Ugail H (2006): "Method of Trimming PDE Surfaces" *Computers & Graphics*, 30 (2): 225-232.

Method of Trimming PDE Surfaces

Hassan Ugail

School of Informatics
University of Bradford
Bradford BD7 1DP, UK
h.ugail@bradford.ac.uk

Abstract

A method for trimming surfaces generated as solutions to Partial Differential Equations (PDEs) is presented. The work we present here utilises the 2D parameter space on which the trim curves are defined whose projection on the parametrically represented PDE surface is then trimmed out. To do this we define the trim curves to be a set of boundary conditions which enable us to solve a low order elliptic PDE on the parameter space. The chosen elliptic PDE is solved analytically, even in the case of a very general complex trim, allowing the design process to be carried out interactively in real time. To demonstrate the capability for this technique we discuss a series of examples where trimmed PDE surfaces may be applicable.

Key words: PDE Surfaces, Trimming, Laplace Equation
C & G Keywods: Modeling , Boundary representations, Curve and surface representations, Numerical Algorithms and Problems

1 Introduction

Surface trimming can be broadly described as a process whereby a given surface patch is mathematically re-defined subject to the condition that a given unwanted portion of the original surface be removed. This process involves the generation of new surface geometry for the trimmed patch by means of closely approximating the corresponding portion of the surface on the original patch.

Surface trimming is considered to be a key design component of any free-form surface modelling system [5,8]. Trimming, for instance, is an essential part of Boolean operations commonly used in complex surface modelling. A simple such operation which is commonly performed involves generating a hole on the surface for shape blending purposes.

Preprint submitted to Elsevier Science *19 October 2005*

There exist a wide variety of well-established techniques for free-form surface design for which surface trimming is an essential component. Such techniques include boundary based methods such as polygon based design [12], extrusions and surface of revolution [20], polynomial patches [8]; procedural modelling such as implicit surfaces [2]; and volumetric models such as constructive solid geometry [21] and subdivision [6]. Among these free-form surface design techniques the most widely used are polynomial patches and subdivision.

In the case of polynomial surface patches the problem of trimming was most probably discussed by Hoscheck [13–15] at first. These early methods involved the division of the parameter space of a given polynomial surface patch into a set of subsets whose specific forms are defined by cubic Bézier surface patches. The definition of the Bézier surfaces in this case involved a combination of knot insertion and change of basis functions. Applegarth, around the same time, proposed an approach to trim spline surfaces by using a method based on clipping trim curves [1]. Here the work of Rojas is also notable. Roja's methods involved the use of Fleming's constraint-based inversion algorithm to define a new set of B-spline surface patches which make up the corresponding regions of interest in the original surface [9] . More recent work on trimming of spline surfaces, especially for NURBS, involve the identification of a trim region within the domain whose evaluation is skipped by computing the pre-image of the trim curve in the parametric domain [17,18,18]. It is noteworthy that generally speaking the computations involved in trimming spline surfaces demand high computational resources.

Subdivision surfaces, which have emerged to be a popular alternative, have been considered as a way of overcoming topological limitations of spline surfaces. Due to the underlying formulations of subdivision surfaces it is often not necessary to carry out trimming operations during the design process. A notable piece of work on trimming subdivision surfaces has been discussed by Litke [19]. This method is based on the use of combined subdivision schemes which guarantee exact interpolation of trim curves. The latter, for example, ensures that if two surfaces share a trim curve, they will meet exactly at the trim curve. Unlike the trimming operations for spline surfaces, a new control mesh is generated for each trim operation and the multiresolution details are utilised to correct any unnecessary perturbation of the surface near the trim during this multistage process. In order to enhance the computational performance, a novel set of quasi-interpolation operators are utilised. For a detailed discussions on this the interested reader may wish to consult [19] in which a number of trimming examples for subdivisions are also given.

The work discussed in this paper is on the trimming of surfaces based on the so called PDE method for surface design introduced by Bloor and Wilson [3]. The method adopted for design here is somewhat different from other methods in that a boundary-value approach is adopted whereby a surface is characterised by defining a number of space curves with associated derivative information, so as to form the surface's edges, and then the surface is generated between these curves by solving an elliptic partial differential equation (PDE).

