Revisiting Traditional Curve Lofting to Improve the Hull Surface Design Process

Marcus Bole, AVEVA Solutions Ltd, Gosport/UK, marcus.bole@aveva.com

Abstract

The popularity of surfaces in hull form design often results in approaches entirely based on curves being considered inferior. Yet they can offer a quick and simple solution to design process often with far less definition. Moreover, when the definition geometry is unfair or incompatible, curves indicate the affected region in a more obvious manner than possible with a surface. This paper discusses specific characteristics of existing surface design and their associated generation algorithms which may cause imperfections and offers up a parallel strategy in the form of a modern interpretation of traditional curve lofting. It may be used to explore hull form shape in shorter time scales and supports a greater range of shape generation algorithms. The technique may be used both as a way of indirectly improving a surface representation through visual comparison or used to directly improve a surface using fitting techniques.

1. Introduction

The use of mathematical surfaces in hull form design is now considered mature technology. NURBS (Non-Uniform Rational B-Spline Surfaces) are all but the dominate mathematical representation assisted by their adoption into CAD exchange file formats and the control polygon, a geometric grid of control points which easy to manipulate using computer peripherals such as mouse or digitisers. However, it still takes a great deal of skill to create a hull form and even experienced users can often find that it takes a considerable amount of time to achieve satisfactory definition. Why is this? The fact that it is still a challenge to create a new hull surface definition means that new software tool continue to appear but seldom is there a significant gain in the design experience. Consequently, rapid design tools such as parametric hull generation continue to be a stated requirement for some, providing a quick solution which allows them to focus on designing more important aspects of a vessel without getting slowed by the lack of a hull surface.

Fig. 1: Challenging design areas of a hull surface are often found at features where a wide variation in curvature is required, such as the top of the bulb or at a stern post.

Designing a smooth and fair surface is not difficult. However, modern vessels, both ship and submarines, often have surface features which improve hydrodynamic performance, allow the ship to be manoeuvrable, make shell plates easier to produce, welded joints easier and more robust. Sometimes vessels have unique features such as bulges to improve internal cargo handling or to contain sensor systems. The surface design challenge begins once it become necessary to incorporate these feature into a smooth surface. The surface may need to transition from areas of low single...
curvature to high double curvature in a short distance. There may need to be a corner in the edge of a surface but without a knuckle line. Incorporating these features often leads to distortion in the definition and the surface can become warped with no obvious means of correction. A designer can sometime try to create a shape which is not geometrically possible and will be faced with frustration. Furthermore, a designer may not recognise that definition needs to be refined to improve shape because often refinement increased the complexity of the definition and is avoided at the early stage if changes in the design are expected. This situation can only be resolved with experience and this only comes with time. Time is expensive and not always available when operating commercially.

The author has spent several years developing surface design software capable of rapid design and change. However, the fact that the surface design process remains challenging has continued to be a puzzle even with improvements to user interface design and the development of tailored hull surface design operations. In recent years, the author has had the opportunity to work with AVEVA Lines, one of the first computer based hull design tools in a role which often involves training new users of the software. AVEVA Lines is used to create a hull surface but design of shape is primarily achieved using curves in a procedure reminiscent of traditional lofting procedure. This approach is often considered outdated by users more familiar with modern surface design software but if the correct approach is followed it is capable of consistently achieving higher quality surfaces than other techniques and is particularly rewarding when it comes to production fairing. While surfaces have been a great technology for geometric modelling and data exchange, are they the best technology to adopt when designing complex surface shapes like ship hull forms?

2. Designing with Surfaces

In general, modern hull surface design software tools present the user with two approaches. The user may manipulate the surface directly, moving the definition into place and controlling shape using control points. Alternatively, the surface may be generated from a network of curves representing cross sections through the surface. The latter method is often referred to as Cross Sectional Design and has the advantage of supporting descriptive attributes added to the definition which can be used to create specific shapes in the surface and eliminate the need for the user to manage large groups of control points to create, for example, a tangent along the edge of a surface. Although direct surface definition manipulation and Cross Section Design appear as alternatives, Cross Sectional Design can be seen as an evolution of direct surface manipulation as far as the design tool requirements and implementation is concerned. It is a logical extension resulting from the challenges users encounter as they attempt to design a more complex hull surface and control shape as part of an engineering process.

![Fig. 2](image_url)

Fig. 2: It is often difficult to manage hull forms represented by individual surfaces. Once the smaller features are refined enough, other areas of the surface with low curvature often have many superfluous control points making controlled change harder.

