.12), however it has a basis function, which is described as:

$$B_r(\beta_1,\beta_2;u)=\sum_{g=0}^{3}c_{g,r}(\beta_1,\beta_2)u^g \text{ for } 0\le u<1 \text{ and } r=-2,-1,0,1 \tag{3.16}$$

Here the coefficients $c_{g,r}$ are a function of two parameters, β_1 and β_2. It is beyond the scope of this work to go into a detailed derivation of all parameters for a β-spline curve. Barsky (Barsky,B,A., 1988) provides a complete and in-depth derivation and analysis of this function. It is however important to note several important characteristics of the β-spline. The two parameters β_1 and β_2 are called Bias and Tension respectively. They may be kept constant for any given curve or they may be different at some control points. Their choice strongly affects the shape of the curve.

Bias is defined as a ratio of the first parametric derivatives at the start and end of each individual segment within a spline. Intuitively, β_1 measures the relative influence of this tangent direction on either side of a joint of two segments, thus is called bias at the joint. Large values of β_2 parameter at any given control point tend to shorten the distance between the control point and the curve, therefore having an attractive effect on the curve, hence the term tension. The negative values of this parameter tend to increase the distance between the curve and the control point, as if pushing it away. Figures 3.8-3.13 show β-spline curves defined by the same control points and β_1 parameter, but with a varying β_2 parameter.

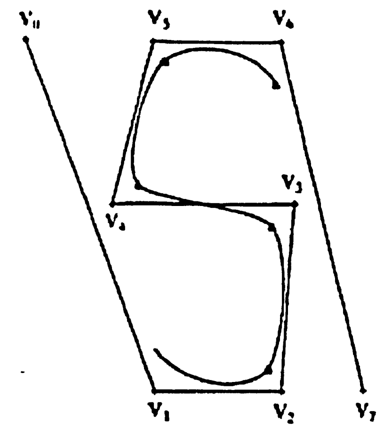

Figure 3.8 b1 = 1 with b2 = 5 (Barsky, 1988)

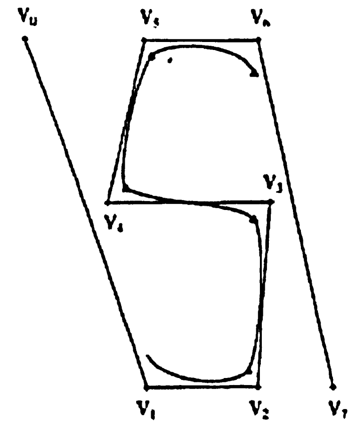

Figure 3.9 b1 = 1 with b2 = 10 (Barsky, 1988)

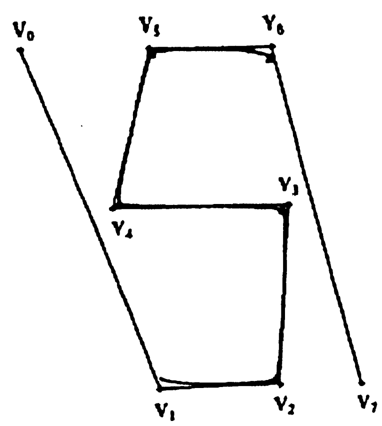

Figure 3.10 b1 = 1 with b2 = 50 (Barsky, 1988)

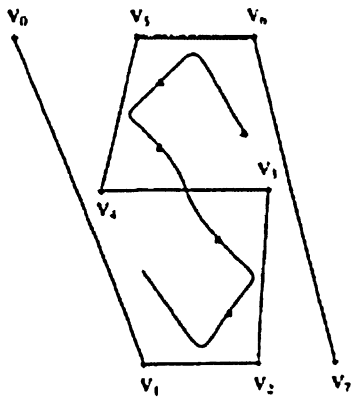

Figure 3.11 b1 = 1 with b2 = -5 (Barsky, 1988)

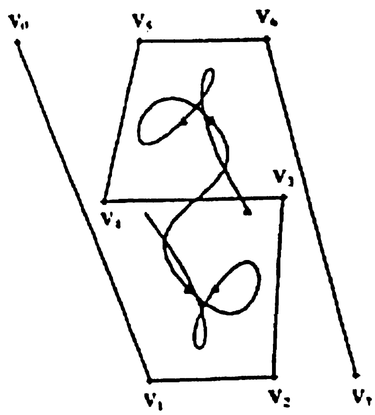

Figure 3.12 b1 = 1 with b2 = -8 (Barsky, 1988)

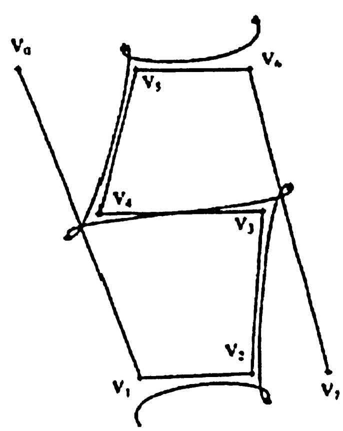

Figure 3.13 b1 = 1 with b2 = -25 (Barsky, 1988)

Several important properties can be inferred from the β-splines, as seen from above examples. They do not have the *clamping* property of B-splines, and as such the curve start and end point cannot be clearly determined from the control points alone. For the same control point lay out, the actual geometry can vary greatly with even small changes to the bias and tension parameters. There is a local effect on the curve if the control point is moved similar to that seen in the B-spline curve.

Overall this type of representation can be seen as the most flexible of the ones covered so far. The geometry that can be described by the β-spline can be reproduced using B-spline or NURBS representation but it would require a greater number of control points or higher degree. The downside of the β-spline function however is its extreme sensitivity to the bias and tension variables, which can sometimes combine to give unpredictable results and may require a close attention from the designer.

3.2.2 Parametric Surfaces

B-Spline Surface

There are several additional considerations when surface representation is concerned (as opposed to curve representation). It is much more useful to have a surface representation of the part than simple wire frame (curve). The surface representation allows for easier manufacturing of the components, hence most CAM software packages are surface based. It also allows to do some of the physical analysis of the component. If the surface model is "watertight", that is continuous without any breaches; a solid model can be usually easily created by exporting it into a CAD application. This in turn allows for a complete thermal, fluid and stress analysis to be conducted on the part.

A B-spline surface is described by a bi-directional net of control points, two knot vectors (one for each direction) and the product of the unvaried B-spline function (Figure 3.14).

The general form is given as:

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u) N_{j,p}(v) P_{i,j} \tag{3.17}$$

Where both knot vector U and V have the same properties as described in equation 3.14.

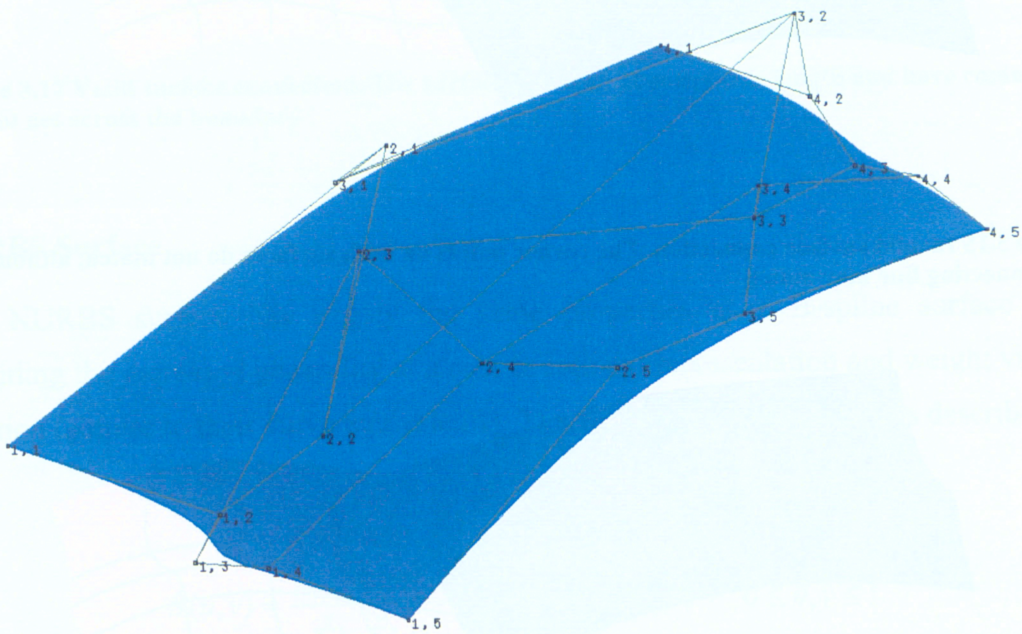

Figure 3.14 B-Spline surface with control point net

Several characteristics of the B-spline surface should be noted. Because of the bi-directional control point net and the clamping property that extends to the surfaces as well through the knot vector definition, any B-spline surface must have four sides and is best used when describing a rectangular patch. Also, if it is desirable to connect two surfaces they must join at the corners of a given side and their control net definition must be identical across the boundary. The degree of the surface in each of the two directions may be different. Figures 3.15 and 3.16 show surface connections which would be invalid between two B-spline surfaces. Figure 3.17 shows an example of a valid B-spline connection.

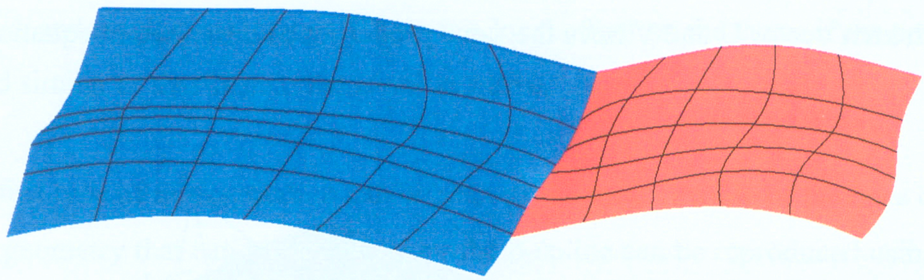

Figure 3.15 Invalid surface connection. The corner points on both surfaces do not match, although the connecting line may match

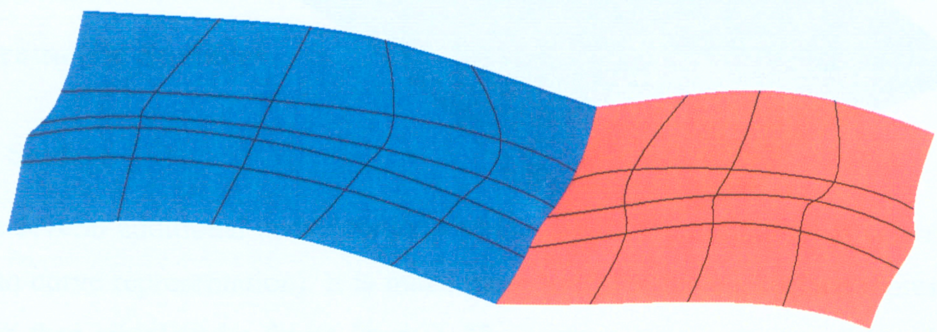

Figure 3.16 Invalid surface connection. The control net is not continuous from one surface to another

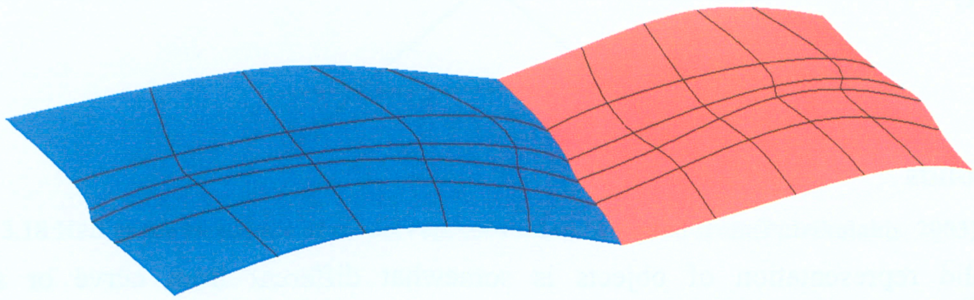

Figure 3.17 Valid surface connection. The surfaces connect at entire side length and have common control net across the boundary

NURBS Surface

The NURBS surface has largely the same properties as a B-spline surface while providing the increased generality at a cost of rational representation and weight variable addition (similar to their curve equivalents). The general NURBS surface is described as:

$$S(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)w_{i,j}P_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)w_{i,j}} \qquad 0 \le u,v \le 1 \qquad (3.18)$$

The limitations and properties associated with the NURBS surfaces are largely the same as B-spline surfaces. The basic underlying shape of the surface patch has four distinct sides. Also same constraints apply when two surfaces are to be joined at a common interface.