The PDE method was originally developed as a mechanism of blend shape generation. Since this initial application of the method on shape blending, in recent years, PDE based shapes have broadened their uses in shape description, e.g. [16,7,22]. This is due to the fact that PDE shapes in principle have the advantage that most of the information defining a shape is based on a boundary representation making it possible for complex geometry to be generated and controlled by very few design parameters [24].

The aim of this paper is to present a methodology by which complex trimming operations on PDE surfaces can be carried efficiently within an interactive environment. Thus, we show how a trim curve which is defined on the parameter space defining a PDE surface can be utilised to formulate a lower order boundary value problem. This boundary-value problem, which in this case is based on the well known Laplace equation, is then solved within the parameter space enabling geometric meshing of the trimmed surface to be performed efficiently. An analytic solution procedure is outlined which enables the boundary-value problem to be solved over trim parameter space fast enough to be able to design trimmed PDE surfaces interactively in real time.

The paper is organised as follows. Section (2) presents the outline of the PDE method within the context of free-form surface design and discusses how the chosen PDE is solved subject to a given set of function boundary conditions. The discussions in Section (3) focus on the main point of the paper where the methodology for trimming PDE surfaces is discussed. Section (4) then presents a set of examples describing how the method can be utilised to carry out a wide variety of trimming operations on free-form PDE surfaces.

2 The PDE Method for Surface Design

The PDE method produces a parametric surface, defined by two parameters u and v over a finite region $\Omega \subset R^2$, as a solution to a suitably chosen partial differential equation (PDE). The chosen PDE is solved subject to a set of boundary conditions which are usually defined at the edges of the surface patch. The PDE method has been discussed before in a number of different references, e.g. [3,22,7]. It has been shown how surfaces satisfying a wide range of functional requirements can be created by a suitable choice of the boundary conditions and appropriate values for the various design parameters associated with the method [23,24].

For the work described here, and for the majority of previous work carried out using the PDE method, the PDE chosen is of the form,

$$\left(\frac{\partial^2}{\partial u^2}+a^2\frac{\partial^2}{\partial v^2}\right)^2 \underline{X}(u,v)=0, \tag{1}$$

where $\underline{X}(u,v)$ are the points of the surface in the physical space. The conditions on the function $\underline{X}(u,v)$ and its normal derivatives $\frac{\partial X}{\partial n}$ can be imposed at the edges of the surface patch. Thus, in broad terms, one could think that the PDE method generates a <u>smooth</u> surface patch by solving an elliptic PDE such as Equation (1) subject to a set of boundary conditions that are imposed at the edges of the surface patch.

The parameter a is a special design parameter [3]. Essentially the parameter a produces a 'waist' effect within the interior of the surface patch where the higher the value of 'a' the more waist it produces. This technique has been utilised in an interactive setting to control the interior of the surface as discussed in detail in [22].

It is noteworthy that the PDE method has several desirable features for surface design. These include the ability to be able to define and control the shape of the surface through a set of boundary data, the ability to generate smooth surfaces and the ease by which shape parameterisation can be incorporated enabling functional design to be carried out.

2.1 Outline of the PDE Solution Method

There exist many methods to determine the solution of Equation (1) ranging from analytic solution techniques to sophisticated numerical methods. For the work described here restricting to periodic boundary conditions a closed form analytic solution of Equation (1) is utilised.

Choosing the parametric region to be $0 \leq u \leq 1$ and $0 \leq v \leq 2\pi$, for the work discussed in this paper, we take the periodic boundary conditions to be of the form,

$$\underline{X}(0,v)=\underline{P}_0(v), \tag{2}$$

$$\underline{X}(s,v)=\underline{P}_1(v), \tag{3}$$

$$\underline{X}(t,v)=\underline{P}_2(v), \tag{4}$$

$$\underline{X}(1,v)=\underline{P}_3(v), \tag{5}$$

where $0 < s, t < 1$.

Note that the boundary conditions $\underline{P}_0(v)$ and $\underline{P}_1(v)$ define the edges of the surface patch at $u = 0$ and $u = 1$ respectively while the surface patch will also pass through the curves $\underline{P}_1(v)$ and $\underline{P}_2(v)$ at $u = s$ and $u = t$.