Working with a single NURBS control polygon is initially very easy. However, as the hull surface need to incorporate specific features, such as parallel middle body and bulbs, different resolutions of definition are required for each shape. Using Knot insertion it is possible to refine the surface the refinement operation introduces control points across the surface. This may include areas of low
curvature making subsequent design changes more difficult due to the additional degrees of freedom introduced by the extra control points. Techniques such as Hierarchical B-Spline refinement, Forsey and Bartels (1988), Subdivision Surfaces and T-Splines, Sederberg et al. (2003), improve the situation by localising refinement to the area of interest but often it is useful to describe the hull surface using several patches. A major benefit is that as surface patch edges behave like curves the designer has more direct control of shape along that edge. A number of hull design software tools offer the possibility of creating a design by connecting patches together. Subdividing the hull into patches also assists the migration of a design into solid modelling tools because these systems cannot accept parametric degeneracies within the internal definition of a surface as it causes surface/surface intersection algorithms to fail at degeneracies where derivatives become undefined.

Separating a surface into patches assists the initial construction of a surface but it is difficult to use it in areas where there is not a break in curvature because it is not always possible to maintain a smooth transition. In these cases, the user has to make do with the control points and often this definition can become quite distorted around bulbs and propeller skegs. When definition becomes distorted detailed fairing later in the project may be impossible and it may be necessary to redefine the hull surface.

### 2.2. Cross Sectional Design

Cross Sectional Design offer as solution when designing complex hull forms by allowing the designer to explicit define slices through the surface. By using a collection of intersecting curves, the designer can control the density of definition using more curves where there is increased detail in the shape and minimal curves if the surface is flat. Surface generation algorithms are used to interrogate the design curves and generate a surface consisting of multiple patches. Consequently, the design definition and the resulting surface representation are no longer directly geometrically dependent. This allows additional information to be associated with the curves which can be used by the algorithms to produce curve tangency, knuckles and surface tangencies. Curve definitions can be dependent on intersections with other curves, actively managing the number of free control points in favour of the designer. The process used to generate the surface is more complex than direct surface definition. Briefly, it can be split down into the following four steps:

**Step 1 – Generate Representation Topology:** Create a boundary representation, forming vertices from curve intersections, edges from curves and identify faces which will become the location of patches, Fig.4. Algorithms such as Bagali and Waggenspack (1995) and Hopcroft and Tarjan (1974) may be used to achieve this. However, these techniques may fail in situations where the boundary representation become topologically ambiguous (if there are far fewer curve intersections than curves) but a hull surface designer is unlikely to experience this.

![Fig. 4: creating a boundary representation structure from intersecting curves, Lee (2001).](image)

**Step 2 – Establish Ribbon Tangents along each Definition Curve:** Once the topology is established a surface tangent needs to be generated along each curve, Fig.5. This is achieved by fitting a (curve) function that models the tangent vector field along the length of the design curve by interpolates the tangent vectors all of the intersecting curves at the point of intersection, Fig.5. In reality this is again a
non-trivial activity as it is necessary to deal with all the different configurations of curves the designer may generate and deal with the different attributes that can be associated with curves, any discontinuities due to knuckles and intersecting curves where tangent vectors are parallel. As shape of the ribbon tangent along the curve is created by fitting through specific points along the curves, it is only, at best, an estimate of the intended shape.

Fig. 5: A Ribbon Tangent generated along a design curve.

Step 3 – Subdivide non-four sided faces to allow representation by four sided patches: Faces which are not surrounded by four edges need additional processing when the mathematical patches used to represent the surface are restricted to four edges. Two and three sided faces may be handled by using degenerate sides although this has implications for curvature continuity. Faces with more than four sides need to be subdivided into faces which can be represented by four-sided patches. Faces which resemble a regular polygon can be subdivided by creating a central vertex, subdividing the surrounding edges halfway along and then connecting the mid-edge vertex to the central vertex, Chiyokura (1988), Fig.6 (left). An alternative is to map the face on to a rectangle, identifying four ‘corner’ vertices then adding subdividing edges to connecting the non-corner vertices to the opposite edge, Fig.6 (right). When representing hull forms, as multi-sided faces are often geometrically rectangular in shape the latter method is more suitable. However in both cases it is necessary to generate new curve geometry to represent each subdividing edges. The shape of these curves is based on derivative information available at the edges of face but is ultimately an estimate of the shape of the surface across the face.

Fig. 6: Subdividing non four-sided faces into faces that can be modelled by a four sided patch.