β-Spline Surface

β-Spline surfaces are similar to the B-splines in the fact that they too are defined over a roughly four-sided shape. The surface uses the same parameters as the β-spline curve, and as a result a wide variety of geometry may be created. The design using β-spline surfaces has seen application in computer animation algorithms, modelling complex models such as facial features. The same basic principles apply to the surfaces as to the curves, i.e. a very high flexibility of the design, but in need of close supervision by the designer.

3.2.3 Solids

The solid representation of objects is somewhat different from curve or surface representation. For starters there are currently several different widely used representation methods (as opposed to surface/curve representation where NURBS seems to be becoming an industry standard). There are many ways to classify the solid representation types. However, if the final solid geometry is to be created through curve/surface geometries the classification of solid may be separated into two distinct categories: boundary based solids and volume based solids. As the name implies boundary based solids are defined by the boundary elements (e.g. vortices, faces (surfaces), edges (curves)), while the volume based solids are defined by the (de)composition, subdivision or Boolean combination of some basic units of volume.

Boundary Based Solids

There are several distinct boundary based models that are currently used in the CAD industry. Several of the more popular ones will be briefly covered here, including half-space technique, Euler operators and boundary representation.

In the half-space technique the solid is represented by successively dividing the space in half and selecting the half space on the specified side of the solid, eventually inclosing the solid region (Figure 3.18). The intersections of the half spaces represent the solid. This method can only describe convex solids, unless unions are employed.

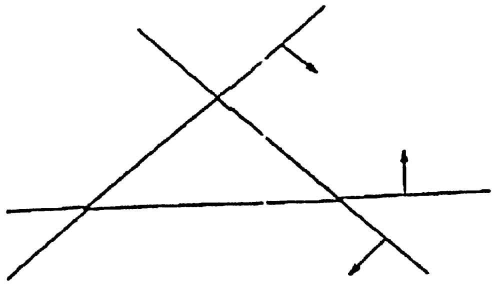

Figure 3.18 Half space technique of model representation. Adapted from(Patrikalakis, 2003)

In the Euler operators form of representation, the object is described as a sequence of Euler operators. An example in Figure 3.19 shows a sphere being topologically modified using Euler operators such as:

- make shell, vortex
- make edge, vortex
- make edge, face

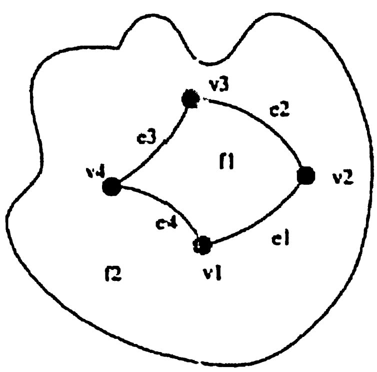

Figure 3.19 Euler operations boundary representation. Adapted from (Mantyla, 1985)

The operators ensure that Euler's Formula is always satisfied:

$$V - E + F - L_i = 2(S - G) \qquad (3.19)$$

where:

- V is the number of vertices
- E is the number of edges
- F is the number of faces
- L_i is the number of internal loops
- S is the number of surfaces
- G is the number of genus (handles or through holes)

The most widespread method of solid representation is boundary representation. Here the objects are represented in terms of their boundary elements: vortices (points), edges (curves) and faces (surfaces) (Figure 3.20). These elements are related through a property known as adjacency. This system is clearly best suited to represent geometries that have been "built up" from base curve and surface shapes.

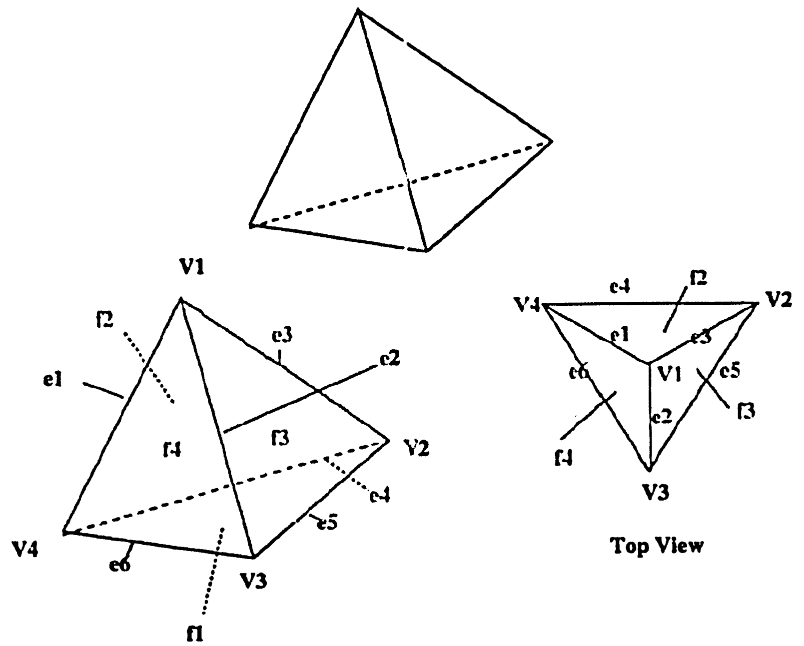

Figure 3.20 Boundary representation (tetrahedron). Adapted from (Hoffmann, 1989)

Volume Based Solids

Decomposition models represent the solid in terms of subdivision of space. This model utilizes continuous space subdivision to represent the solid. Figure 3.21 shows a flat piece represented as a subdivision tree. In this case each square of the space is subdivided into four equal sub-squares and so on. This sort of representation is mostly used for finite element mesh generation and properties approximations (such as volume, center of gravity, moment of inertia, etc.). It is an approximate model and may be memory intensive.

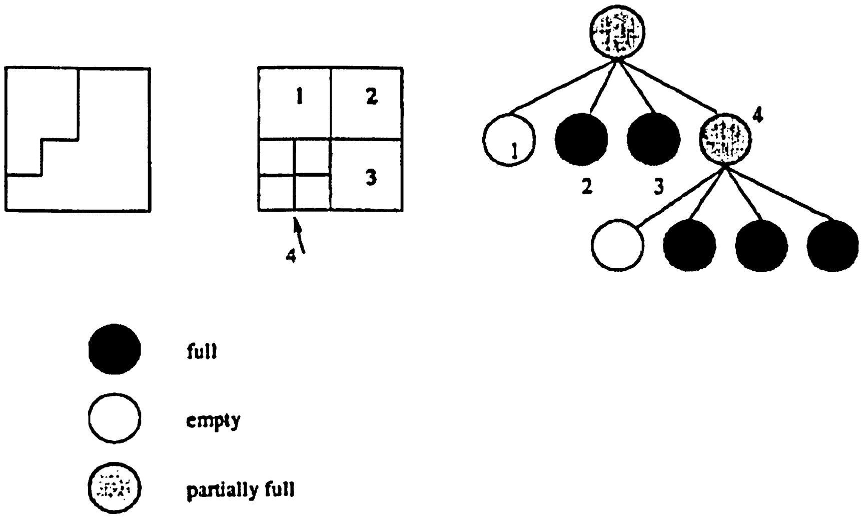

Figure 3.21 Decomposition model of volume based solid representation. Adapted from (Patrikakalkis, 2003)

Constructive modeling is the Boolean combination of primitive volumes that include the surface and interior. Figure 3.22 shows a solid being constructed from three primitives using Boolean operations. This representation is often very efficient with respect to memory. It is usually used for display representation as well as for machining purposes, since the survey of machine elements has found that 90% to 95% of parts can be represented accurately using CSG.

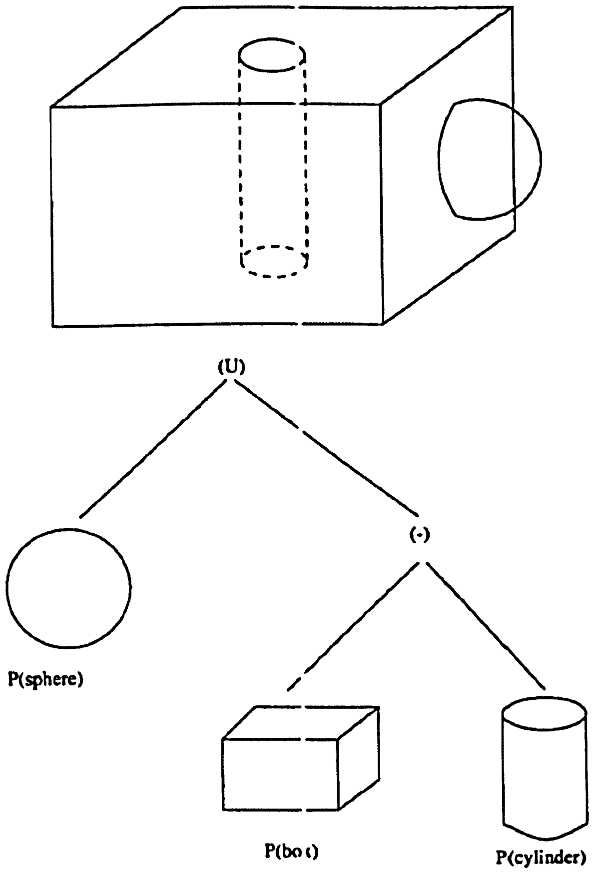

Figure 3.22 Constructive solid geometry representation. Adapted from (Mantyla, 1988)

CHAPTER 4 Parametric Analysis and Application To Turbine Blade and Vane Design

This chapter provides an in-depth look at the core of the research work conducted as a part of this thesis. The analysis of the require attributes of the system, including compatibility, existing standards and component geometry, is presented along with some of the existing work currently done in the field. Methodology for the parametric representation of the turbine blade geometry is developed, and applied to some of the sample cases. The final analysis summarizes the development process and draws the conclusions derived from this work.

4.1 Preliminary Considerations

In order to create a suitable parametric system for turbine blade & vane design, several considerations were taken into account. The compatibility of the mathematical representation with existing CAD packages, established international standards, design flexibility allowed by the methodology structure and the robustness of the scheme in generation of the geometric models.

4.1.1 Compatibility & Standards

The major design feature of any independent parametric system is compatibility with existing CAD packages, in particular the compatibility of the final resulting geometric model. There are several levels of compatibility. The most basic level is achieved through a transfer of cloud of points. In this case the parametric algorithm must break down its own final geometrical model into sets of points, which can be passed into a CAD system (Figure 4.1). Such level of compatibility may be sufficient for rough geometry comparison, and in some cases for FEA analysis (with points as nodes). It is however difficult to obtain an accurate geometrical model from such an output, since there may be a loss of accuracy and a considerable manual effort must be put in through CAD interface to recreate model as curve/surface or solid type.

The next level of compatibility is achieved when curves and surfaces can be directly imported from the parametric system into CAD environment (Figure 4.2). This type of system allows for a much more accurate geometrical representation of the final product in the CAD environment, and can be used for aerodynamic, thermal and structural analysis with minimum "touch-up" work required through the CAD interface itself. It does however have several limitations. The ability to model complex geometry may be somewhat limited, and some work will be required in order to obtain a solid model.

