Parametric patches and volumes in function representation of geometric solids

K.T. Miura A.A. Pasko V.V. Savchenko

Abstract

We take an approach to specify and control implicitly defined n-dimensional free-form solid primitives using parametric functions of n variables. Outside of the domain, the parametric function value increases or decreases rapidly that is not suitable to design geometric objects.

In this paper, we introduce a functional clipping operation which allows a free-form primitive to be treated as traditional implicit primitives in set-theoretic modelling based on R-functions. We also introduce an extended Bézier clipping operation for fast inverse mapping in the case of the changed domain. Examples of set-theoretic, offsetting, hypertexturing and metamorphosis operations on free-form primitives are given. Tessellation of trimmed parametric patches is explained as an application of our new geometric modelling method.

Introduction

The parametric representation of curves and surfaces is the most common one in geometric modelling and computer graphics. One of its main advantages is that the parametric representation leads to a design paradigm based on a control-point array. On the other hand, a representation of curves and surfaces as zero sets of real functions of several variables (so-called implicit or function representation) have proven to be useful in computer-aided design, computer graphics and animation. This representation is closed under important geometric operations such as offsetting and blending. However, it is more difficult to control implicitly defined objects than parametric ones. Usually, the set of primitive implicit shapes is restricted by quadrics, tori, superquadrics and skeleton-based models.

The combination of two representational styles is quite promising. In this paper we use an approach to specify and control implicitly defined n-dimensional free-form solid primitives using parametric functions of n variables. This approach has already been applied to construct piecewise algebraic surfaces [9, 6, 2] and thick free-form shells [1]. In contrast, we wish to model solid primitives (not surface patches or shells) and to use them in functionally based constructive modelling. We have found two major problems:

1. The domain of each parameter t of these parametric functions is usually limited(for example, $0 \leq t \leq 1$). Outside of the domain, the function value increases or decreases rapidly. It does not have negative values everywhere outside the domain that leads to generation of undesirable solids. We propose to apply functional clipping to change behaviour of the function outside the domain.

2. Changing the domain of patches and volumes by moving control points is a powerful modeling tool. A regular rectangular grid of control points can become an irregular one. If control points have been moved in the plane (space) of function arguments, one has to apply inverse mapping of a given point to evaluate the defining function. The Bézier clipping method [3] does not work for points outside the domain and is too slow for inside points. We propose here extended Bézier clipping to deal with these problems.

The application of R-functions [7, 10] makes set-theoretic operations closed on the function representation. It means that a complex constructive object can be described by a single real function. Although to evaluate such a function is time consuming, this approach gives a basis for quite general solutions to blending, offsetting, hypertexturing, and collision detection problems [5]. We introduce functional clipping and extended Bézier clipping operations which allow a free-form primitive to be treated as traditional implicit primitives in set-theoretic modelling based on R-functions. Examples of set-theoretic, offsetting, hypertexturing and metamorphosis operations on free-form primitives are given. Tessellation of trimmed parametric patches is explained as an application example.

Function representation with parametric patches and volumes

Function representation of geometric solids

We consider closed subsets of n-dimensional Euclidean space E^n with the definition:

$$f(x_1, x_2, ..., x_n) \geq 0 \tag{1}$$

where f is a real continuous function on E^n. We call f a defining function. The defining inequality (1) we call the *function representation* (F-rep) of a geometric object. The function can be defined in various ways: analytically, with an evaluation algorithm, or with tabulated values and an appropriate interpolation procedure. The major requirement of the function is to have at least C^0 continuity. Equation (1) allows the definition of closed n-dimensional objects in E^n space with the following characteristics:

$$f(\boldsymbol{X}) > 0 \quad \text{- for points inside the object:} \tag{2}$$

$$f(\boldsymbol{X}) = 0 \quad \text{- for points on the object boundary:} \tag{3}$$

$$f(\boldsymbol{X}) < 0 \quad \text{- for points outside the object,} \tag{4}$$

where $\boldsymbol{X} = (x_1, x_2, ..., x_n)$ is a point in E^n.