Step 4 – Blend patches using edge curves and ribbon tangents: The final stage is to generate the patch itself. This involves extracting the curve and tangent information from the edge and blending a surface patch using techniques such as Coons Patch, although Subdivision Surface techniques may be used. Again, the use of a blending technique to generate a surface which is an estimate of surface shape based on position and tangent information available at the edges of the face. There are different algorithms and mathematical patches that may be chosen to represent the surface but they must all address the aspect of “twist”, which is a need for the rate of change of the tangent vectors along each edge to be compatible at each vertex, and will be discussed in 3.4. Furthermore, the advantages of using B-Splines to represent the curve network can be seen with the benefits of subdivision it is possible to extract sub-curves from the main definition as discrete B-Spline curves and use B-Spline operators, Piegl and Tiller (1999), to subtract and multiply edge and tangent representations together.
3. Factors Affecting Surface Quality

Cross Sectional Design aims to overcome the limitations that occur when directly manipulating the raw surface definition and, in doing so, introduces a considerable amount of software processing. Ribbon tangent information is generated at curve intersections and must be interpolated along curve edges. Non four-sided faces may need to be subdivided. Patches are generated from the position and tangent information, by blending information from surrounding edges. These tasks interpret information available within the curve network but may not always produce the most ideal shape.

3.1. Choice of Patch Mathematics

Once an alternative structure is used to represent the hull definition the possibility of using other surface representation techniques beyond NURBS is now open and techniques such as Transfinite and Subdivision Surfaces can be explored. However, in most cases a blending method based on the Coon’s Patch or a Boolean sum of surfaces is necessary. In the author’s experience, the choice of surface mathematics does not have a significant effect on aesthetic shape. It is only once compatibility between surface patches is analysed numerically that differences can be seen. Transfinite surfaces, Coon’s Patches with Gregory Square, Gregory (1974), or Browns Square, Barnhill (1993), seem best suited to producing patches in situations where Ribbon tangent information is not completely compatible and this means that patch shape may be less prone to undesirable shapes. However, these surfaces can only be represented as mathematical functions and are not supported by any CAD exchange formats. Subdivision Surfaces were found to be promising although there were few techniques, Levin (1999), which offered the ability to interpolate the curve network. Those that did were found to be much slower and did not offer the ability to perform detailed surface analysis because only a faceted representation was generated. In the end, the software developer is faced with a dilemma. Use NURBS surfaces because of their support in CAD formats even though they have poor Twist capability or use a surface representation performs better but is unique to the software. In practical terms, if a designer adds more definition or spends more time to perfect the surface the quality argument becomes academic. NURBS succeed because of interoperability.

3.2. Blending Patches

Blending is fundamental to the patch generation process (excluding Subdivision Surfaces), combining the position and ribbon tangent from each edge. In the four-sided case, two cubic surfaces are generated between the two opposing pairs of edges and a third surface generated from derivatives at the corners to account for twist. The three surfaces are then combined producing a representation entirely based on edge information. The representation could be considered an estimate or average of the information at the edges. As edges diverge from a shapes that could be represented by a cubic function the estimate of the surface shape can become poor. Certainly the shape of the centre of the surface is beyond the control of the user. While hull surfaces are generally smooth, often a sharp change in curvature can occur near boundaries, from low to high particularly near the ends the hull form to accommodate hydrodynamic and production requirements. Blending algorithms find these cases hard to deal with and the user needs to be aware that additional definition curves are necessary to subdivide changes in curvature. However, additional curves increase the number of patches and the calculation time.

In general, blending is performed using cubic interpolation and can capture both the position and tangent information. Curvature is not blended across the surface and consequently is not constrained. This is surprising because curvature continuity is a requirement. In order to blend curvature it would be necessary to perform bi-quintic blending and as a result the whole surface generation process would become more complex with the need to gather and interpolate the curvature around the curve network like tangency. It is also likely to make fairing the definition harder as the additional constraint would mean that the “sweet spot” of fairness may be more elusive.
3.3. Interpolating Tangent Ribbons

Tangent ribbons estimate tangency along a definition curve based on crossing (tangent) vectors of any intersecting curves. In general, the fit may be performed by treating tangent vectors as points and using a curve fitting algorithm. Koelman (1999) suggests the use of a monotonic interpolation algorithm. Representing both curve edges and tangents as B-Spline curves allows the use of B-Spline Operations, Piegl and Tiller (1999), which are particularly useful when generating NURBS patches as it minimises the need to evaluate the curve functions by working directly with the control points coordinates.

There are, however, some limitations. The magnitude of the tangent vector produced by the fitting procedure varies when interpolated in Cartesian space, suggesting that some form of spherical interpolation along the lines of SLERP, Shoemake (1985), might perform better. A further limitation is that in cases where a curve edge turns through a large angle the interpolation of the ribbon tangent can start to diverge from the ideal shape, Fig. 7. This leads to unfairness in the surfaces produced by the blending algorithms even though the resulting surface is mathematically smooth. This can be resolved by adding further design curves to cross the edge refining the tangent ribbon, along with additional patches.