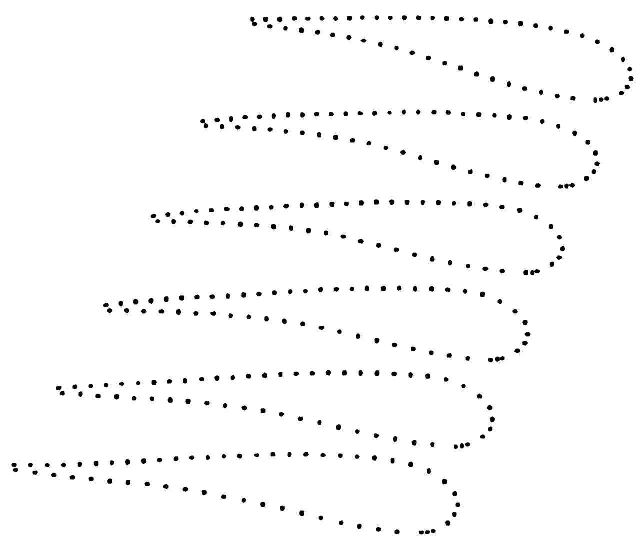

Figure 4.1 An example of point cloud compatibility. This set of points may be passed in, to represent a blade geometry

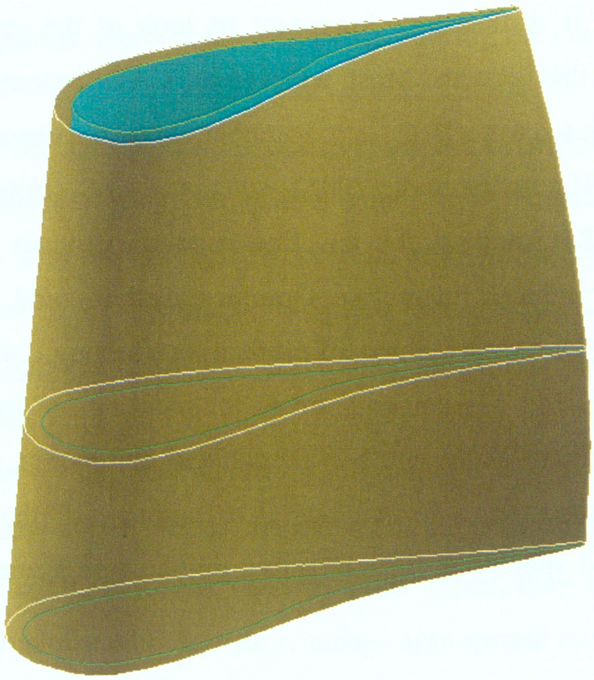

Figure 4.2 An example of curve/surface compatibility. Much more accurate representation than point cloud

A more advanced level of compatibility includes solid representation in addition to curves and surfaces. At this level, model transferred to CAD can be analyzed without any additional effort, and the most accurate geometrical representation is available for manufacturing purposes. It would however, still be difficult to do any kind of additional design work in the CAD itself, since the model would not have memory of how it was created.

A most desirable (and most difficult to obtain) level of compatibility includes not only an equivalent geometric representation but also a compatible hierarchical process of geometry creation (i.e. a construction tree), where each element can be traced back to its most basic components. This requires the parametric system to use identical operations to those used by the CAD system itself. Such a system, although very desirable can be difficult to achieve.

In order to achieve any but the most basic (point cloud) level of compatibility with a given CAD system it is therefore important to look at the geometric representation algorithms used by that system. Most widespread CAD systems available today such CATIA, UniGraphics, AutoCAD, ProEngineer, etc. incorporate the international standards systems (Hammershmidt, 2005), such as IGES (Initial Graphics Exchange Specification), STEP (Standard for the Exchange of Product Model Data), PHIGS (Programmer's Hierarchical Interactive Graphics System) and others. IGES is most widely used format today for exchange of product data among CAD/CAM/CAE systems. Developed in mid 1980's it specifies formats for exchange of graphical and geometric data. STEP is more recent endeavour by international community to standardize not only the geometrical representation format but also topology of the model. PHIGS is an international standard specifying an interactive graphics-programming interface (device independent).

NURBS is rapidly becoming a standard for piecewise parametric representation, and by extension so are B-spline and Bezier curves. Both CATIA and UniGraphics incorporate the ability to model NURBS based curve and surface geometry. Unfortunately, neither of these CAD packages includes β-spline modelling capability.

4.1.2 Design Flexibility

Another important aspect of parametric system is design flexibility. Provided that a necessary level of compatibility with a CAD system is reached, it is still important that the parametric system be able to create various geometric components that are required by the design. When determining the right level of design flexibility it is important to achieve a balance. As a rule the designer does not specify all geometric parameters of the given model. Instead he or she may specify some physical and some of the geometric parameters, depending on the system set up, and it is up to the parametric algorithm to come up with a suitable geometrical model. In light of this, it is important to not only guarantee the design flexibility, but to also make sure that the system would not generate inadmissible designs. Any set of designer input can usually generate more than one set of geometric models that satisfy all of the input. It is therefore important to try and somewhat limit the internal geometry generation flexibility to such an extent as to allow a wide range of design options, while at the same time limiting the inadmissible sets.

This certainly is a very difficult balance to achieve, but it can help to look at the inherent flexibility offered by the various types of mathematical representation covered so far. Bezier curves provide a good variety of representation, but at the same time do not provide local control over the geometry. B-Spline (a more general version of Bezier curves) provides even greater flexibility and local control over the geometry. NURBS incorporate B-Spline flexibility and an ability to model conic sections, although at a cost of increased complexity by the way of rational format and *weight* variable introduction. β-Spline curves provide a very high degree of design flexibility, but require two parameters (β_1, β_2) in addition to one used by other types, they also have the highest chance of producing inadmissible geometry, unless appropriate constraints are imposed on the β_1 and β_2 function parameters.

When looking to strike the necessary balance between flexibility and design integrity these characteristics become extremely important. From the objective analysis of these properties as well as compatibility concerns, it was deemed that a B-Spline or NURBS based system would be best suited as an underlying mathematical model for the independent parametric system. B-Splines offer enough flexibility and local control to create most of the geometric designs, while limiting the inadmissible sets. NURBS provide the same flexibility while allowing to create exact conic sections. Although it is possible to very closely approximate conic sections with B-Splines, this would usually mean a very high degree geometry, which for reasons already covered in chapter three should be avoided. It is therefore desirable to construct a hybrid B-Spline/NURBS system, where B-Spline is the primary mode of representation and NURBS are occasionally used when a conic section is represented.

4.2 Existing Models

There have been few studies into parametric representation of turbine blade & vane geometry. For most part these are concerned with external airfoil component, while only one study actually parametrically models the internal geometry. Although the present work does not concern the external geometry definition of turbine blades or vanes, some of the existing research on the subject is provided in the next section, as it does provide some insights into parametric modeling as a whole and because some of the geometric shapes that are modeled do bear some likeness to the ones covered in the present work, particularly the similarity between external airfoil shape and internal wall thickness geometry. The subsequent section takes a look at an in-depth study on the parametric representation of the internal cooling scheme in turbine blades and vane (Martin, 2001).

4.2.1 External Geometry Definition Models

Two turbine blade/vane external geometry definition models were studied in the course of the research for the present work. Both have a certain degree of similarity although they were conducted by different researchers. Both started out as systems that used Bezier curves for mathematical representation, but eventually matured to using B-splines for greater flexibility.

The first system (Anders et al., 1999), constructs a turbine blade from initial geometric input variables (Figure 4.3). The final product of the system is a complete three dimensional blade or vane. This final result is achieved by lofting several two dimensional design sections, where most of the parameters are specified by the designer.

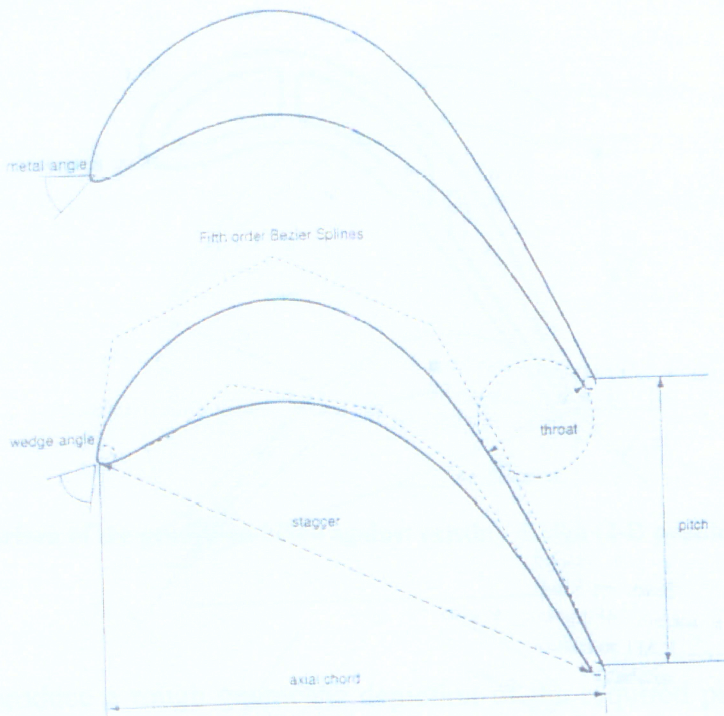

Figure 4.3 2-D sections and some parameteres used in the AutoBlading design system. Adapted from (Anders et al., 1999)

This system uses Bezier and B-Splines for mathematical representation of 2-D Curves, and has been subsequently interfaced with an optimizer algorithm, which authors indicate performed with success. This particular model was not transferred to any CAD software but was rather viewed through its own viewer. The parametric variables are standard for turbine blade creation and include conic section definitions of leading and trailing edge as parabola and circle respectively and a B-spline definition of the pressure and suction side. The authors (Anders et al., 1999) report using B-spline for greater flexibility, over the previously used Bezier curves.

The second system (Corral et al, 2004), creates 2-D sections of the turbine blades and vanes. The parametric definition is similar to that of the previous work. The suction and pressure side curves are defined using a 5-th degree B-spline curves. Authors report using B-spline in order to achieve accuracy as well as to have a good local control over the geometry (Figure 4.4).

Figure 4.4 Some of the parameters used in the external turbine blade definition algorithm. Adapted from (Corral, 2004)

4.2.2 Internal Geometry Definition Models

As mentioned before, to the best of author's knowledge, there is currently only one published work that takes a look at the parametric definition of the internal cooling geometry (Martin, 2001). The described model uses β-splines for mathematical representation and outputs point clouds that represent the part geometry. The algorithm is capable of reproducing the external airfoil and some of the internal cooling geometry features, including wall thickness distribution, cooling passages, pin fins and others (Figure 4.5).

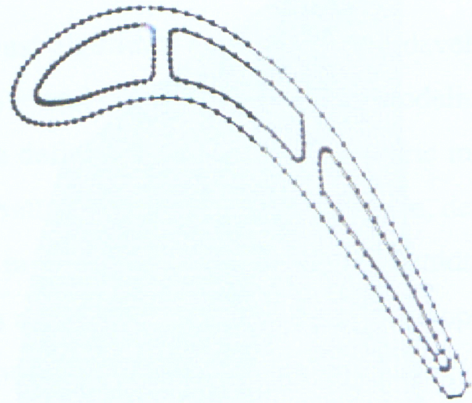

Figure 4.5 Comparison of the generated blade against existing design (2-D profile. Adapted from (Martin, 2001)

The aim is to produce a rough geometric depiction of the required part and to create a preliminary model for Finite Element Analysis (FEA) (since points can be used as nodes). This study encompasses the complexity of the turbine design and includes parametric definition for many of the cooling features (Figure 4.6).

As inclusive and flexible as it is, this system does have several shortcomings. The use of β-splines does not allow a level of compatibility beyond point cloud level. In case the designer wants to have a surface or solid model for higher accuracy or more involved analysis, manual work has to be done through CAD interface to a bring the model to that level.

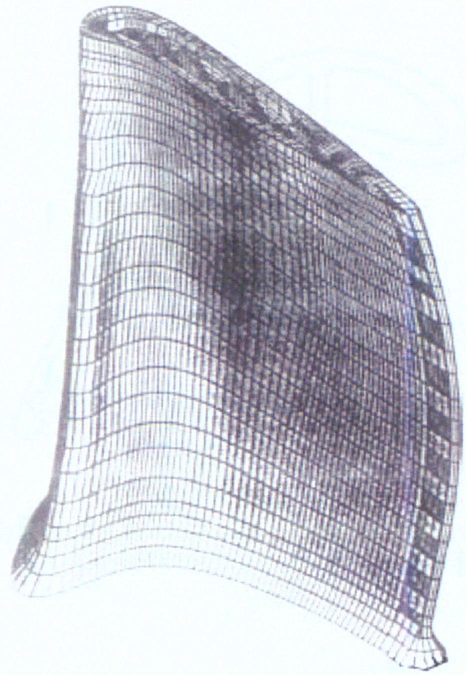

Figure 4.6 An example of boundary element mesh generated for structural analysis. Adapted from (Martin, 2001)

4.3 Cooling Scheme Design Scope

4.3.1 Overview

The need for a robust parametric scheme has already been examined in chapter two. In addition many of the elements that are commonly used in the turbine cooling design have been introduced as well. This primary objective of this work is to show that the parameterization of these components can be done through the proposed mathematical model and as such includes the design, creation and testing of the necessary algorithms to cover some of the most widely used components. The cooling components that have been created using the specially developed algorithm are covered over the next several sections. They include wall thickness creation, split plane definition, and cooling features designs. All these components together constitute the basis for the most rudimentary cooling scheme designs used in the turbine blades and vanes.