Parametric patches and volumes

Several types of parametrically defined free-form patches have been proposed to define 3-dimensional objects with complicated sculptured surfaces. A point on a patch is given by

$$(x, y, z) = \mathbf{S}(u, v). \tag{5}$$

Bézier patches and NURBS(non-uniform B-spline) surface are popular because of their nice properties for geometric calculations. In this paper, we use integral Bézier patches and volumes as illustrative examples for simplicity. However, it will not cause any theoretical difficulties when we apply our method to any other parametric functions like rational Bézier patches, NURBS surfaces, or Gregory-type patches to represent geometric objects.

A Bézier patch $\mathbf{S}(u, v)$ is defined as follows:

$$\mathbf{S}(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} B_i^m(u) B_j^n(v) \mathbf{P}_{ij} \tag{6}$$

where $\mathbf{P}_{ij}$ are 3-dimensional points called control points. $B_i^n (i = 0, ..., n)$ are Bernstein basis functions defined by

$$B_i^n(t) = \binom{n}{i}(1-t)^{n-i}t^i. \tag{7}$$

Similarly, a Bézier volume is defined as follows:

$$\mathbf{S}(u,v,w) = \sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{k=0}^{n} B_i^l(u)B_j^m(v)B_k^n(w)\mathbf{P}_{ijk} \tag{8}$$

where $\mathbf{P}_{ijk}$ are 4-dimensional control points in this case. An extension of this representation scheme to n-dimension is straightforward by use of n parameters with (n+1)-dimensional control points. By changing positions of the control points, we can modify the shape of the geometric objects according to our intension.

How to define geometric solids

Here, as the first step, we define a 2D solid within a unit square region on the x-y plane. Of course, the function $f(x,y)$ used in the function representation should be defined in the whole 2D space. Hence, we will introduce a new technique called functional clipping to make $f(x,y)$ to have negative values outside of the unit square domain. Then we extend it to be defined within more general regions. To define a 2D solid by a function representation $f(x,y) \geq 0$ in the domain $\{(x,y) | 0 \leq x, y \leq 1\}$, we assume that the x and y coordinates of $\mathbf{S}(u,v)$, $\mathbf{S}^x(u,v)$ and $\mathbf{S}^y(u,v)$ are

$$\mathbf{S}^x(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} B_i^m(u)B_j^n(v)\mathbf{P}_{ij}^x \tag{9}$$

$$= u, \tag{10}$$

$$\mathbf{S}^y(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} B_i^m(u)B_j^n(v)\mathbf{P}_{ij}^y \tag{11}$$

$$= v, \tag{12}$$

i.e.

$$\mathbf{P}_{ij}^x = \frac{i}{m}, \quad \mathbf{P}_{ij}^y = \frac{j}{n}. \tag{13}$$

Then we can use the z coordinate of $\mathbf{S}(u,v)$, $\mathbf{S}^z(u,v)$ to define a 2D solid as follows:

$$f(x,y) = \mathbf{S}^z(x,y) = \mathbf{S}^z(u,v). \tag{14}$$

To define a 3D solid by a function representation $f(x,y,z) \geq 0$ in the domain $\{(x,y,z) | 0 \leq x,y,z \leq 1\}$, we assume that the x,y, and z coordinates of $\mathbf{S}(u,v,w)$, $\mathbf{S}^x(u,v,w)$, $\mathbf{S}^y(u,v,w)$, and $\mathbf{S}^z(u,v,w)$ are

$$\begin{aligned}
\mathbf{S}^x(u,v,w) &= \sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{k=0}^{n} B_i^l(u) B_j^m(v) B_k^n(w) \mathbf{P}^x_{ijk} && (15)\\
&= u, && (16)\\
\mathbf{S}^y(u,v,w) &= \sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{k=0}^{n} B_i^l(u) B_j^m(v) B_k^n(w) \mathbf{P}^y_{ijk} && (17)\\
&= v, && (18)\\
\mathbf{S}^z(u,v,w) &= \sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{k=0}^{n} B_i^m(u) B_j^m(v) B_k^n(w) \mathbf{P}^z_{ijk} && (19)\\
&= w, && (20)
\end{aligned}$$

i.e.