![Fig. 7: The tangent ribbon may not produce a perfect estimate of the surface tangent along an edge as it does not account for the position of neighbouring curves which do not cross it. It may only be corrected by adding further curves which do cross the design curve (right). The phenomenon is exacerbated if the tangent ribbon vectors are orthogonalised with respect to the curve as indicated by most literature.](image)

3.4. Representing Twist

The characteristic often termed “Twist” is the need for the rate of change of tangency along edges adjacent to a vertex to be compatible. Methods such as the Coons Patch with Gregory Square or Browns Square or the Gregory Bezier patch, Chiyokura (1983), can handle twist. Methods such as NURBS, with regular control polygon grids, are particularly poor at representing twist resulting in subtle unfairness due to loss of continuity between patches and warping of the surface. This phenomenon is not something that is highlighted to users of surface design software and they must proceed with whatever experience they have to resolve the situation. The limitation comes from the fact that each patch corner, the surface tangents at the control points adjacent to the corner point are dependent because of their shared connection to the control point diagonally in from the corner point. To represent twist correctly the surface tangents need to be independent. However, as both are connected, twist cannot be represented correctly at the corner on either edge and the resulting surface is compromised, Fig. 8.

A further limitation appears when twist is considered for a number NURBS patches around a vertex, Fig. 9. The need to maintain tangential continuity means that pairs of control points representing the surface tangent along each edge should be aligned with respect to the tangent ribbon. Around vertex this results in a connection between all control vertices adjacent to the vertex. In the case of four patches a “square” of control points is formed. It is possible to skew this arrangement of control
points to address twist. However, if there are an odd number of patches around vertex the points can no longer be skewed and the ability to account for twist is lost all together, effectively the position of these control points becomes locked and flattening will be experience in the surface around the vertex. This situation is particularly bad for Cubic Bezier patches as ultimately all control points are used to address continuity and twist around vertices without any further control points (degrees of freedom) remaining to control the internal shape of the patch.

![Difference between tangent vectors](image1)

**Fig. 8:** At a corner, a NURBS surface may not be able to match the twist characteristics exactly because the tangents along each edge are controlled by a single vertex. Consequently, the patch may not be able to achieve tangential continuity with its neighbours.

From a tangent and curvature continuity perspective this limitation is significant, but many surface design tools use NURBS surface and they are an important component in Solid Modelling. This suggests that the continuity experienced in practice is adequate and adding further design curves is an acceptable method of improving surface shape.

![Fig. 9](image2)

**Fig. 9:** When four patches meet at a vertex there is some adjustment for twist. However, when an odd number meet this adjustment is lost and zero twist should be enforced causing a flat spot in the surface at the vertex.

### 3.5. Non-Four Sided Faces

Non-Four sided faces can frequently occur in a hull form definition and often they appear in difficult places of high curvature and where the surface needs to accommodate specific features, such as at the bulb root or at the transom around propeller skegs or stern posts. Often they appear because the designer has not refined the definition to the extent where the surface shape is well defined, however, in early design it is desirable to minimise the amount of definition around detailed features. Consequently there is an expectation that the software should make a proper estimate of the shape across any space. Surface patches which support non-four sided arrangements do exist both in transfinite, *Gregory (1983)* and control point form, *Zheng and Ball (1997)*. However, none are supported by any CAD exchange format which means it will be necessary to convert the geometry into something which can be represented by a four sided patch, i.e. NURBS. This is unfortunate as, compared with the subdivision techniques discussed in 2.2, the shape produced by these representation is far superior. Particular in cases where the face is quite distorted and the curves used to subdivide the face in four-sided cases estimate shape poorly.
4. Designing a Hull Surface using Curve Lofting

While hull surface design gives the user a powerful ability to create an accurate shape in three dimensions, the complexity of the tool can get detract from the creative process itself. Unfortunately, as a surface design tool becomes more powerful the user is exposed to more complexity in terms of the number of tools and the structure of the definition. With the additional complexity that comes with approaches like cross sectional design it is understandable why surface definition based on a single NURBS control polygon remains popular. However, often the capabilities of cross sectional design outweigh the simplicity of basic techniques when it comes to productivity when dealing with complex surfaces like ship hull forms.

Curves prove to be very effective when designing the hull boundaries, features and cross sectional shapes. Furthermore, if surface patches are do not have the desired shape it has become standard practice to increase the number definition curves. However, this is not a particular efficient way of improving surface quality. It reduces the magnitude of the inconsistencies below a detectable tolerance but it does not eliminate it. With all this effort being made to improve the skinning results, it begs the question what would happen if curves alone are used to represent the hull form? Can we learn anything from traditional lofting techniques to improve the surface design process?