4.3.2 Present Scope

In order to prepare a robust algorithm and shorten the development time two geometric libraries were used, NLib™ and GSNLib™ (© Solid Modeling Solutions), which contain the NURBS and B-spline definitions as well as geometric modeling operations. The use was also made of some Pratt & Whitney Canada software, developed with these libraries, through assistance of the in-house developer (Hammershmidt, 2005). In addition to these tools, new functionalities and procedures have been developed during the course of this study and have been incorporated in the design. Using all these tools, a program algorithm has been created that could generate the geometric representation of the cooling scheme components from user defined parametric input. Figure 4.7 shows a schematic overview of the methodology and the research algorithm. The models generated were compatible on the level of curves and surfaces with the CAD software (CATIA V5 in this case). The solid modeling through the algorithm was not possible at the time because of the lack of compatible solid definition library, but future work in this area is certainly feasible.

As mentioned before, the design process of the cooling scheme starts with the wall thickness definition. The designer specifies the "hollow" volume within the blade or vane which will then be available for cooling scheme definition and can include additional features. The wall thickness is an important and crucial first step, as it has to take into account structural and thermal considerations specific to turbine environment. The algorithm developed in the process of this work similarly starts with the wall thickness definition module, which asks the user to specify all of the parameters that were deemed necessary after an in-depth analysis of the design procedures and case studies. The output of this module includes a 3-D wall thickness definition, which is then available to all subsequent modules of the algorithm for further design. It allows the user to get a very early visual overview of the design and if necessary allows for some quick structural analysis (with little preliminary work done through CAD interface to obtain a solid model) to make sure the design is structurally sound.

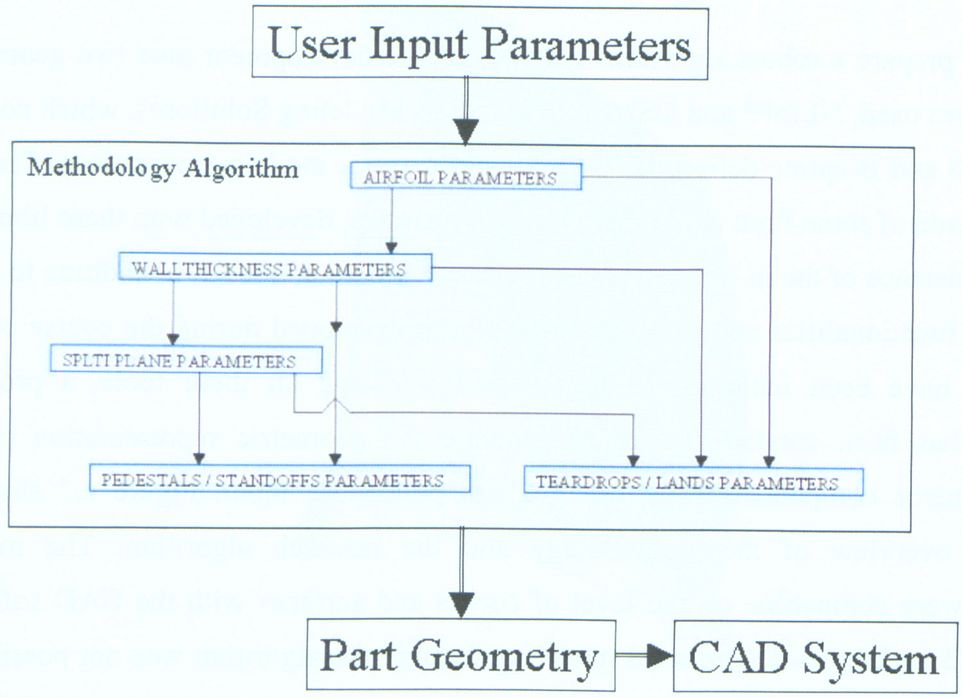

Figure 4.7 Present Methodology and Algorithm Overview

Once the wall thickness definition is complete, the design moves into the internal cooling definition scheme. Before any design can be done though, pull planes must be created on the blade. Pull planes are not actual geometric components; rather they are imaginary planes, which are used to define all of the internal cooling geometry. In the manufacturing process section described in chapter two, the process of investment casting used for the turbine blades and vanes is described. Because of the complex geometry of the turbine blades and vanes, it is sometimes necessary to produce the core die in several sections. Each of these sections has a plane associated with it. In order to create a feasible design, the designer must identify the number of these sections and their respective plane early on and then design each of the components in the sections on the plane. Once the cooling components have been defined on the section planes (known also as split planes) they may be projected into the blade or vane for their actual 3-D representation. The split plane module in the research algorithm presented here lets the designer specify the split planes and their respective orientation, as well as includes a sophisticated procedure to determine the blade or vane limits on the plane, to facilitate a robust and intuitive design.

Following the definition of the split plane, the designer then can begin the task of defining the precise cooling geometry complete with all necessary components. These may include the vane insert, which facilitates impingement cooling in vanes, serpentine design which is used in the blades, and other features including turbulence inducers, pedestals, stand-offs, teardrops (also known as lands) and others. The research algorithm contains the complete definitions of two parameters most commonly used in the design: the pedestals and the teardrops. The pedestals are circular cylinders, which extend from the pressure side of the wall thickness to the suction side, connecting the two. They play a role in cooling of the blade acting as fins and also provide structural integrity to the blade and vanes preventing the suction and pressure sides in the vicinity from ever expanding or collapsing into each other. The teardrops (or lands) are used in the trailing edge ejection geometry. In the designs where the cooling air is ejected from the trailing edge, the teardrops are used for the role similar to that of the pedestals, i.e., facilitating heat transfer and maintaining the structural integrity of the blade at the tip.

The continuation of this work will examine other cooling components listed above, as well as any additional ones. As mentioned before, the components that have been modeled already constitute a cooling scheme and were used to re-create some of the existing designs for validation. The next sections provide an in-depth look at the design methodology developed in this work for the parametric algorithm, as well as some of the results generated by the algorithm compared to the existing turbine blade and vane cooling geometries.

4.4 Wall Thickness Design

The wall thickness design is an important first step in the creation of the internal cooling geometry. Traditionally, the wall thickness is defined by taking the solid blade definition and creating several planar cuts. The designer then works within these planar cuts to specify the 2-D wall thickness geometry at that particular section (Sreekanth, 2005). Once all the 2-D definitions are made, the resultant curves are lofted to get a final 3-D wall thickness distribution surface.

Inside the 2-D sections, the design can be separated into three components: the leading edge specifications, the trailing edge geometry, and the wall thickness distribution along pressure and suction sides. The design of each of these components was studied in depth and the resulting algorithm was based largely on the existing design practice.

4.4.1 Section Selection

In order to create the 2-D wall thickness definitions, it is first necessary to select the appropriate sections. Traditionally, these sections are planar; however, cylindrical or conical cuts may desired if the blade shape is more accurately represented with them. Figure 4.8 and 4.9 show an example of planar and cylindrical sections cuts, respectively.

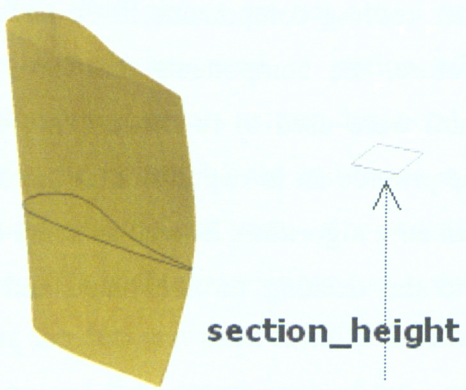

Figure 4.8 Planar section of vane

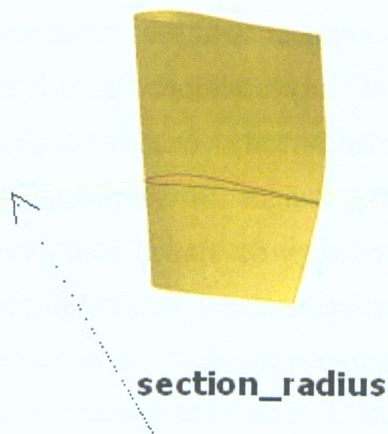

Figure 4.9 Cylindrical section of the vane

4.4.2 Real vs. Planar Offset

Before going into the detail of the wall thickness methodology, a note should be made regarding the wall thickness definition at each design section sections. The wall thickness has traditionally been defined within the design section (i.e., in a planar section the wall thickness offset curves would be created with the section as well), in other words the thickness would be defined with respect to the section normal vector (or local normal vector, if the section is not planar). During the course of the research, however, another method was suggested (Hammerschmidt, 2005), where the wall thickness offset vector would be normal to the actual surface at the specific location. The sections design would still be used, but instead of the offset with respect to the local section vector, a local vector normal to the airfoil surface will be used. This suggestion was not implemented in the present algorithm because of the complications involved with offsetting a curve in 3-D, but it is nevertheless a worthwhile concept to explore and will be studied in the continuation of this work.

4.4.4 Parametric Analysis

Design Parameters

The wall thickness algorithm was modeled closely to the development path taken by the designer. The parametric variables provided by the designer to specify the section definition are shown in table 4.1. The conical section example is shown in figure 4.10.

Table 4-1 Parameters used for section selection in wall thickness definition module

Parameter	Purpose
section_type	Indicates if the section is planar, cylindrical or conical
number_sections	Indicates number of design sections
section_height	If section is planar, indicates section height
section_radius	If section is cylindrical indicates section radius
cone_origin	If section is conical indicates cone origin
cone_angle	If section is conical indicates cone angle

Figure 4.10 Conical section of the turbine vane

Once the sections have been determined, the designer would specify the leading edge, trailing edge and pressure and suction side wall thickness distribution. The parameters used for leading edge thickness distribution are given in Table 4.2

Table 4-2 Parameters used in leading edge definition. Wall thickness design

Parameter	Purpose
le_type	Leading edge offset type: uniform, variable or stagnation point
le_crv_offset	Uniform curve offset value
le_var_offset	Variable curve offset value
le_stag_offset	Stagnation point offset value

There are several different types of leading edge wall thickness offset, including the uniform offset, variable offset and stagnation point offset. The definitions for each type are given in Figure 4.11.

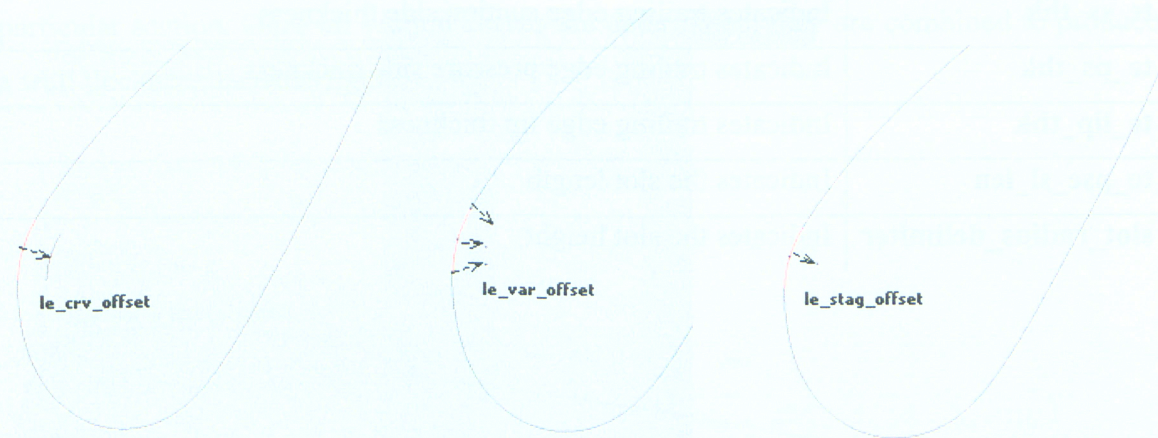

Figure 4.11 Definition of different types of offset available for the leading edge geometry

The trailing edge definition is somewhat more involved than the leading edge. There are three most widespread types of the trailing edge design: solid trailing edge, trailing edge ejection and pressure side cutback. The solid trailing edge is characterized by a geometry somewhat similar to that of the leading edge, i.e. a closed conic section. The trailing edge exit geometry allows the cooling air to escape through the trialing edge via slot. The pressure side cutback is similar to the trailing edge ejection, except for the fact that the ejection slot is not at the trailing edge but rather at the pressure side of the airfoil next to the trailing edge. Table 4.3 and figure 4.12 show the parametric definition of the trailing edge geometry.