$$\mathbf{P}^x_{ijk} = \frac{i}{l}, \quad \mathbf{P}^y_{ijk} = \frac{j}{m}, \quad \mathbf{P}^z_{ijk} = \frac{k}{n}. \tag{21}$$

Then we can use the ω(4th) coordinate of $\mathbf{S}(u,v)$, $\mathbf{S}^z(u,v)$ to define a 3D solid as follows:

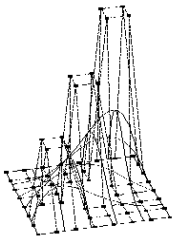
$$f(x,y,z) = \mathbf{S}^\omega(x,y,z) = \mathbf{S}^\omega(u,v,w). \tag{22}$$

Figure 1(a) shows a Bézier patch of degree 7×7 and Figure 1(b) is the boundary (contour map $f = 0$) of the 2D solid defined by the patch in the unit square domain. Figures 1(c, d) and 1(e, f) are also examples which reveal the expression versatility of the method being discussed.

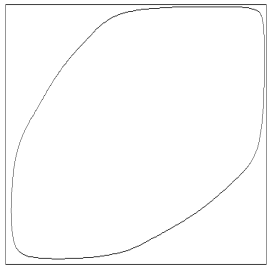
Clipping patches and volumes

Functional clipping

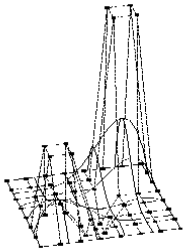
Figures 2(a) and (b) show the surface and the contour map of a Bézier patch on the rectangle bigger than the unit square. The



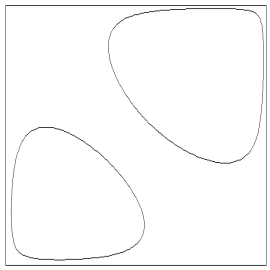
(a) Bèzier patch



(b) Contour map

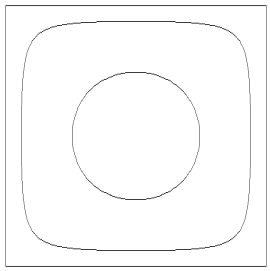


(c) Bézier patch



(d) Contour map: disjoint components

(e) Bézier patch



(f) Contour map: 2D torus

Figure 1. 2D solid examples

patch was constructed to produce only one implicit closed curve in the unit square. As it can be observed, the behaviour of the parametric function outside the unit square contradicts the requirement of negative values of a defining function for points outside a defined object. It results in undesirable zero contours outside the unit square.

One could suggest to use the unit square (or the unit cube in the 3D case) as a bounding volume to be checked by any application procedures. However, it is not applicable to a general transformed domain. On the other hand, it is quite attractive to deal with considered "parametrically defined implicits" and with traditional ones in the same manner. To keep the uniformity that is one of the advantages of the function representation, we introduce the following functional clipping procedure.