Using the method of separation of variables, the analytic solution of Equation (1) can be written as,

$$\underline{X}(u, v) = \underline{A}_0(u) + \sum_{n=1}^{\infty}[\underline{A}_n(u)\cos(nv) + \underline{B}_n(u)\sin(nv)], \tag{6}$$

where

$$\underline{A}_0 = \underline{a}_{00} + \underline{a}_{01}u + \underline{a}_{02}u^2 + \underline{a}_{03}u^3, \tag{7}$$

$$\underline{A}_n = \underline{a}_{n1}e^{anu} + \underline{a}_{n2}ue^{anu} + \underline{a}_{n3}e^{-anu} + \underline{a}_{n4}ue^{-anu}, \tag{8}$$

$$\underline{B}_n = \underline{b}_{n1}e^{anu} + \underline{b}_{n2}ue^{anu} + \underline{b}_{n3}e^{-anu} + \underline{b}_{n4}ue^{-anu}, \tag{9}$$

where $\underline{a}_{00}, \underline{a}_{01}, \underline{a}_{02}, \underline{a}_{03}$ $\underline{a}_{n1}$, $\underline{a}_{n2}$, $\underline{a}_{n3}$, $\underline{a}_{n4}$, $\underline{b}_{n1}$ $\underline{b}_{n2}$, $\underline{b}_{n3}$ and $\underline{b}_{n4}$ are vector constants, whose values are determined by the imposed boundary conditions at $u = 0$, $u = s$, $u = t$ and $u = 1$.

For a general set of boundary conditions, in order to define the various constants in the solution, it is necessary to Fourier analyse the boundary conditions and identify the various Fourier coefficients. For certain types of boundary conditions when the boundary conditions can be expressed exactly in terms of a finite Fourier series, the solution given in Equation (6) will also be finite. In general, however, it is often not possible to get a finite Fourier series representation of a given set of boundary conditions. In such cases the solution will be the infinite series given Equation (6).

The technique for finding an approximation to $\underline{X}(u, v)$ is based on the sum of the first few Fourier modes and a 'remainder term', i.e.,

$$\underline{X}(u, v) \simeq \underline{F}(u, v) + \underline{R}(u, v), \tag{10}$$

where

$$\underline{F}(u, v) = \underline{A}_0(u) + \sum_{n=1}^{N}[\underline{A}_n(u)\cos(nv) + \underline{B}_n(u)\sin(nv)]. \tag{11}$$

Here N is a small integer value, usually taken to be 6.

The function $\underline{R}(u, v)$ is a remainder function defined as,

$$\underline{R}(u, v) = \underline{r}_1(v)e^{wu} + \underline{r}_2(v)ue^{wu} + \underline{r}_3(v)e^{-wu} + \underline{r}_4(v)ue^{-wu}, \tag{12}$$

where $\underline{r}_1$, $\underline{r}_2$, $\underline{r}_3$, $\underline{r}_4$ and w are obtained by considering the difference between the original boundary conditions and the boundary conditions satisfied by the function $\underline{F}(u, v)$.

The point about the above solution method is that although the solution is approximate the method guarantees that the chosen boundary conditions are exactly satisfied since the remainder function $\underline{R}(u, v)$ is calculated by means of the difference between the original boundary conditions and the boundary conditions satisfied by the function $\underline{F}(u, v)$ [4].

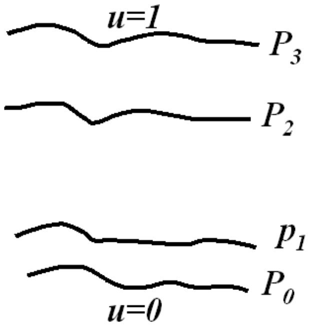


Fig. 1. A set of boundary conditions for the PDE defined in terms of curves in 3-space.

Figure 1. shows a series of curves defined in 3-space which can be utilised to generate a four-sided PDE surface patch. As described above here the curves P_0, P_1, P_2 and P_3 will all be contained in the resulting surface patch.

Since the curves P_0, P_1, P_2 and P_3 are not periodic the parametric region for which the PDE solved here is taken to be $0 \leq u \leq 1$ and $0 \leq v \leq \pi/4$.

Figure 2 shows the PDE surface patch generated using the boundary conditions described in Figure 1 whereby the above described solution method has been utilised. The value of a in this case has been taken to be 1.1. Note that the shape of the surface patch can easily be controlled by the shape of the boundary curves, in particular the boundary curves P_0, P_1, P_2 and P_3.