Table 4-3 Parametric definition of trailing edge geometry. Wall thickness module

Parameter	Purpose
te_type	Leading edge offset type: uniform, variable or stagnation point
side_preference	Indicates which side the slot is adjusted to
te_ss_thk	Indicates trailing edge suction side thickness
te_ps_thk	Indicates trailing edge pressure side thickness
te_lip_thk	Indicates trailing edge lip thickness
te_pse_sl_len	Indicates the slot length
slot_radius_delimiter	Indicates the slot height

The last piece of information necessary to create a wall thickness section distribution is the thickness on the pressure and suction sides. Here, the user specifies the critical sections along the suction and pressure sides and the wall thickness values at these points (Figure 4.13). The internal algorithm then determines the optimal thickness distribution along the entire suction and pressure side and creates appropriate curves.

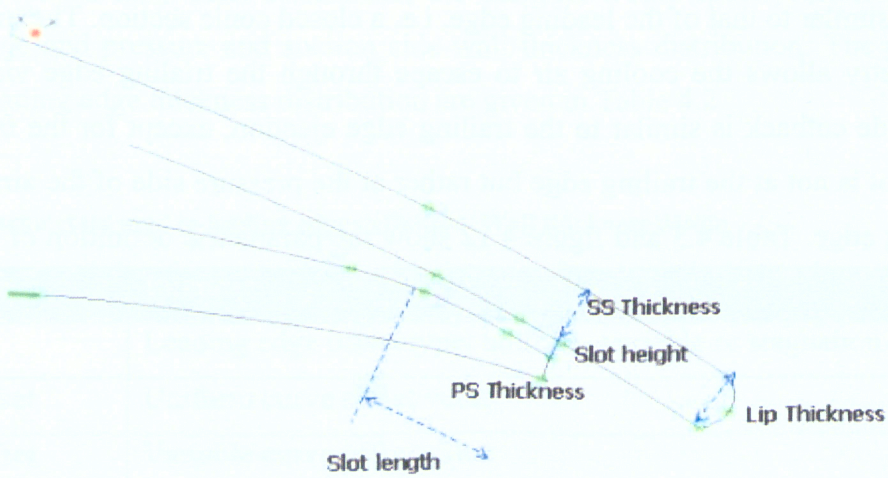

Figure 4.12 Trailing edge parameters definition

These section curves (leading edge, trailing edge, suction side, pressure side) are then splined together to produce a uniform curve that describes the wall thickness at the particular section. Once all section curves are determined, they are combined to produce a wall thickness surface (figure 4.14).

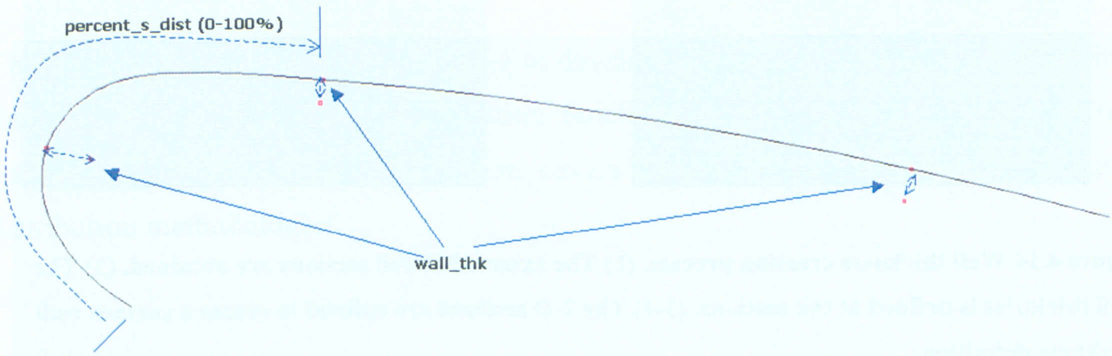

Figure 4.13 Wall thickness (suction and pressure) parameter definition

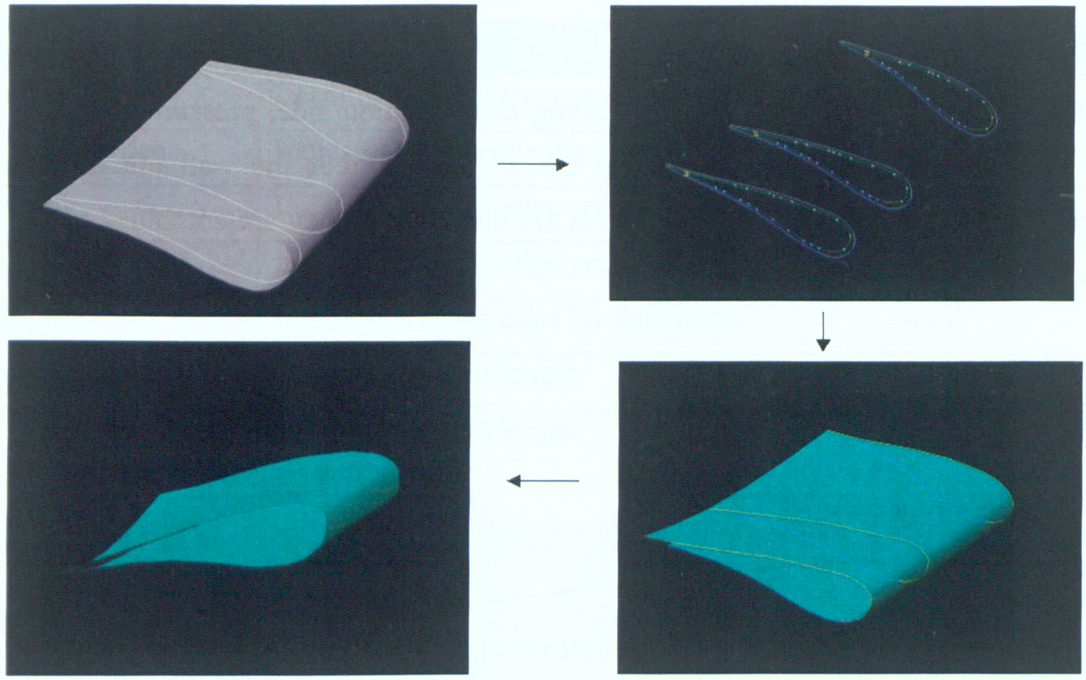

Figure 4.14 Wall thickness creation process. (1) The external airfoil sections are obtained. (2) The wall thickness is defined at the sections. (3-4) The 2-D sections are splined to create a surface wall thickness definition

Implied Considerations

Looking at the parametric input provided by the user it becomes clear that there are several implied considerations. If the designer specifies the wall thickness values only at several strategic locations on suction and pressure sides, it is clear that the overall thickness distribution should be gradually blended between the user specified locations. In addition it should be blended such that the local minima and maxima (low thickness and high thickness) points specified by the designer should remain local minima and maxima in the output. In addition to the blending should also be carried over smoothly from the leading edge curve to the suction and pressure side and to the trailing edge curves.

In the case of smoothly joining two curves together (i.e. leading edge and suction side curves) several strategies have been implemented and evaluated. One approach imposed tangential constraint on the suction side and pressure side curves at their limits, which correspond to those imposed by the leading and trailing edge geometry. These constraints were then used along with the critical points to determine the final suction and pressure side geometry. This approach had limited success as it tended to create kinks in the side curves. Another more successful approach was used where all curves were constructed independently and then re-splined by sample point method with a given constraint level (typical constraint level was maintained at around 10^{-5} inches). However, even more challenges were encountered in the process of creation of the suction and pressure side thickness curves from the critical point locations specified by the designer.

Therefore very close attention was paid to developing a thickness distribution algorithm that could create a thickness distribution function that could blend between the user specified inputs. The following section covers the study done on different thickness distribution methodologies.

Thickness Distribution Study

Thickness distribution study was conducted in order to determine the best possible way to create a continuous wall thickness distribution curve from just several critical location supplied by the designer. Figure 4.13 shows an example of parametric input provided by the designer when specifying wall thickness geometry. It is therefore up to the program algorithm to determine the best possible continuous thickness distribution curve. As mentioned before the implied considerations must be respected in order to create the geometry that reflects the designer's intent. Several thickness functions were therefore implemented and tested.

A simple linear function was implemented first. This approach has several advantages: The simplicity of the linear function and the guaranteed property of having designer's minima and maxima thickness point respected. There were however one notable drawback, namely the sharp thickness distribution change around the critical points, since the discontinuity in the first derivative. Following that a spline function was tested as a thickness distribution curve. This provided the smooth wall thickness curve, but would not always respect user specified minima and maxima.

A third approach was tried where the spline function was still used for the thickness distribution, but now in addition to the designer specified critical points, the first derivative at the local minima and maxima point was also set to zero. This allowed for a smooth distribution of the curve and at the same time maintained user specified minima and maxima. Figure 4.15 shows the comparison of the three approaches. The user specified thickness points are given in black while the three curves show the difference in the methodology.

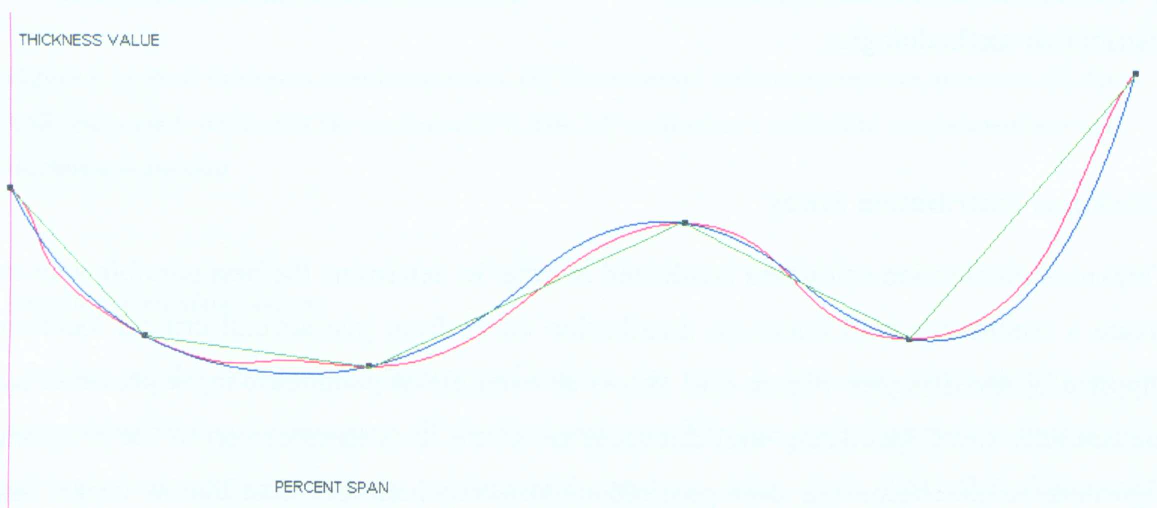

Figure 4.15 Thickness distribution study. In black are the designer specified thickness values at several critical locations. The green line denotes the linear thickness distribution, Blue line denotes spline thickness distribution while Red denotes the spline thickness distribution with implied min/max values control

Overall Results

The parametric methodology implemented in the research algorithm has been tested against some of the existing blade and vane designs. Normally the designer provided between four and six critical points for suction and pressure side along with the other necessary parameters for the leading and trailing edge geometry. The section selection depended on the original selection used in the creation of individual components through CAD interface and varied from three to seven. All of the sections used in the study were planar following the ones found in manually created models. The wall thickness surface created by splining the section curves varied from second to fourth degree in the splining direction, depending on the number of sections used (i.e. in a three section design the surface was second degree in the splined direction).

Overall the results obtained were very favorable. Not only were the thickness distribution curves identical in the design section to the original models, but they also matched very closely in the off-design sections (i.e. random sections sampled at locations other than the design sections). Figure 4.16 shows one such comparison with the original wall thickness curve shown in solid line and the one generated by the parametric research code as a dashed line. The maximum difference in the actual thickness values was observed to be in the range of 10^{-4} to 10^{-3} inches and was well below the manufacturing tolerances.

Figure 4.16 Comparison between existing wall thickness definition (solid line) and that generated by the parametric algorithm (dashed line)

4.5 Split Plane Design

4.5.1 Overview

Split plane is an imaginary plane which is used to construct most of the cooling scheme components. The plane corresponds to one of the pull planes defined on the blade or vane. The pull plane selection and definition largely depends on the blade or vane airfoil shape. If the suction and pressure sides of the blade possess high enough curvature the designer may need to include two or more split planes in the design. The split plane allows the user to design the cooling scheme definition in 2-D on a plane, plotting the outlines of the serpentine passages, pedestals, teardrops, tabulators and other components. In the CAD design environment, the designer would then project the curves onto the actual blade parts and through a series of manipulations obtain a solid geometry. The process is intuitive to the designer and simplifies the complex task of defining a highly flexible and multi-partial component in 3-D.