Let $\mathbf{S}(u,v)$ is a function defining a parametric patch and $S(u,v) = \mathbf{S}^z(u,v)$. Consider the intersection of the unit square and the implicit primitive defined by this function:

$$S_{clip}(u,v) = S(u,v) \& F_s(u,v), \tag{23}$$

where $F_s(u,v) = F_b(u) \& F_b(v)$ is a defining function of the unit square with F_b defining the unit strip by each variable:

$$F_b(t) = (1-t)t \tag{24}$$

and & is the symbol of R-intersection defined as:

$$g_1 \& g_2 = g_1 + g_2 - \sqrt{g_1^2 + g_2^2}. \tag{25}$$

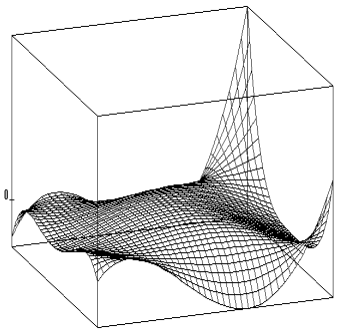
The application of this procedure is illustrated by Figure 2(c) and (d). The defining function is negative outside the unit square with the desirable implicit curve being unchanged. The same procedure has to be applied in 3D case:

$$S_{clip}(u,v,w) = S(u,v,w) \& F_c(u,v,w), \tag{26}$$

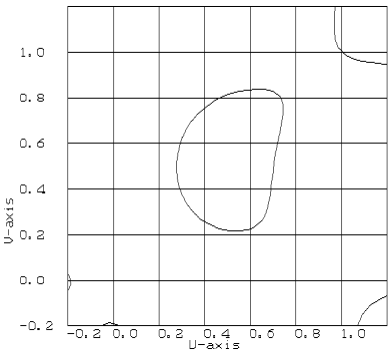
with the unit cube defined as

$$F_c = F_b(u) \& F_b(v) \& F_b(w). \tag{27}$$

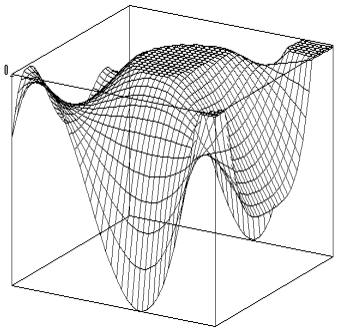
In our implementation the procedures defining parametric patches and volumes encapsulate the introduced functional clipping. It allows us to apply any operation suitable for traditional implicits to free-form primitives.



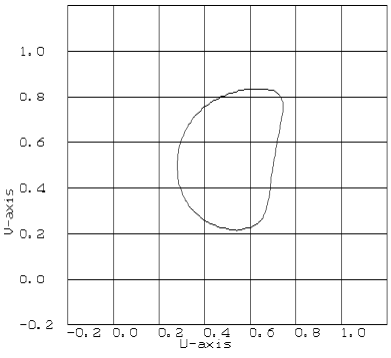
(a) Original Bézier patch



(b) Contour map $f = 0$ of (a)



(c) Clipped Bézier patch



(d) Contour map $f = 0$ of (c)

Figure 2. Functional clipping operation

When the domain of the patch or the volume is transformed to a general region, at first, we map (x, y) to (u, v) or (x, y, z) to (u, v, w) and perform the functional clipping for a unit square as we explained above. Then we calculate the function value of $S_{clip}(u, v)$ or $S_{clip}(u, v, w)$ as the defining function.

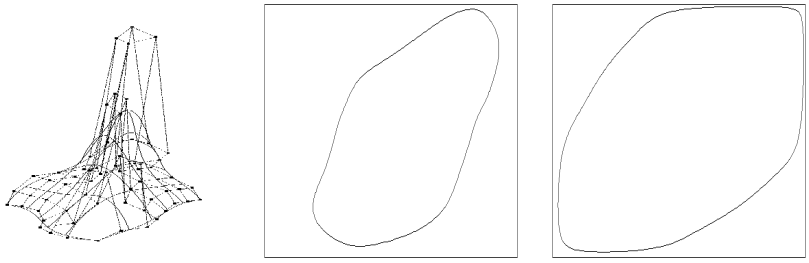
Extended Bézier clipping

Here, using the 2D solid as an example, we will mention the change of the domain of a function $f(x, y)$ defined by a parametric function $\mathbf{S}(u, v)$. The change of the domain of $f(x, y)$ means that although the domain of $\mathbf{S}(u, v)$ remains a unit square, we will change the range of $(x(u, v), y(u, v))$ from a unit square to some other region. The extension of our method to higher dimensional cases is straightforward. Although the domain becomes complicated, it is easy to perform functional clipping operations in the parameter space after mapping from (x, y) to (u, v).