An important point one should bear in mind here is that due to the nature of the boundary conditions taken here the method for surface generation is somewhat different from that originally proposed in [3]. In [3] and other previous work on the PDE method the boundary conditions are usually defined using a set of functions and derivatives defined at edges $u = 0$ and $u = 1$ of the surface patch. In this work, however, we take a set of four function conditions which enables the surface to satisfy exactly those conditions and hence the surface is

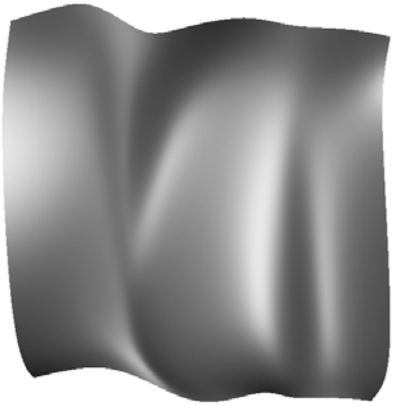


Fig. 2. The PDE surface patch generated using the boundary conditions shown in Figure 1.

a smooth interpolation between the chosen boundary conditions. One should also note that the creases that appear on the surface patch shown in Figure 2 are due to the shape of the boundary curves chosen and are not due to the chosen PDE nor due to the solution method utilised. In particular, one can see that there are sharp edges on the curves marked P_1 and P_2 which result in the interior creases as seen on the surface patch.

The above example demonstrates how a typical PDE surface patch can be generated. For the remainder of this paper we use the above example to discuss how PDE surfaces can be trimmed. Thus, we show how complex trimming can be carried out within the interior of the above the surface patch. It is noteworthy that although we show how trimming can be carried out using a single example, it is clear that one could appreciate that the trimming techniques discussed below can be applied to any PDE surface or for that matter any parametric surface patch.

3 Method of Trimming

In this section we discuss the main contribution of this work whereby we show how a given complex shape can be trimmed out from a PDE surface. For the purpose of demonstration we assume that the trim on the PDE patch can be defined using the a corresponding trim curve defined on the (u, v) parameter space. Given a trim curve in the (u, v) parameter space, the main task in the process of trimming involves the determination of the interior region of the trim curve whereby all the mesh points belonging to the region is then discarded.

In previous work, described in [22], a simple procedure involving transfinite interpolation has been successfully utilised to perform meshing of the (u, v)

parameter space after performing a trim. There are several issues with this procedure. For example, for complex regions where the trim curve is non-convex, the meshing process has to be carried out in stages whereby the trim curve has to be re-defined as a series of sub curves all of which are convex. Furthermore, for complex trims involving curves which are not necessarily connected to each other, this procedure is found to be not robust enough.

One could argue that there exist other well known robust triangulation procedures that could be utilised to overcome the above problem. One such well known technique is Delaunay triangulation [10] which is capable of meshing complex regions. Experience, however, shows that in the particular case of PDE surfaces the use of Delaunay triangulation to mesh the trimmed (u, v) parameter introduces more problems than it solves. For example, Delaunay triangulation produces triangular meshes which are not necessarily uniform and therefore prohibits utilising numerical solution techniques such as the Finite Difference methods to solve the PDEs. Furthermore, for complex regions the speed of existing Delaunay triangulation algorithms may prohibit them using for real time shape manipulation work.

Here we show a fast method that can be utilised to perform complex meshing on trim regions of (u, v). The method is based on solutions to the Laplace equation involving the trim curves over the (u, v) parameter space as discussed below.

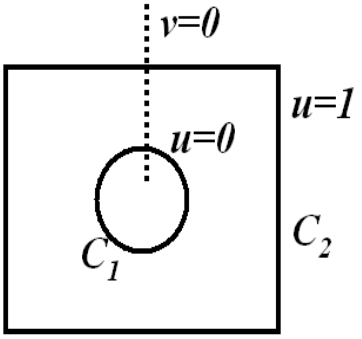


Fig. 3. A trim curve defined on the parametric domain. The boundary conditions defined by C_1 and C_2 are utilised to solve the Laplace equation within the parametric domain.