An example of split plane is shown in Figure 4.17, as generated by the research code and imported into CATIA V4. Here the wall thickness surface is shown with the gas path delimiting surfaces (clear) on either side, and two projections of the split plane. Two projections are used in order to accommodate the components that only make contact with one side of the wall thickness surface (i.e. turbulators lie on either suction or pressure wall, they do not join two sides together and are therefore not a function of both sides). In order to get the split plane outline in the CAD environment, the designer projects the core blade geometry and the limiting gas path surfaces on to the plane and then works within the imposed boundaries to design the cooling scheme.

4.5.2 Design Parameters

The design methodology employed in the research algorithm follows some of the design practices. The user specifies the number of split planes and their relative orientation via rotation specifications (Table 4.4). The surface is then obtained from the wall thickness generation module and along with gas path delimiting surfaces is used to project and limit on the working plane. Here however, instead of simply creating the limiting curves, the algorithm goes a step further and creates a planar surface B-spline patch (known as Coon's patch). This surface is then re-parameterized with respect to its length and normalized, meaning that u and v parameters now correspond handily with the actual geometric shape of the surface (Figure 4.18). The addition of this feature allows for a very fast and easy geometry definition on the split plane, by using u and v parameters. The designer can simply specify the location of certain component using this local coordinate system. This innovation has been fully exploited in the subsequent parametric design methodology of the cooling geometry components.

Table 4-4 Parameter definition for split plane geometry creation. Split plane module

Parameter	Purpose
number-splitplanes	Indicates the number of pull planes used on the model
angle_x	Indicates the rotation of the pull plane with respect to x axis
angle_z	Indicates the rotation of the pull plane with respect to xzaxis

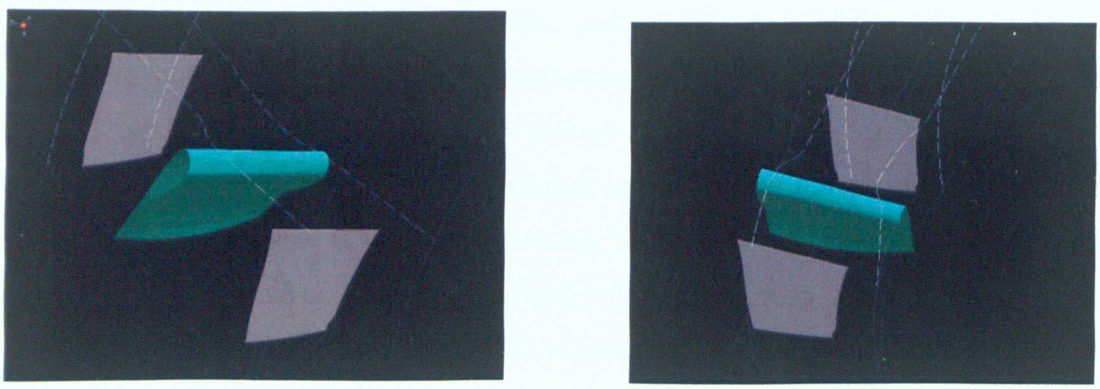

Figure 4.17 Split plane visualization

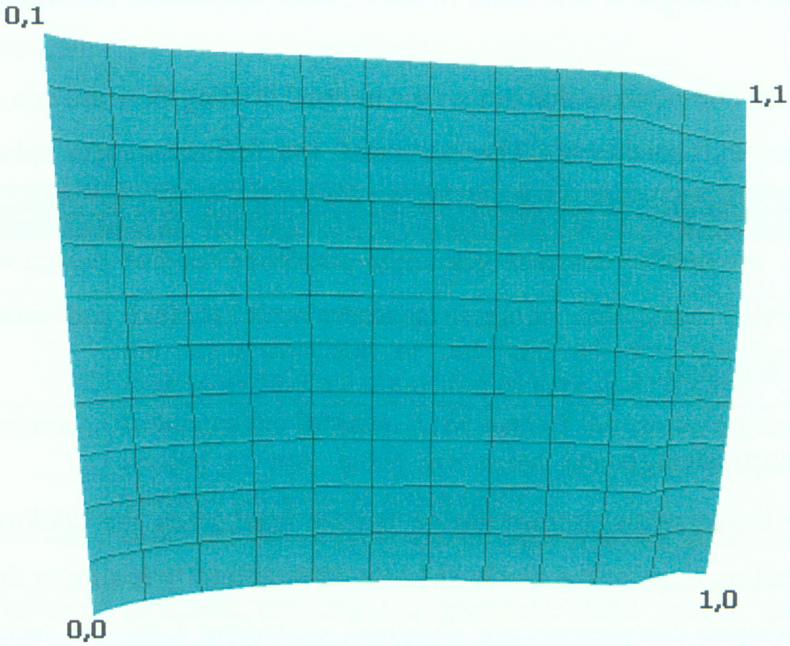

Figure 4.18 Split plane as a Coons surface with iso-parametric net, parameterized with respect to length

4.6 Feature Design

Almost all of the components that define the internal cooling flow are considered features (except for the wall thickness). These features are defined on the split plane as 2-D components and are then projected into the part to obtain a true 3-D geometry. There is a large number of various features currently employed in the design of the cooling geometry. In this work two of these features were studied and modelled, namely teardrops and pedestals.

4.6.1 Teardrops Design

Teardrops, or lands, are a cooling design feature used when the cooling air is ejected through the trailing edge of the blade. In order to structurally reinforce the trailing edge, the teardrop shaped components are designed, extending from the suction to pressure side. They are arranged in a row towards the trailing edge of the blade and are oriented according to the local flow vector. The parametric definition of the teardrop can divided into three major sections.

The first section contains a set of parameters that contain the information about the location of every land. The lands can be equally spaced or positioned at varying intervals. They can extend from root to tip of the blade or be contained to some specific section in between. The second section specifies the directional definition of teardrop. The teardrop general direction is usually inferred by taking an average root and tip split plane vectors towards the leading edge, however, sometimes the design may call for a different direction and as such the custom direction designation is available as well. The last section of the teardrop parameter definitions contains information on the individual land geometry. Figure 4.19 shows the basic land geometry and parameters that go into making it. These parameters are usually kept constant for all lands, but may be modified by the designer for individual lands.

All these considerations were taken into account in the teardrop parametric definition module. Table 4.5 shows the parameters required for land definition in the research code.

Using these parameters as designer input, a set of lands was created on the split plane, as shown in Figure 4.20.

Table 4-5 Parametric definition for teardrop (land) design. Teardrop module

Parameter	Purpose
_lan_splitplane	Indicates which split plane the land is defined on
_lan_dir_definition	Indicates the directional definition of lands: all lands pointing in the same direction or not, and whether the land direction is determined by the algorithm or supplied by user
_lan_loc_definition	Indicates the location definition of the lands: all lands are equally spaced or not, lands are fitted based on the land number or average distance between lands
_lan_dim_definition	Indicates the dimensional definition of the land: whether all lands have same dimensions or not
lan_angle_definition	Indicates the angle a particular land makes with respect to average direction (if applicable)
lan_position	Indicates the position of each individual land (if applicable)
lan_number	Indicates the number of lands (if applicable)
lan_pitch	Indicates the distance between the lands (if applicable)
major_radius	Indicates the land circle radius
taper_angle	Indicates the taper angle of each land
const_length	Indicates the length of constant thickness part of the land

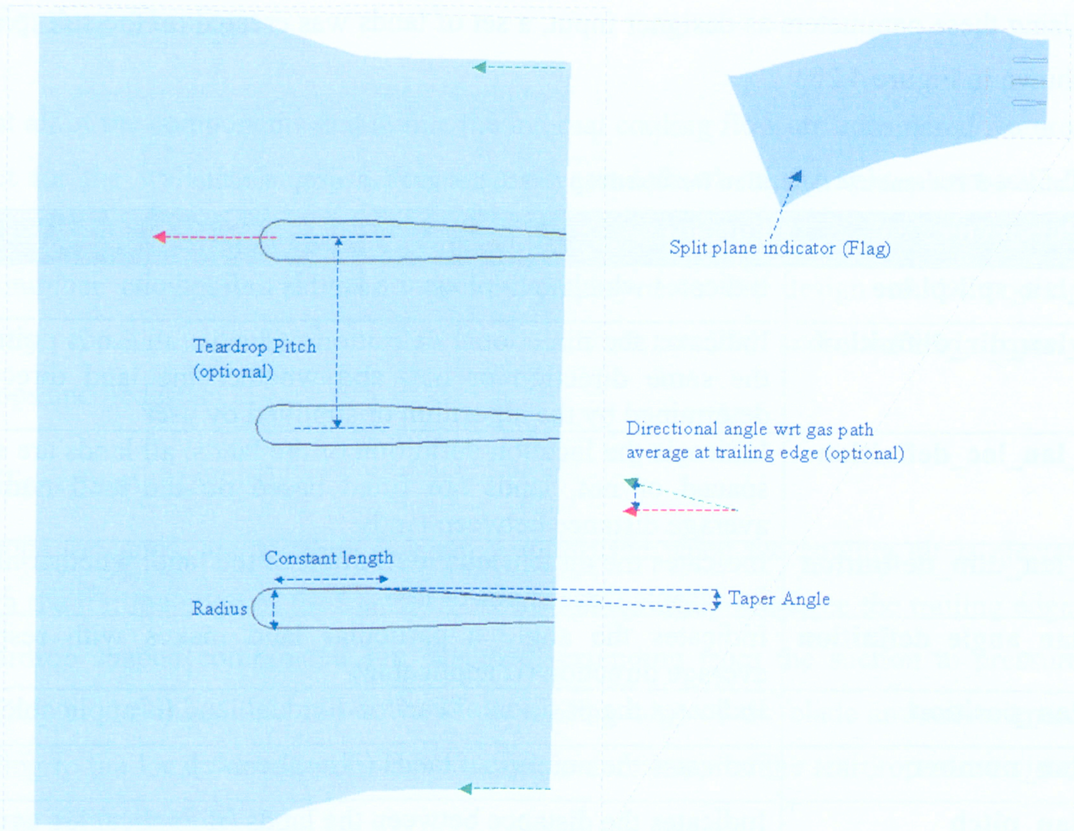

Figure 4.19 Parameter definition for teardrop design

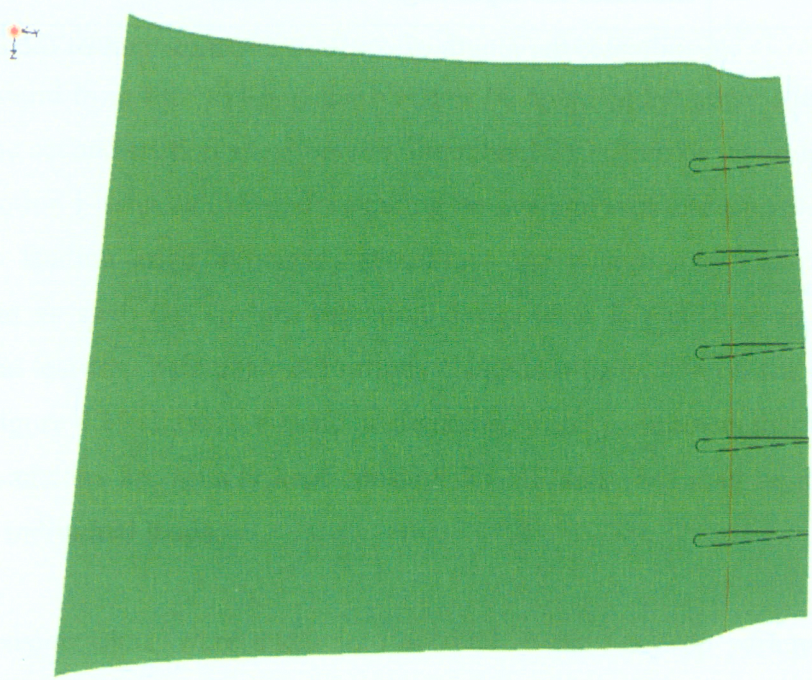

Figure 4.20 A set of lands on the splitplane

4.6.2 Pedestal Design

The pedestals are defined on the respective split planes. Each blade or vane can have a multiple set of pedestals in its geometry. The pedestals definition can be broken down into three main components. Firstly, there are the set type definition parameters. Since the pedestal location and combination are a lot more flexible than those of lands, several additional high-level indicators are included in this section, that define the type of set used, as well as indicate the general types of the set (i.e., default type sets many parameters to predefined values, making the design faster, but providing limited flexibility, while the general option allows the designer to specify all parameters manually). The parameters used in this general sub-set are listed in table 4.6.