AVEVA Lines (initially developed as BLINES by the British Ship Research Association in the 1970s) began as a software equivalent to draughtsman’s spline and ducks replacing them with B-Spline curves and was advanced for its time considering that little had been published on B-Splines at the time of development. Over the years it has acquired facilities to generate surfaces from the curves but fundamentally it remains a design tool based on curve lofting, with the user advised to complete the task of designing the shape of the curves before even considering generating a surface. The software is popular in shipyards where the traditional ideas of lofting hold and with mature users that may have be trained in manual lofting techniques during their apprenticeships. However, for new users who have only experienced modern surface design tools it is challenge to introduce the software and highlight the benefits of the approach. This challenge is not helped by the fact that the software does not appear to have an extensive set of features; it does not need it, and does not readily reveal how the loft process should be approached. The process is one that users familiar with traditional techniques would understand but for new users it is an approach that needs to be discovered and this applied to the author on joining AVEVA.

The approach used in AVEVA Lines is one based on curve intersection and curve fitting and demonstrates that all hull surface design can be accomplished with these basic tools. This intersection can be performed on individual and groups of curves and macros can be generated to intersect and fit successive groups. However, it is not possible to have this performed while the user is moving control points with the mouse, a feature that a modern hull surface design tool would have. So the question arose: What would the design experience be if the traditional lofting technique of curve intersection and curve fitting was implemented in a software environment representative of modern hull surface design tools?

4.1. Traditional Lofting using Cross Sectional Design Definition

Rather than develop a completely new tool, the author’s existing surface design tools, Bole (2010), based on cross sectional design were chosen to support the investigation. The design curves used in the investigation support features typical of other cross section design tools, such as the ability to represent tangents in the curves, create straight and blended segments and assign specific surface tangents along the length of a curve. Unlike AVEVA Lines, these design curves support the ability to define a (child) curve control point which is embedded on another (parent) curve, a relationship consistent with the ‘Bead’ concept as described by Letcher et al. (1995). Changes to parent curves propagate through into child curves resulting from the hierarchy of relationships. In AVEVA Lines, it is possible to achieve a similar effect but the relationships are established dynamically by the software rather than captured in the curve definition structure.
A traditional lofting process can be set up using the design definition curves by adding subsequent curves to represent the sets of sections, waterlines and buttocks. Each group, added in turn, interpolates all previous definition using curve fitting. While it is possible to set this up within the existing software, it performed poorly. The curves could actively update when changes to the definition were made, but referenced control points would often swap position, creating loops in the curves, Fig.10. It is impractical to resolve these instances manually due to the large number of curves and it would be far more practical for the software to perform sorting automatically. In addition, the update was slow due to the fact that each curve was updated individually. This highlighted that the design definition curves provided by software are not particularly suited to this activity because they are optimised for human use.

Fig. 10: A lofted hull form produced manually by adding individual curves to create a process which fits sections, then buttocks to boundary curves and waterlines (thick lines). Without sorting, loops often form, right, which need to be resolved manually. Automating the fitting process resolves this and allows task to be optimised to generate large numbers of curves at once.

The solution required the curve generation technique to be executed as a software process where algorithms could optimally perform the intersection and fitting as a block of activity. This would have the benefit that the process could be parametrically configured allowing the order of fitting of sections, waterlines and buttocks to be changed and to allow the order to be different in the forward end of the hull form to the aft end. At this point, it should be highlighted that by automating this process in a closed algorithm the flexibility offered by AVEVA Lines, i.e. allowing the user the ability to intersect and fit any curve at in any order, is no longer available. But automating this process is very valuable in early design

4.2. Automating the Curve Lofting Process.

The automated process is quite simple and fits sets of planar curves (sections, waterlines and buttocks) in an order of the user’s choosing and that order may be different in the forward and aft portions of the hull form. The user specifies the position of each planar cut to use. For each planar cut, the following process is used:

1. Intersect definition curves and any previously generated curves with the current planar cut. Surface tangent attributes associated with definition curves are also captured.

2. Sort points with respect to their sequence in space. This involves looking for “nearest neighbours” rather than sorting in any particular coordinate direction.

3. As intersection points can contain tangent information and typical Cubic B-Spline algorithms usually only account for tangent information at the end of curves, the intersection points are subdivided into groups so that points with tangents appear at start and end of a fit sequence.