We can apply several standard techniques of geometric operations like translation, rotation, scaling, reflection, shear and others to transform the unit square to some other region. These transformations can be expressed by using a 3×3 matrix $\mathbf{N}$ with homogeneous coordinates. It is necessary to obtain (u, v) values to calculate $f(x(u, v), y(u, v))$. However, a simple multiplication with an inverse matrix $\mathbf{N}^{-1}$ is enough to get a (u, v) value for a given point (x_0, y_0).

Since for x and y coordinates of each control points, arbitrary values can be specified, we can define a generally shaped domain bounded by four Bézier curves. Figure 3 shows an example of a general transformation of the domain explained above. Each control point is translated without changing its z value from the original patch. It may be interpreted as one more modification method instead of changing z values without changing x or y.

In order to get values of parameters (u, v) at a given point (x_0, y_0), we have to calculate an intersection point between a patch $\mathbf{S}(u, v)$ and a straight line parallel to the z axis passing through a point $(x_0, y_0, 0)$. Several methods are available to calculate intersections between a Bézier patch and a straight line like the Bézier clipping method proposed by Nishita et al.[3]. The problem with using this method is that it does not provide any mapping for points outside the domain. For points inside the domain it works too slow. We propose extended Bézier clipping here that is a combination of ap-



(a) A Bézier patch with moved control points

(b) Contour map $f = 0$

(c) Contour map $f = 0$ of the original Bézier patch shown for comparison

Figure 3. General domain case

proximate inverse mapping and Bézier clipping. The procedure is as follows:

1) Apply approximate inverse mapping to the given point (x_0, y_0). This mapping uses the scattered data interpolation by the volume spline based on the Green's function [8]. In the 2D case it is well-known as the thin plate spline. We apply this spline to interpolate control points displacements and to calculate displacement values for any given point.

2) If the resulting point (u, v) of the previous step lies inside the unit square, define a bounding box for it and apply Bézier clipping.

This procedure provides inverse mapping for any point and is faster than pure Bézier clipping for the same given accuracy as shown in Table 1.

Modelling with Parametric Patches and Volumes

We have modeled a sample free-form primitive shown in Figure 4(a) using a tri-quadratic Bézier volume with 27 control points and functional clipping. In the following subsections we illustrate several

Algorithm	Tolerance of parameters	
	10^{-6}	10^{-10}
Extended Bézier clipping(using 16 points)	3.49 sec	4.11 sec
(using 25 points)	3.57 sec	4.18 sec
(using 36 points)	3.67 sec	4.27 sec
Pure Bézier clipping	4.84 sec	5.71 sec

Table 1. Comparison of processing time with 900 points for the patch shown in Figure 3(a) by Indy MIPS R4600 64Mbyte.

operations on this primitive. Note that each operation results in a new continuous real function of three variables that provides closure of the representation. The halftone images were produced using ray-marching algorithm with the step along the ray defined by the user. This algorithm is quite slow. To get the interactive speed of visualization we apply a parallel version of our polygonization algorithm [4].