3.1 Fast Solution Method for Laplace Equation

Consider Figure 3. showing the parametric region where the circular curve marked as C_1 is a trim curve. In order to define a valid mesh within the interior of the parametric region which discard the trim region, we solve the Laplace equation subject to a set of two boundary conditions. Assuming the region in which meshing is to be performed can be mapped to a finitely discretised region belonging to $0 \leq u \leq 1$ and $0 \leq v \leq 2\pi$ we assume the mesh $\underline{M}(u,v)$ to be the solution of the Laplace equation,

$$\left(\frac{\partial^2}{\partial u^2} + \frac{\partial^2}{\partial v^2}\right)\underline{M}(u,v) = 0. \tag{13}$$

The boundary conditions for the Equation (13) is taken to be the curves C_1 and C_2 where C_2 is the square border curve. Note that both curves C_1 and C_2 are periodic with $v = 0$ being at the positions marked by the dotted line in Figure 3. Thus, one could imagine that by solving Equation (13) between the region defined by the curves C_1 and C_2 we can obtain a smooth transition between the two curves which will enable us to obtain the required mesh.

In order to carry out the trimming fast enough to be able to generate trimmed geometry in real time we utilise analytic solution scheme where the above defined periodic boundary curves are utilised. This solution scheme is in fact very similar to the analytic solution scheme for the Biharmonic Equation previously discussed. Thus, for Equation (13) the periodic boundary conditions can be expressed as, $\underline{M}(0,v) = C_1(v)$, $\underline{M}(1,v) = C_2(v)$. Now using the method of separation of variables similar to that used to solve the Equation (1) we can write the solution of Equation (13) as a Fourier series such that,

$$\underline{M}(u,v) = \underline{A}_0(u) + \sum_{n=1}^{\infty}[\underline{A}_n(u)\cos(nv) + \underline{B}_n(u)\sin(nv)], \tag{14}$$

where

$$\underline{A}_0 = \underline{a}_{00} + \underline{a}_{01}u, \tag{15}$$

$$\underline{A}_n = \underline{a}_{n1}e^{anu} + \underline{a}_{n2}ue^{-anu}, \tag{16}$$

$$\underline{B}_n = \underline{b}_{n1}e^{anu} + \underline{b}_{n2}ue^{-anu}, \tag{17}$$

where $\underline{a}_{00}, \underline{a}_{01}$, $\underline{a}_{n1}$, $\underline{a}_{n2}$, $\underline{b}_{n1}$ $\underline{b}_{n2}$, are vector constants, whose values are determined by the imposed boundary conditions at $u = 0$ and $u = 1$.

Again in cases where trim curves are simpler the boundary conditions can be expressed exactly in terms of a finite Fourier series. In such cases the solution given in Equation (14) will also be finite. However, for complex trim curves this is often not possible, in which case the solution will be the infinite series given in Equation (14). For the purpose of meshing the interior region of the trimmed surface an approximate solution of Equation (13) is obtained which satisfy exactly at the given boundary conditions.

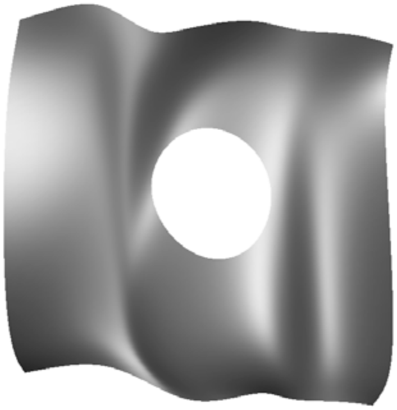


Fig. 4. The trimmed PDE surface patch generated using the trim region shown Figure 3.

This approximate solution of Equation (13) is given as,

$$\underline{M}(u,v) \simeq \underline{A}_0(u) + \underline{F}(u,v) + \underline{R}(u,v), \tag{18}$$

where

$$\underline{F}(u,v) = \underline{A}_0(u) + \sum_{n=1}^{N} [\underline{A}_n(u)\cos(nv) + \underline{B}_n(u)\sin(nv)], \tag{19}$$

The specific form of the function $\underline{R}(u,v)$ used here is,

$$\underline{R}(u,v) = \underline{r}_1(v)e^{wu} + \underline{r}_2(v)e^{-wu}. \tag{20}$$

where $\underline{r}_1$, $\underline{r}_2$, and w are obtained by considering the difference between the original boundary conditions and the boundary conditions satisfied by the function $\underline{F}(u,v)$.