4. Fit a cubic B-Spline, *Piegl and Tiller (1996)*, through each group of points and assemble together as a multi-segment B-Spline curve. Some surface tangent attributes, where the tangent direction at the end of one group is dependent on the end condition applied an adjacent group, require the order in which each group is fitted to be adaptable.
It is possible to assemble these multi-segment curves into a single B-Spline curve by merging control points and knot vector. However, as this reduced the performance of the intersection process, because there is more data, this is not performed. The intersection process is the most critical in terms of time. Two stages are used to find all intersections between a curve and plane. The first discretises the curve into a set of points and derivatives based on the knot vector sequence. This data is then searched to find any pairs of points which neighbour a potential intersection given that there may be more than one along the curve. A Newton-Raphson search is then used to find each point of intersection within a certain tolerance, bracketed by the curve position from first stage points. This process is simple when compared with the complexity of generating a surface definition. There are few if any cases where particular arrangements of geometry required specific processing to produce the complete surface representation. Furthermore, implementation was achieved in a matter of days/weeks compared to the years of development that may be required to produce a successful surface generation technique.

Fig. 11: Hull form entirely represented by curves. Design Curves are drawn in thick lines. The entrance is fitted using waterlines, sections, then buttocks and the run, buttocks, waterlines, then sections. The generation process took 0.096 s on an average laptop (2010).

4.3. Curve Lofting vs. Multi-Patch Surface Generation: User Experience

As this curve lofting technique used the same definition curves as the author’s surface generation technique it was possible to experience each solution on the same input data. The difference between the two techniques from the user perspective was stark. With the surface generation technique, for a typical ship hull form, it takes about 1 s to generate an entrance or run surface (they are normally defined separately to improve performance). Subsequently, its necessary to generate the surface contours and this is dependent on the number of contour intersections defined by the user, but typically it might take a further second. In contrast, the curve lofting technique would produce the same set of curve intersections within 0.2 s (the author’s preferred minimum threshold for interactive surface manipulation) and include the presentation of section curvature, something that requires additional processing for surfaces. While the time required for surface generation is not considerable, its performance is outside what can be tolerated by the user if they require the surface contours to update as they move a control point with the mouse. The user loses the mental image of the previous state and the user is unable establish a feedback relationship between moving the mouse and the changes appearing on screen. Performance is degraded. However, the speed of update with the curve lofting technique is fast enough to allow contours to be updated as the mouse is moved with performance being similar to that experienced by users manipulating individual B-Spline surfaces when designing small craft.

This improvement in performance allows the user to investigate and improve areas of the surface definition which previously may have been regarded as acceptable. For example, cutting surface contours very near to the flat of side and flat of bottom planes often highlight areas of unfairness. This area is difficult to improve because the shape is dependent on the curvature of definition curves intersecting with these boundaries. The position and tangency is already constrained by attributes on the flat of side/bottom curves. In addition, the use of degenerate patches (square patches with a zero length edge) along the flat of side, something which often happens in definitions where the designer is
not being too fastidious about patch quality, often results in poor shape and unfairness. The rapid performance of the curve lofting allows subtle changes in control points to be made and the improvements. Consequently, Curve lofting can be used as a way of rapidly improving definition curves and, correspondingly, improve the quality of a generated surface which is dependent on the same definition curves.

4.4. Combining Curve Lofting and Multi-Patch Surface Generation: User Experience

Surfaces have become so important for transferring shape between different design software and between the different stakeholders in the ship design process that promoting hull form definitions based on curves would be seen as a retrograde step. However, the previous section discusses how the benefits of using Curve Lofting can transfer into a surface based on the same definition. Previously, it has been highlighted that when generating the patches of a complex surface there is a degree of estimation in the generation of tangent ribbons, subdividing n-sided faces and in the blending algorithms used to generate patches and that sometime the results are different to the designer’s expectation. However, it can be difficult to decisively identify when a surface is not behaving correctly because often the effects are quite subtle. From the designer’s perspective: “The surface somehow doesn’t look right, but I can’t identify why!” Using Curve Lofting and Surface generation based on the same definition allows the user to identify problematic areas based on where the contours produced by each technique diverge. The location and scale of divergent areas provides the designer clear guidance on where to focus their attention and on how close they are to achieving a satisfactory definition. Working with two techniques offers the user some confidence of not adversely affecting the (design) definition as they seek to improve it as its possible to add curves to one technique leaving the other alone.

![Check Curve Family](image.png)

Fig. 12: Overlaying the surface and the lofted curves generated by the same definition visually highlights areas that need checking or correcting. Local bumps and twists are often a good indicator that there is change in the way the curves are connected altering the curve family, see 4.5.