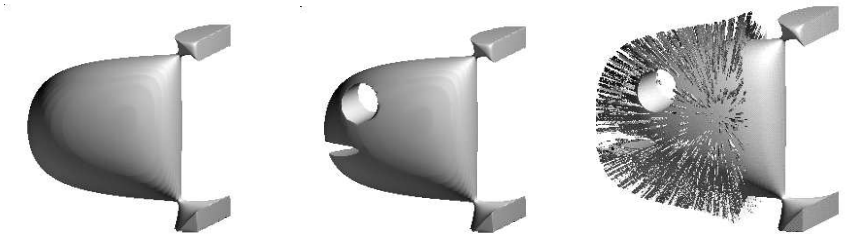
Set-theoretic operations

As we mentioned above, after functional clipping is applied a free-form primitive has a defining function suitable to set-theoretic operations using R-functions. Figure 4(b) shows the result of set-theoretic operations on the free-form primitive. Details on these and other operations can be found in [5, 10]. Note that objects resulting from set-theoretic operations with R-functions are not guaranteed to be regular solids. They can have dangling low-dimensional portions and their defining functions can have "internal zeroes". In some practical applications non-regular objects are required. If regularity of the objects is required, one can apply a special method of constructing defining functions [10].

Offsetting and hypertexturing

Figure 4(c) illustrates offsetting and hypertexturing operations. Thin needles are generated with solid noise and then are cut by the offset object [5]. We apply here an offset operation described as

$$f_2(x,y,z) = f_1(x,y,z) + d(x,y,z), \tag{28}$$



(a) Solid defined by a Bézier volume

(b) Fish-shaped solid

(c) Hypertextured solid

Figure 4. Set operations and hypertexturing

where f_1 is a defining function of the initial object, d is a positive continuous displacement function. In the simplest case $d(x,y,z) =$ const defines iso-valued offsetting.

Metamorphosis

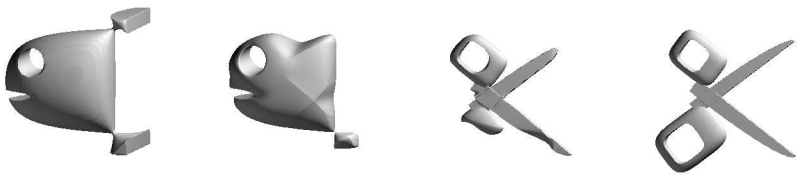
Figure 5 shows several phases of metamorphosis between two functionally defined objects with defining functions $f_1(x,y,z)$ and $f_2(x,y,z)$. The intermediate object is described as

$$f_3(x,y,z,t) = (1-t)f_1(x,y,z)h_1(x,y,z,t) + tf_2(x,y,z)h_2(x,y,z,t), \quad (29)$$

where parameter t varies from 0 to 1, h_1 and h_2 are positive continuous modulation functions. In the simplest case $h_1 = h_2 = 1$. In the given example f_1 defines an object constructed with the free-form primitive shown in Figure 4(b) and f_2 defines an object constructed with traditional implicit primitives: quadrics and superquadrics shown in Figure 5(d)

Application: Tessellation of Trimmed Patches

Trimmed surface patches have a fundamental role in computer-aided geometric design. We consider here tessellation of trimmed parametric patches as an application example. A trimmed surface patch is defined by



(a) Phase 1 (t=0.0) (b) Phase 2 (t=0.13) (c) Phase 3 (t=0.8) (d) Phase 4 (t=1.0)

Figure 5. Metamorphosis

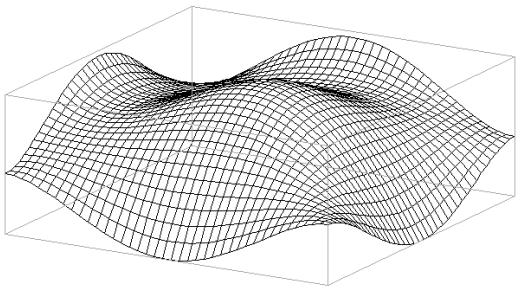
1) parametric patch (in our case a Bézier patch shown in Figure 6(a)),

2) a set of trimming curves lying within the parameter space of the patch (Figure 6(b)).