Figure 4. shows the PDE surface discussed originally where the circular portion has been trimmed out using the method described. Thus, Equation (13) is solved over the trim region subject to the boundary conditions defined by the curves C_1 and C_2 shown in Figure 3 to generate a uniform mesh $\underline{M}$ which is then utilised to generate the PDE surface with the trim.

3.2 Comparison with Delaunay triangulations

In order to test the speed and efficiency of the our proposed method we compared it with a commonly utilised algorithm for surface meshing namely the

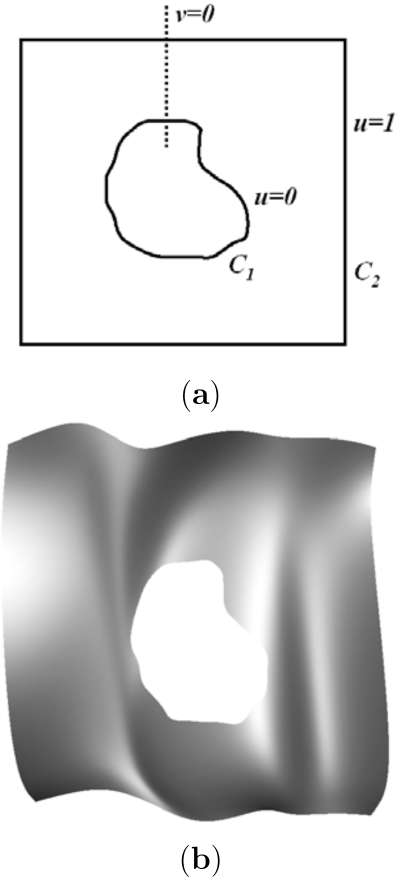
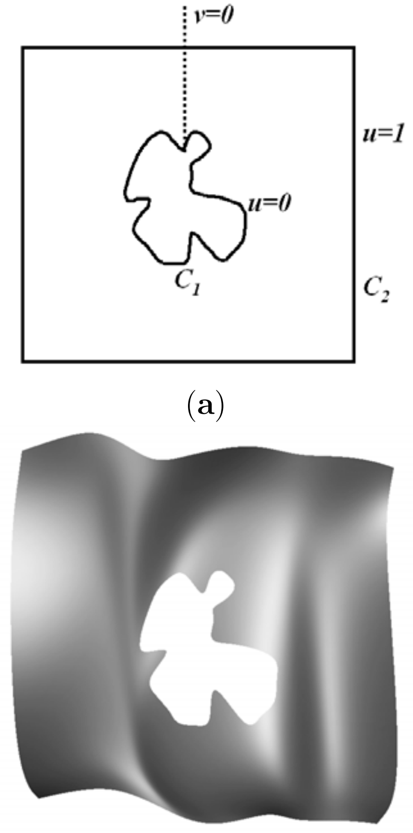


Fig. 5. Example of interactive trimming. (a) Trim curve defined by means of a cubic Spline. (b) The corresponding trimmed PDE surface patch.

Delaunay triangulation. The Delaunay triangulation essentially involves creating from the sample points a set of non-overlapping triangularly bounded facets, where the vertices of the triangles are the input sample points. There are a number of Delaunay triangulation algorithms available. For the purpose of our test we utilised that of Fortune [11]. To undertake the comparison we have essentially compared the time it took for our proposed method and the Delaunay triangulation to produce an interior mesh for the trim region shown in Figure 3. Table 1 shows the results of the time in milliseconds (ms) for different mesh resolutions. The computations were performed on a fast Windows based laptop computer. As one can see from these results our proposed method in all cases is more efficient in producing the mesh when compared to the Delaunay triangulation.



(**a**)

(**b**)

Fig. 6. Example of an interactive trim manipulation. (a) Trim curve defined by means of a cubic spline starting from that shown in Figure 5(a). (b) The corresponding trimmed PDE surface patch.

Mesh (u, v)	Laplace (time in ms)	Delauney (time in ms)
10×20	0.0313333	0.1391000
20×30	0.1561000	0.3333000
40×70	0.3912000	0.8900000
140×170	4.2265000	7.10128000

Table 1
Comparison of the proposed method with Delaunay triangulations. Time in milliseconds to generate the mesh.

4 Examples of Trimming

In this section we further discuss the method of trimming described previously. In particular, we discuss a series of examples which demonstrate how the method can be applied to perform complex trims with ease.