The pedestal set consists of rows of pedestals. The second sub-set of the pedestal parameter definition module includes the row definition information. Through these parameters the type of row stagger, row to row distance, row direction and number of rows in any given set can be specified (table 4.7). Lastly, there is the actual pedestal definition set. Here, pedestal to pedestal distance and the location of the first pedestal (or start of the set) can be specified. Since the pedestal is basically a cylinder, its geometric projection of the split plane is a circle. As such, the designer needs to only specify the circle radius to complete the pedestal definition algorithm.

The algorithm takes advantage of the surface (u, v) net created in the split plane module and allows the designer to specify the location of the set using local u,v, parameters. It also allows for the rows in the given set to follow isoperimetric curves (u or v) along the surface. This provides a pedestal distribution which blends with the overall geometric composition of the blade and is very hard to reproduce without the appropriate (u, v) domain.

Figure 4.21 shows a set of pedestals on the surface produced by the research code. Note the curvature in the row shape. Here the rows follow the constant u parameter along the curve, giving it a nice blend while at the same time creating an optimal pattern for aerodynamic flow and maximizing their cooling efficiency.

Table 4-6 Pedestal set type parametric definitions. Pedestal module

Parameter	Purpose
_pedestal_set_number	Indicates the number of pedestal sets
pedestal_splitplane_reference	Indicates the split plane on which the particular set of pedestals is defined
input_option_reference	Indicates the type of set definition preferred: here the designer is given the choice between default and advanced setting

Table 4-7 Pedestal row type parametric definitions. Pedestal module

Parameter	Purpose
row_type	Indicates the type of row definition used: i.e. linear of following a constant u or v curve on the split plane
number_rows	Number of rows in a set
row_inclination_angle	Indicates the angle at which the rows are oriented (if applicable)
row_uv_direction	Indicates the parameter (u or v) which the isoparametric row curves will follow (if applicable)
stagger_type	Indicates the stagger type for the set
set_direction	Indicates the direction (+/- u, +/-v) in which the set expands after the first pedestal
row_row_distance	Indicates the distance between the rows

Table 4-8 Pedestal definition parameters. Pedestal module

Parameter	Purpose
first_pedestal_loc	Indicates the location of first pedestal, using (u, v) coordinates on split plane
pitch_distance	Indicates normalized distance between pedestals (if applicable)
radius_pedestals	Indicates the radius of each pedestal

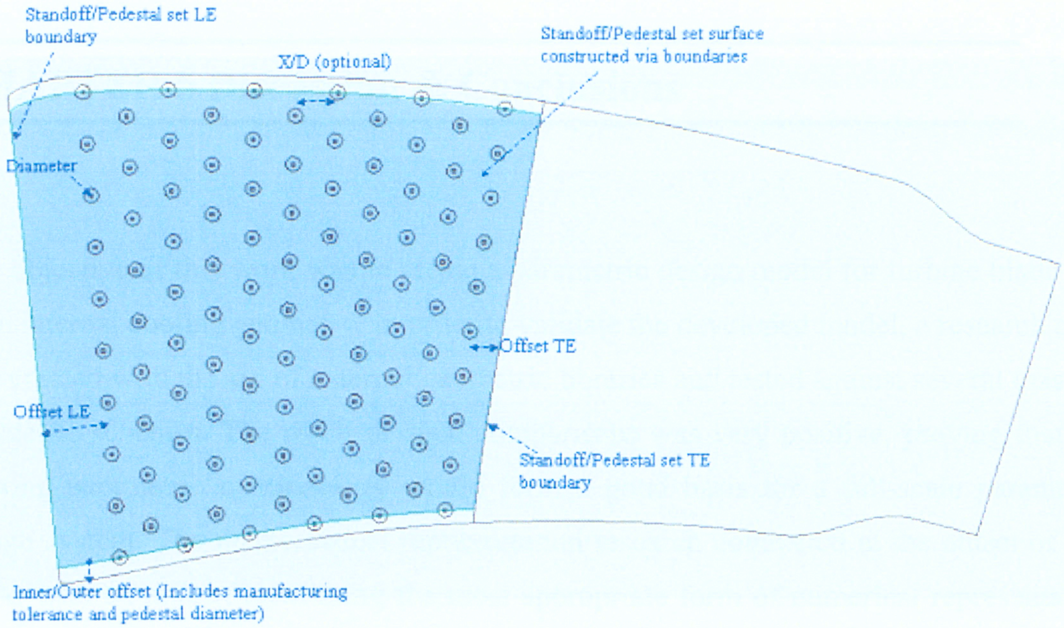

Figure 4.21 A set of pedestals on the split plane

CHAPTER 5 Discussion & Conclusions

The objective of this work was to create a parametric design model for turbine blade and vane internal cooling geometry. In order to validate the developed model, a research code was created with the aid of external geometric libraries and tested against several existing geometric schemes. The result of such comparisons was very positive, proving that the existing parametric methodology would form a good basis for a full-scale parametric design system. The mathematical representation research conducted at the outset of this work, with an aim of identifying the most appropriate form of numerical representation for the CAD independent parametric geometry generation model, examined most of the commonly used forms of representation used in the CAD industry today. Ultimately the B-spline/NURBS representation method was selected as it provided the desired high degree of geometric flexibility while still being controllable enough to implement an algorithm-based check and balance system that would minimize the amount of inadmissible sets generated. This geometric representation is also compatible with most contemporary CAD packages, including CATIA (both V4 and V5) and UniGraphics. This feature enabled a quick design comparison during the course of testing and is a very important feature in the full-scale model, necessary for fast geometry re-creation and analysis.

In order to better understand the philosophy behind the design practices currently employed in creation of the cooling schemes, some time was spent studying the physical characteristics of the jet turbine in general and blade and vane operating environment characteristics in particular. The aerodynamic, thermal and structural considerations were examined with respect to their effect on the blade geometry. Manufacturing constraints were examined as well to understand their effect on the design process.

The current capability of the research code allows the designer to parametrically create some of the most rudimentary cooling design schemes. The code is also structured in the such a way as to facilitate an easy expansion. The code integrity and methodology were verified in a series of comparative studies. In these studies the wall thickness, split plane, teardrop and pedestal geometries were successfully re-created. In some cases improvements were made on the design process, as in the case of introduction of a surface representation to the split plane definition. The robustness and flexibility of the parametric methodology was demonstrated with generation of various pedestal sets and independently defined teardrop geometries. Figure 5.1 shows an example of the flexibility permitted by the parametric methodology, as it demonstrates the various pedestal sets and land definitions created with a single execution of the algorithm.

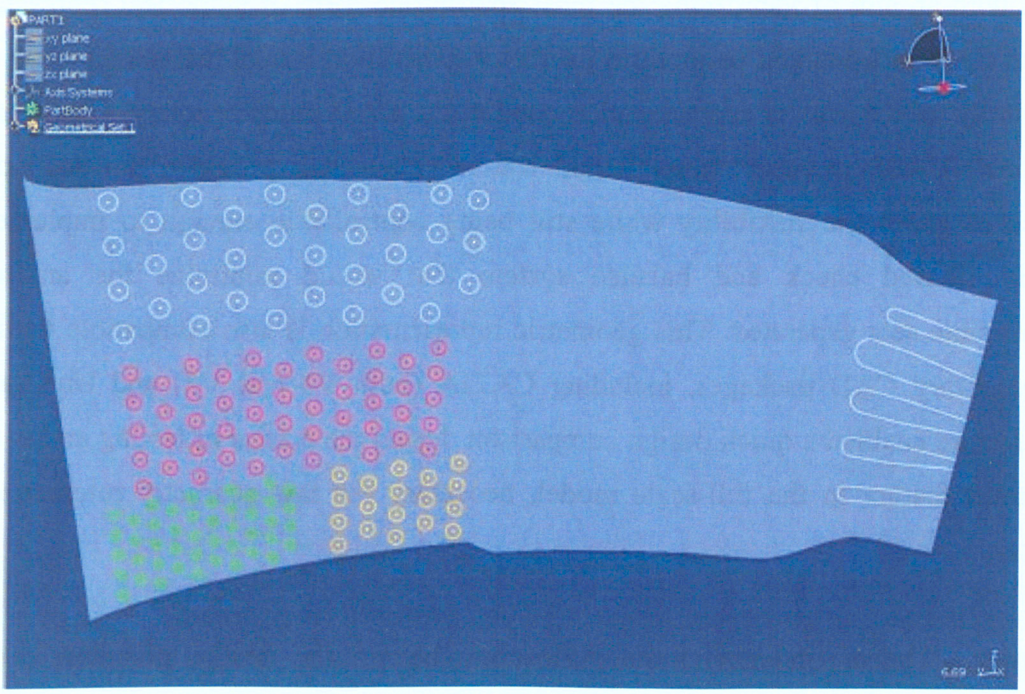

Figure 5.1 Design flexibility permitted by the developed parametric methodology algorithm. Here multple sets of pedestals are created in a single generation cycle

In order to demonstrate the further usefulness of this design methodology, an additional module was introduced with a purpose of transforming the 2-D feature definitions into a 3-D geometric representation on a part with surfaces. Figure 5.2 shows the true 3-D components generated by the algorithm. This capability allows for a rapid creation of solid geometry inside the CAD software for further analysis and provides more useful information for the designer

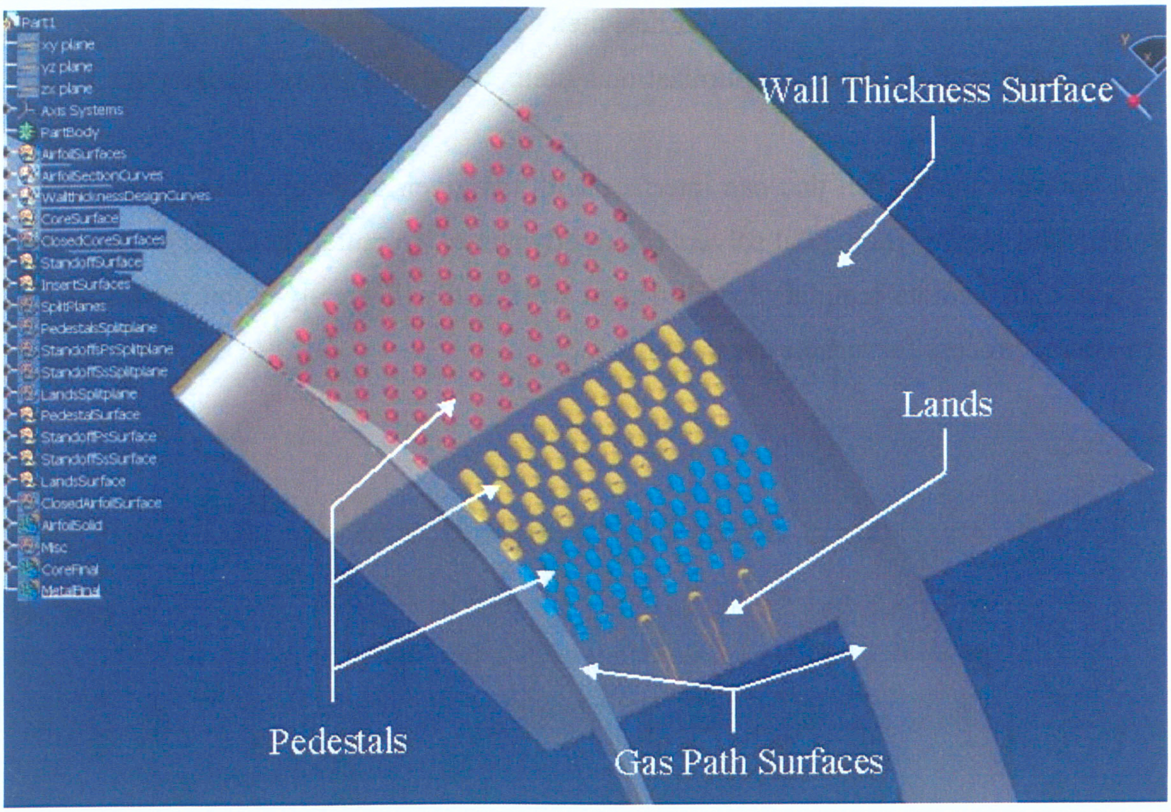

Figure 5.2 3-D Components generated by the parametric algorithm. The components are represented with surfaces

Based on the results obtained from the test runs, the parametric design methodology devised in this work can be developed even further to include more features and eventually cover all of the variable geometry used in the turbine blade and vane design. The expansion of the parametric system within the given framework of CAD independent algorithm using the B-splint/NURBS based parametric methodology will allow the designer to create and analyze various blade and vane designs on a preferred CAD system.