4.5. Working with Curve Families

A drawback of working with a highly interactive surface design tool based on the cross sectional design approach is that while it is very easy to add, remove and change the control points on definition curves, it can lead to variations in the number of points on neighbouring curves and inconsistency in the relationships used to connect curves together. If neighbouring curves are defined consistently, i.e. form a family of curves, Fig.13, a smooth surface definition results with very little effort. From the user’s perspective it is desirable to have the flexibility to have different numbers of points and constraints, however, it causes inconsistencies in the way curves are parameterised and this can leads to subtle unfairness in the surface patches. While it is difficult to spot this phenomenon visually in the patches at any particular instant, because surface blending algorithms accommodate variations in parameterisation, it becomes noticeable when fairing if it becomes difficult to achieve a smooth surface by moving control points alone. This can often be caused by inconsistencies in the way neighbouring curves are defined and can only be improved by changing the number or points or relationships between curves.
Fig. 13: Curve families occur when neighbouring curves are generated with the same number and types of input. A smooth “surface” shape results. The transitions between curve families can be sharp and result in a significant change of curvature. It is advantageous to minimise the number of curve family changes.

This is a situation that is difficult to identify and is something that can only be done with experience. However, the Curve Lofting approach was found to readily identify the worse inconsistencies between curves, the cases where there were variations in relationships (master-child). As Curve lofting generates many curves through intersection planes in one direction and then another, changes in consistency result in sharp localised disturbances in curvature, effects which would be averaged out in the case of a patch blending algorithm. These are obvious to the user when displaying curvature and require investigation, Fig.14.

Once the user has an appreciation how variations in the definition affect the fairness of a surface they include that knowledge into the design process when creating a new surface. Unfortunately, this does suggest that a regular structure is used in construction of the cross section design definition curves which goes against its benefits of allowing arbitrary topology. But experience suggests that a regular and well organised definition structure produces a higher quality surface.

Fig. 14: Loft curves highlight changes in Curve Families in the curvature analysis. In this case caused by a design curve, a yacht midship section, that is not connected consistently with other curves. In this case it is corrected by removing the design curve from the lofting curve generation process (right)

4.6. Unstructured and Arbitrary Definition

One of the limitations of working with surfaces is the need to work with structured definition whether that is a regular mesh or arbitrary connected curves. The user must therefore spend some time organising or establishing the correct structure. This is a divergence from the design process. An advantage working with Curve Lofting is that the definition is not required to have any structure whatsoever as the composition is entirely generated by intersections and the connectivity of curves is based on the relative position of intersection points. Definition curves do not need to be connected and can take any shape. In practice, the user will probably start with a structured set of boundary curves
but beyond that curves controlling shape can be established using more artistic consideration. Consequently, it can take far less curves to develop the same shape when compared with structured approaches. Furthermore, unstructured definition curves cannot be readily used to generate a surface although the fitting techniques discussed later may be used to establish a structured definition.

Fig. 15: Curve lofting does not require the use of a connected mesh of design curves. Consequently, the designer is free to experiment with space curves and often a good form can be produced with minimal curves.

4.7. Integrating Curve Lofting and Multi-Patch Surface Generation

A limitation of using tangent ribbons and patch blending to generate multi-patch surfaces is that the process which generates the patch itself is restricted to the position and tangent information available in adjacent edges, which is primarily estimated. A significant proportion of the blending algorithms effort is focus towards achieving twist compatibility. Consequently, there is minimal focus on shape in the centre of a patch and the quality of shape across the wider patch neighbourhood. Curve lofting overcomes these limitations because the fitting process is applied across the whole surface and is only broken by breaks in curvature due to tangents in the surface and knuckle lines. Why not combine the two techniques where the shape of each patch is based on the curves that cross it? There is a variety of ways that this could be achieved. Rhim et al. (2005) suggests a method using virtual iso-parameters, curves generated from the curves that cross each patch, to which the patch is subsequently fitted. However, a method that introduces further geometrical elements often brings a need for additional estimation, in this case by using Bessel end conditions on the iso-parameter curves. AVEVA Lines has pioneered the use of a least squares method where Uniform B-Spline patches are fitted by sampling the curves representing patch edges and crossing each patch face. The method makes no guarantee of continuity except in the position of adjacent patch edges. However, experience suggests that trained users are able to produce a surface of higher quality more quickly than when using methods that enforce tangential continuity. In fact, the software allows the possibility to add tangential continuity constraints to each patch although this often results in poorer quality surfaces. One may conclude from this that while this technique may not enforce tangential or curvature constraint between neighbouring patches, the continuity of the resulting surface, checked using visual presentations such as Isophotes and Gaussian or mean curvature is more than adequate for ship production, Fig.16.