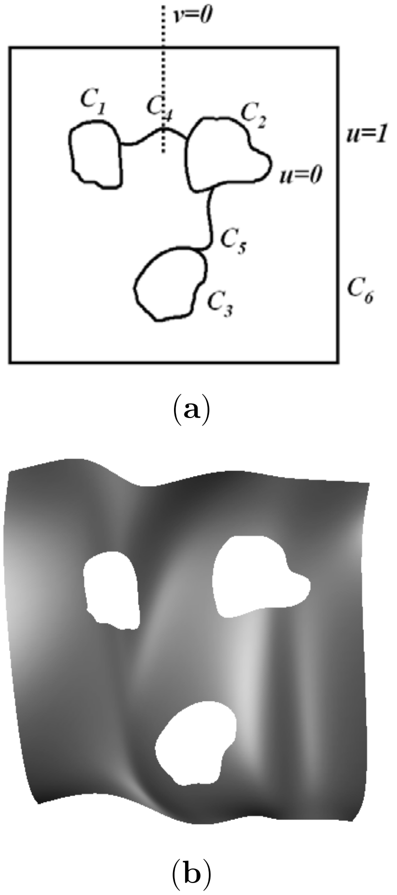


Fig. 7. Example of trimming where more than one disjoint curve is present. (a) Disjoint trim curves defined by means of cubic splines. (b) The corresponding trimmed PDE surface patch.

As a first example we discuss the trimming problem shown in Figure 5. Here a general shape of a curve is interactively drawn on the (u, v) parameter space and the corresponding shape on the surface is removed. The curve here is generated as a cubic B-spline where its shape is adjusted using the corresponding control points. Note that both the curves C_1 and C_2 are periodic with $v = 0$ being at the positions marked by the dotted line in Figure 5(a). Once the desired shape of the curve is decided, Equation (13) is solved subject to the boundary conditions represented by C_1 and C_2 as shown in Figure 5(a).

As a second example we show the trimming problem shown in Figure 6 where the trim curve is shown in Figure 6(a) is generated by means of interactively manipulating the spline shown in Figure 5 (a). Note that the trim curve here is non-convex. The curves C_1 and C_2 are periodic with $v = 0$ being at the positions marked by the dotted line in Figure 6(a). With these settings the trim surface is generated by means of solving Equation (13) subject to the boundary conditions represented by C_1 and C_2 as shown in Figure 6(a).

Figure 7. shows an example of a trim where multiple trim curves which are disjoint are involved. For such type of problems the disjoint curves are first connected so as it forms a single periodic curve in the (u, v) parameter space. In the case the curve segment C_4 is used to join the curves C_1 and C_2 while the curve segment C_5 is utilised to join the curves C_2 and C_3. Thus, by this mechanism a single periodic curve can be easily generated in a situation where a set of disjoint curves are given.

Figure 7(b) shows the resulting surface after the trim. Like the previous examples, the position corresponding to $v = 0$ on the two boundary conditions can be identified by dotted line in Figure 7(a). The trim surface is then generated by solving the Equation (13) subject to the boundary conditions represented by C_1 and C_2 shown in Figure 7(a).

5 Conclusions

The aim of this paper is to describe a method for trimming surfaces generated as solutions to PDEs. The work presented here utilises the parametric region over which the PDE surface is originally defined. The trim curves are defined on the parametric domain and the projection of these curves on the parametrically represented PDE surface is then trimmed out. To do this we define the trim curves to be a set of boundary conditions which enable us to solve the standard Laplace equation in the parameter domain. Here the Laplace equation is solved analytically, even in the case of a very general complex trim, allowing the design process to be carried out interactively in real time. To demonstrate the capability of this technique we have discussed a series of examples where trimmed PDE surfaces may be applicable.

An important point one should note here is that the method presented here, although very applicable to PDE surface, is not necessary limited to just PDE surfaces. The method can be equally utilised to trim other types of parametric surfaces. Furthermore, it can be utilised as a fast meshing technique where surface meshing is called for.

An interesting new direction of study stemming from this work would be to look into the solutions of other elliptic PDEs especially those of higher orders on the parameter space. This would increase the scope of re-defining the boundary conditions for complex trimming operations in more convenient ways. Often in situations, such as for design analysis, one requires convenient and efficient ways of generating adaptive meshes. By utilising higher order equation the scope of mesh adaptivity within the interior trim regions can also be enhanced.