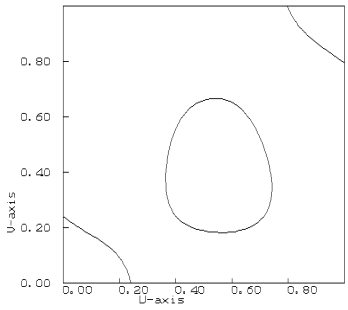
We define trimming curves as a zero set of another Bézier patch. It is quite natural to think of trimming as the set-theoretic difference between the parameter rectangle and the areas bounded by the trimming curves. This operation results in an implicitly defined 2D rectangle with holes. Tessellation of this object using some implicit surface tiler produces 2D polygons. Then one can construct the trimmed Bézier patch by calculating its values in the vertices of obtained 2D polygons. The tessellated Bézier patch is shown in Figure 6(c).

Conclusions

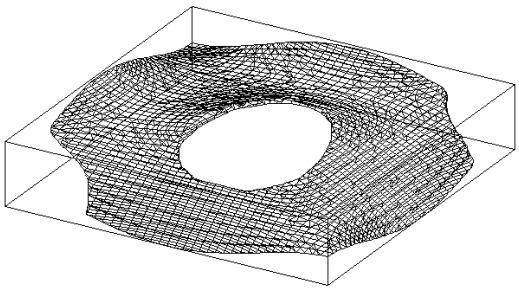
We have developed a new technique of function representation of geometric solids using parametrically defined free-form patch and volume formulations. Even though the function representation for geometric solids is widely used, it has a difficulty of modifying implicitly defined objects. In order to overcome it, we use parametrically defined patches and volumes with the functional clipping and extended Bézier clipping operations proposed in this paper. We can treat a free-form primitive as traditional implicit ones in set-theoretic modelling based on R-functions. We have explained basic



(a) Original Bézier patch



(b) Trimming curves in the parameter space



(c) Trimmed Bézier patch

Figure 6. Tessellation of trimmed patch

treatments of parametric functions to represent and model geometric solids implicitly.

References

[1] J.P. Menon, "Constructive shell representations for free-form surfaces and solids", *IEEE Computer Graphics and Applications* **14**, 2(24-27), February 1994.

[2] A.E. Middleditch and E. Dimas, "Solid models with piecewise algebraic free-form faces", CSG94 Set-theoretic Solid Modelling: Techniques and Applications, Information Geometers, Winchester, (133-148), 1994.

[3] T. Nishita, T.W. Sederberg and M. Kakimoto, "Ray tracing trimmed rational surfaces patches", *Computer Graphics* **24**, 3(337-345), August 1990.

[4] A.A. Pasko, V.V. Pilyugin and V.V. Pokrovskiy, "Geometric modeling in the analysis of trivariate functions", *Computers and Graphics* **12**, 3/4(457-465), 1988.

[5] A. Pasko, V. Adzhiev, A. Sourin and V. Savchenko, "Function representation in geometric modeling: concepts, implementation and applications", *Visual Computer* **11**, 8(429-446), 1995.

[6] N.M. Patricalakis and G.A. Kriezis, "Piecewise continuous algebraic surfaces in terms of B-splines", Geometric Modeling for Product Engineering, M.J.Wozny et al. (Eds.), Elsevier Science Publishers, (3-19) 1990.

[7] V.L. Rvachev, *Methods of logic algebra in mathematical physics*, Naukova Dumka Publishers, Kiev, Ukraine, 1974.

[8] V.V. Savchenko, A.A. Pasko, O.G. Okunev and T.L. Kunii, "Function representation of solids reconstructed from scattered surface points and contours", *Computer Graphics Forum* **14**, 4(181-188), October 1995.

[9] T.W. Sederberg, "Piecewise algebraic surface patches", *Computer Aided Geometric Design* **2**, (53-59), 1985.

[10] V. Shapiro, "Real functions for representation of rigid solids", *Computer Aided Geometric Design* **11**, 2(152-175), 1994.