The parametric design methodology will allow the designer to create desired geometries faster and with greater flexibility. It will also allow new designers to learn the skills sooner as they will be given a structured, parameter based approach to the design. The time saved on the geometry generation can then be spent on more in-depth analysis and research and as a result, lead to better designed components. The parametric system itself can be evolved further into an optimization system, with even greater design potential.

The optimization system that can interface with the given parametric algorithm, a CAD system and various numerical solvers (thermodynamic, structural, fluid, etc) will provide a rapid and accurate template for all design work and will greatly simplify and shorten the design process for turbine components.

APPENDIX A Research Algorithm Pseudo-code

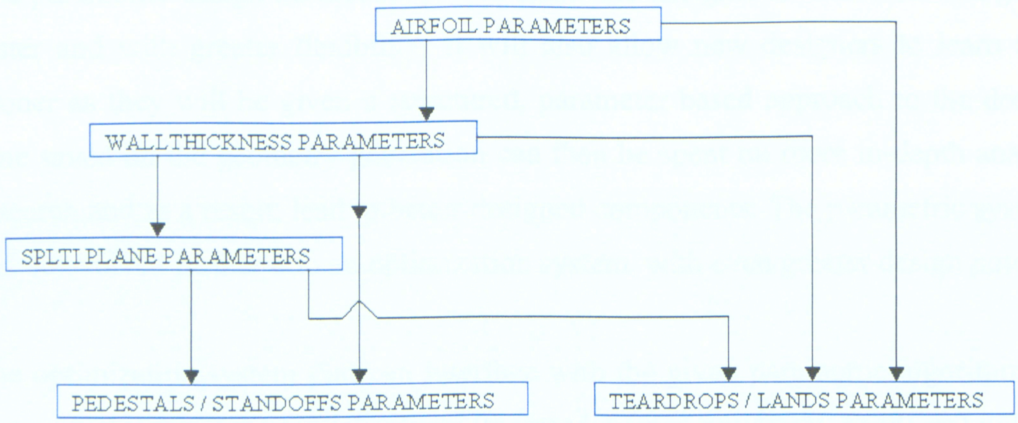

Figure A-1 Algorithm Flow Chart

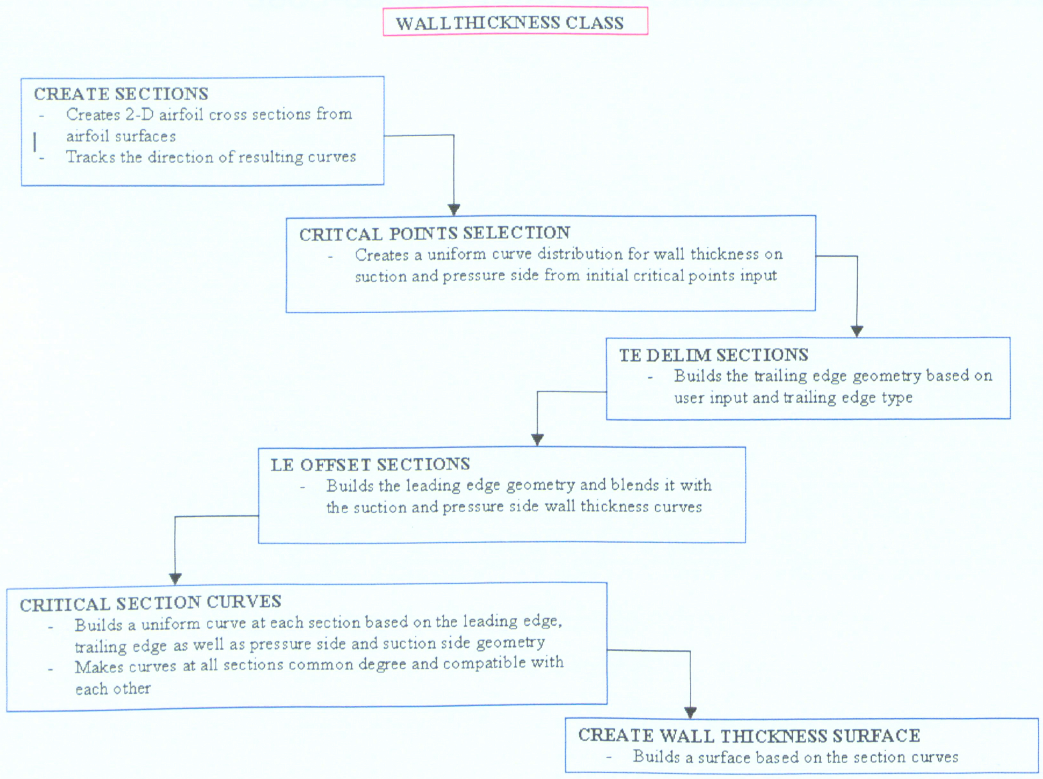

Figure A-2 Wall thickness class algorithm chart

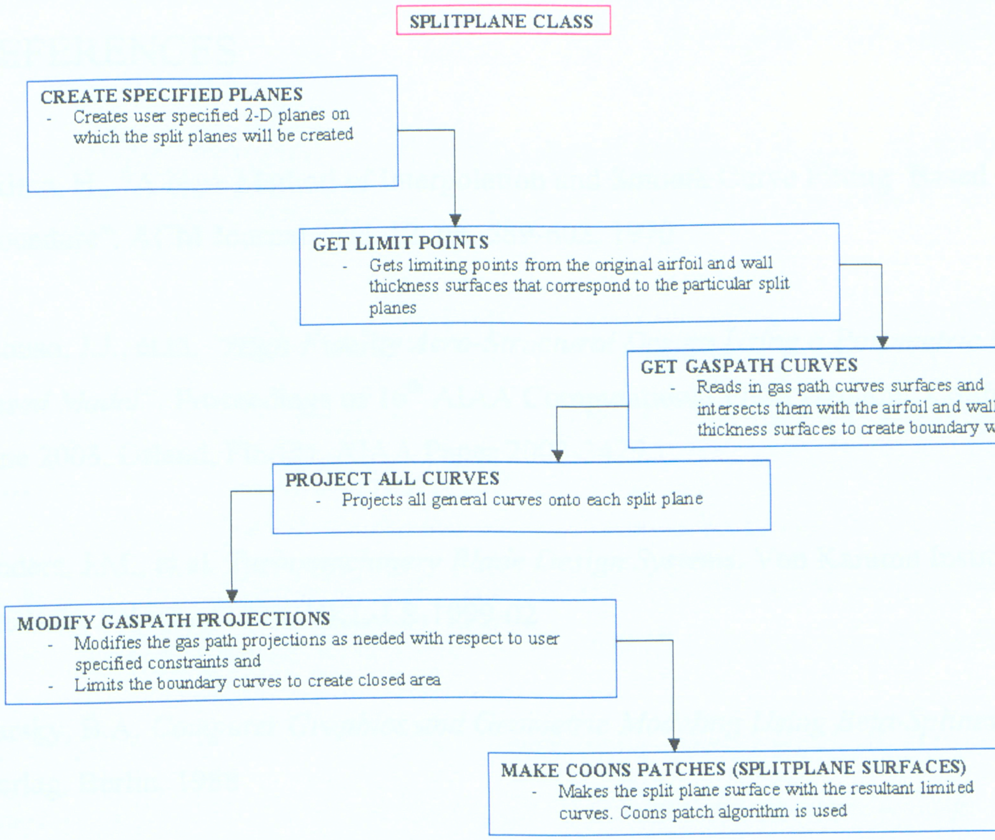

Figure A-3 Split plane class algorithm chart

REFERENCES

Akima, H., "A New Method of Interpolation and Smooth Curve Fitting Based on Local Procedure". ACM Journal, Vol. 17, pp. 589-602, 1970

Alonso, J.J., et.al. *"High-Fidelity Aero-Structural Design Using a Parametric CAD-Based Model"*. Proceedings of 16th AIAA Computational Fluid Dynamic Conference. June 2003, Orland, Florida. AIAA Paper 2003-3429

Anders, J.M., et al. *Turbomachinery Blade Design Systems*. Von Karamn Institute, Brussels, February 1999, VKL-LS-1999-02

Barsky, B.A. *Computer Graphics and Geometric Modeling Using Beta-Splines*. Springer-Verlag, Berlin, 1988

Corral, R., Pastor, G. *"Parametric Design of Turbomachinery Airfoils Using Highly Differentiable Splines"*. AIAA Journal. Vol. 20, No. 9, March 2004, pp 335-343

De Boor, C., *A Practical Guide to Splines*. Springer-Verlag, New-York, 1978

Giles, M.B., Drela, M. *"Two-Dimensional Transonic Aerodynamic Design Methods"*. AIAA Journal. Vol. 25, No. 9, September 1987, pp 1199-1206

Gladden, H.J., Simoneau, R.J., *"Review and Assessment of the Database and Numerical Modeling for Turbine Heat Transfer"*. Towards Improved Durability in Advanced Aircraft Engine Hot Sections, edited by D.E. Sokolowski, IGTI, Vol. 2, pp. 39-55

Hammerschmidt, Arthur, in private correspondence with and with documents from. Application Developer, Computer Science Corporation, 2005

Han, J.C., Dutta, S., Ekkad, S.V., *Gas Turbine Heat Transfer and Cooling Technology*. Taylor & Francis, New York, 2000

Han, J.C., Park, J.S., Lie, C.K., *"Heat Transfer and Pressure Drop in Blade Cooling Channels with Turbulence Promoters"*. Texas A&M University (prepared for NASA CR-3837)

Kennon, S.R., Dulikravich, G.S. *"The Inverse Design of Internally Cooled Turbine Blades"*. Journal of Gas Turbines and Power. Vol. 107, pp 123-126

Korakiantis, T., Pantazopoulos, G.I. *"Improved Turbine Blade Design Techniques using 4th Order Parametric-Spline Segments"*. Computer Aided Design. Vol. 25, No. 5, May 1993, pp 39-329

Lakshminarayana, B., *Fluid Dynamics and Heat Transfer*. John Wiley, New York, 1996

Martin, T.J. *"Computer-Automated Multi-Disciplinary Analysis and Design Optimization of Internally Cooled Turbine Blades"*. PhD Thesis. Pennsylvania State University. May 2001

Mortenson, M. E., *Geometric Modeling*. John Wiley & Sons, New York, 1985

Papple, Michael, in private correspondence with. Staff Engineer, Department of Turbine Cooling and Static Structures, Pratt & Whitney Canada, 2005

Patrikalakis, N.M., et al. *Computational Geometry*. Lecture Notes. Massachusetts Institute of Technology, Cambridge, USA, 2003

Patrikalakis, N.M., Maekawa, T. *Shape Integration for Computer Aided Design and manufacturing*. Springer-Verlag, Berlin, 2002

Piegl, L., *"Coons-type patches"*. Computer and Graphics, Vol. 12, No. 2, pp. 221-228, 1988

Piegl, L., *"Key Developments in Computer-Aided Geometric Design"*. CAD, Vol. 21, No. 5, pp. 262-273, 1989

Piegl, L., Tiller, W. *The NURBS Book, 2nd ed.* Springer-Verlag, New York, 1997

Quan, X., Dutta, D. *"Design of Heterogeneous Turbine Blades"*. Computer Aided Design. Vol. 35, 2003, pp319-329

Sautner, M., et al., *"Determination of Surface Heat Transfer and Film Cooling Effectiveness in Unsteady Wake Flow Conditions"*. AGARD Conference Proceedings 527, pp 6-1 to 6-12

Song, W., et al. *"Turbine Blade Fir-Free Root Design Optimization Using Intelligent CAD and Finite Element Analysis"*. Computers and Structures. Vol. 80, 2002, pp 1853-1867

Sreekanth, Sri, in private correspondence with. Principal Engineer, Department of Turbine Cooling & Static Structures, Pratt & Whitney Canada, 2005.

Suo, M., *"Turbine Cooling"*. In Aerodynamics of Aircraft Gas Turbine Engines. Edited by Oates. Air Force Aero Propulsion Lab., AFAPL TR 78-5, pp. 19-1 to 19-23, 1978

Tiller, W., *"Rational B-splines for curve and surface representation"*. IEEE Computer Graphics and Applications, Vol. 3, No. 6, pp. 61-69, 1983

Treager, J.R., Aircraft Gas Turbine Engine Technology, 2nd ed. McGraw-Hill, New York, 1979