6 Acknowledgements

The author wishes to acknowledge the financial support received from the UK Engineering and Physical Sciences Research Council grants EP/C015118/1 and EP/D000017/1 through which this work was completed.

References

[1] I. Applegarth, D. Catley and I. Bradley. Clipping of B-spline patches at surface Curves, *The Mathematics of Surfaces III*, Clarendon Press, Oxford, 229-242, 1997.

[2] C. Bajaj, J. Blinn, J. Bloomenthal, M. Cani-Gascuel, A. Rockwood, B. Wyvill and G. Wyvill. *Introduction to Implicit Surfaces*, Morgan-Kaufmann, 1997.

[3] M. I. G. Bloor and M. J. Wilson. Using partial differential equations to generate freeform surfaces, *Computer Aided Design*, 22:202-212, 1990.

[4] M. I. G Bloor and M. J. Wilson. Spectral approximations to PDE surfaces, *Computer Aided Design*, 28:145-152, 1996.

[5] M. S. Casale. Free-form solid modeling with trimmed surface patches, *IEEE Computer Graphics & Applications*, 7(1):33-43, 1987.

[6] T. DeRose, M. Kass and T. Truong. Subdivision surfaces in character animation, In *Proceedings of SIGGRAPH 98*. Addison Wesley, 85-94. 1998.

[7] H. Du and H. Qin. A shape design system using volumetric implicit PDEs, *Computer Aided Design*, 36(11):1101-1116, 2004.

[8] G. Farin. *Curves and surfaces for computer aided geometric design: a practical guide*, Academic Press Inc., 1992.

[9] S. Fleming, A. Myklebust. The Enhancement of PHIGS Plus B-spline Functionality for Geometric Modeling in CAD, *Fourth IFIP WG5.2 on Geometric Modeling in Computer Aided Design*, New York, 1992.

[10] S. Fortune. A Sweepline Algorithm for Voronoi Diagram, *Algorithmica*, 2:153-174, 1987.

[11] S. Fortune. Voronoi diagrams and Delaunay triangulations , in *Euclidean Geometry and Computers*, M. J. D.A. Du, F. K. Hwang (eds.), World Scientific Publishing Co., 193-233, 1992.

[12] J. Hoschek, D. Lasser. *Computer Aided Geometric Design*, A K Peters, 1993.

[13] J. Hoscheck and F. Schneider. Spline Conversion for Trimmed Rational Bezier and Bspline Surfaces, *Computer Aided Design*, 22(9):580-590, 1990.

[14] J. Hoscheck. Intrinsic Parameterization for Approximation, *Computer Aided Geometric Design*, 5:27-31, 1988.

[15] J. Hoscheck. Approximate Conversion of Spline Curves, *Computer Aided Geometric Design*, 4:59-66, 1987.

[16] J. Huband, and W. Li. Extracting design parameters from airplane wing data by using bloor-wilson pde surface model, *Mathematical Engineering in Industry*, 8:239-252. 2001.

[17] J. Keyser, S. Krishnan and D. Manocha. Efficient and accurate b-rep generation of low degree sculptured solids using exact arithmetic: II-computation, *Computer Aided Geometric Design*, 16(9):861-882, 1999.

[18] S. Krishnan and D. Manocha. An efficient surface intersection algorithm based on lower dimensional Formulation, *ACM Transactions on Graphics*, 16(1):74-106, 1997.

[19] N. Litke, A. Levin and P. Schröder. Trimming for subdivision surfaces, *Computer Aided Geometric Design*, 18:463-481, 2001.

[20] M. E. Mortenson. *Geometric Modelling*, Wiley-Interscience, New York, 1985.

[21] A. Rappoport and S. Spitz. Interactive boolean operations for conceptual design of 3-d solids. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. ACM, New York, 269-278, 1997.

[22] H. Ugail, M. I. G Bloor, and M. J. Wilson. Techniques for interactive design using the PDE method, *ACM Transactions on Graphics*, 18(2):195-212, 1999.

[23] H. Ugail. On the spine of a PDE surface, in *Mathematics of Surfaces X*, M. J. Wilson and R. R. Martin (eds.), Springer, 366-376, 2003 .

[24] H. Ugail and M. J. Wilson. Efficient shape parametrisation for automatic design optimisation using a partial differential equation formulation, *Computers and Structures*, 81(29):2601-2609, 2003.