Fig. 16: A comparison of Gaussian curvature between the surface generated by blending (left) and generated by fitting to the lofted curves (right). General curvature continuity is improved although there are other areas that have picked up unfairness in the loft curves. The same definition curves are used in both cases.
Compared with other techniques, the least squares method offers a simple solution to surface fitting. A significant benefit of the sampling process is that patches are reparameterised with uniform knot vectors and means that the control points of neighbour patches match geometrically. This has great benefits at it ensures that there are no gaps between patches (a requirement for solid modelling and gridding), reduces degradation of the accuracy of patches when exchanged between CAD systems using text files and allows robust detailed fairing at the control point level of patches because it is possible to manually fair across patch edges. However, as the surface fairness is entirely dependent on the curves it is down the user to achieve this and as mentioned previously, in AVEVA Lines, this requires a good degree of discipline as it is at this point where short cuts taken during the definition process are exposed in the quality of the surface and the time required to rectify them.

An interesting comparison is possible at the point between the method used in AVEVA Lines and in the author’s “automatic” method discussed in 4.2. The automatic method enforces discipline in the order curves are generated. It therefore manages the definition of the curves and does not expose any control points of generated curves to the user. AVEVA Lines on the other hand exposes all control points and does not manage the order in which curves are fitted. This allows a skilled user to make small adjustments to control points as long as they ensure correspondence between any intersecting curves. To achieve this in an automated method can require more definition curves and loft curves. This may not change the complexity of the generated surface but it does impact on processing time and visual clarity on screen if there are too many curves.

Combining information from the multi-patch surface generation technique and curve lofting provides the opportunity to improve many of the deficiencies which occur when position and tangent information has to be interpolated around the definition. A particular area of poor performance is in the subdivision of non-four sided faces where the automatically generated vertices and edges always fair with respect to the rest of the surface. The shape and position of these elements can be again improved using least square fitting with respect to the curve lofting data.

However, the least square fit of patches has been found to perform poorly in two areas. By avoiding constraints, explicit surface tangents along patch edges may not exactly follow the definition and may acquire noise from the numerical solution process, particularly if an iterative solver is used. Another area is that there may be many areas of the surface where there is not enough or in some case no intersecting curves from which sample points to fit the surface can be generated. This occurs in the very small patches that get generated next to particular features such as along a keel or on an end surface. In these cases, patch blending performs much better. The least squares method performs poorly when there are not enough supporting sample points to support a patch and the user will experience this when control points are generated far from the surface definition. It is possible to mitigate this by introducing smoothing, Farin (2002), into the least squares fit. This allows sensible patches to be created in these difficult areas. However, introducing smoothing as a global parameter across all patches reduces curvature continuity and devising an interrogation process which functions at patch level which can selectively introduce smoothing is difficult because it is necessary to determine a reliable test. Neither is it desirable to expose a smoothing parameter for each patch to the user.

The author is experimenting with an alternative approach which combines the patch blending method and least squares based on the benefits of using an iterative solver in this case conjugate gradient. In a situation where the control points of a patch is poorly supported by the sample data, an iterative solver will tend to produce more stable results whereas a direct solver may, through numerical instabilities, cause control points to positioned far from the surface. Iterative solvers require the control points to be initialised and perform better if they are initialised close to the solution. By first fitting a patch to data sampled from the blended result, it is then possible to improve the shape of the patch based on sample points taken from the loft curves. In patches where there is not enough sample data, control points that are not supported by sample data are not moved and in cases where there is no sample data this fit is not performed.
5. Conclusion

This investigation uncovers that curve lofting can play an important role when working with a surface definition. It provides the capability to rapidly evaluate a hull surface definition when using cross sectional design curves and highlights areas of unfairness caused by inconsistencies in position and curve reference information that would often be smoothed beyond visual recognition by a generated surface. Implemented in a modern CAD environment, it can be used to generate a high quality hull definition far faster than a design tool based on surfaces alone and used to explore areas of surface fairness that can be time consuming to address in a surface definition. The fact that the lofting result is entirely governed by relative position of intersections with other curves, the designer is longer concerned with ensuring the definition is specified correctly for the tool. Consequently, surface design becomes a liberating experience where the designer can focus on exploring the shape.

When used alongside a surface definition, the curve lofting can be used to improve the surface definition either as a result of sharing the same design curve definition, or the surface representation itself can be improved by surface fitting patches to loft curves rather than using blending techniques like Coon’s patch. Once a satisfactory surface is achieved, the curve lofting information can be discarded.

Acknowledgements

I would like to acknowledge the assistance of my colleague Ian Applegarth for extending my knowledge of mathematical techniques that such as those used in AVEVA Lines.

References


BOLE, M. (2010), Interactive hull form transformations using curve network deformation, COMPIT 2010, Berlin


CHIYOKURA, H. (1988), Solid Modelling with DESIGNBASE: Theory and Implementation, Addison-Wesley


KOELMAN, H. J. (1999), *computer support of design, engineering, and prototyping of the shape of ships hulls*, Ph.D. Thesis, Delft University of Technology


