The Visual Computer

Function representation in geometric modeling: concepts, implementation and applications

A. Pasko[1,*], V. Adzhiev[2,**], A. Sourin[3,***], V. Savchenko[1,****]

[1] Shape Modeling Laboratory, The University of Aizu, Tsuruga, Ikki-Machi, Aizu-Wakamatsu City, Fukushima, 965 Japan
[2] Department of Computer Science, University of Warwick, Coventry, CV4 7AL, UK
[3] School of Applied Science, Division of Software Systems, Nanyang Technological University, Nanyang Avenue, Singapore 2263

Concepts of functionally based geometric modeling including sets of objects, operations, and relations are discussed. Transformations of a defining real function are described for set-theoretic operations, blending, offsetting, bijective mapping, projection, cartesian products, and metamorphosis. Inclusion, point membership, and intersection relations are also described. We use a high-level geometric language that can extend the interactive modeling system by input symbolic descriptions of primitives, operations, and predicates. This approach supports combinations of representational styles, including constructive geometry, sweeping, soft objects, voxel-based objects, deformable and other animated objects. Application examples of aesthetic design, collisions simulation, NC machining, range data processing, and 3D texture generation are given.

Key words: Geometric modeling – Solid modeling – Real functions – Implicit surfaces – R functions

* e-mail: pasko@u-aizu.ac.jp
** e-mail: valery@dcs.warwick.ac.uk
*** e-mail: assourin@ntuvax.ntu.ac.sg
**** e-mail: savchen@u-aizu.ac.jp
Correspondence to: A. Pasko

1 Introduction

The use of real functions of several variables for defining geometric objects is quite common in mathematics and computer science. The inequality $f(x_1, x_2, \ldots, x_n) \geq 0$ describes a half space in n-dimensional euclidean space. The equation $f(x_1, x_2, \ldots, x_n) = 0$ specifies an implicit function of $n-1$ variables and describes an orientable $(n-1)$-dimensional surface. The properties of these geometric objects are studied in algebraic and differential geometry. In the 3D case, an object defined by the inequality mentioned is usually called a solid (or a volume), and an object defined by the equation is called an implicit surface. Functionally represented volumes and surfaces appear to be useful in solid modeling, computer aided geometric design (CAGD), animation, range data processing, and volume graphics.

Half spaces defined by algebraic inequalities are used as primitives in constructive solid geometry (CSG) (Requicha 1980, Requicha and Rossignac 1992). Duff (1992) solves such important problems as collision detection and rendering on the basis of interval arithmetic for CSG trees with primitives bounded by implicit surfaces. The representation of a whole complex object by a single real function has also attracted interest. An attempt to develop a system of set-theoretic operations based on this representation has been made by Ricci (1973). A serious restriction of the proposed method is that there is C^1 discontinuity in the domain of the functions defining complex objects. The exact analytical definitions of the set-theoretic operations have been proposed in the theory of R functions (Rvachev 1963, 1974) and applied to solving problems of mathematical physics on complicated geometric domains (see Shapiro 1988, 1994 for a survey). Shapiro (1994) uses R functions to construct defining functions of regular sets required in CSG. The theory of R functions was applied to define several operations on multidimensional geometric objects in Pasko (1988) and Pasko et al. (1993a).

Wang (1984) pointed out the theoretical possibility of deriving an implicit description of the surface swept by a moving solid. Symbolic computations were required to reduce the dimension of variables and to yield the representation in an implicit form.

Although parametric representation is the most common in CAGD, attention is also paid to implicit surfaces because of their closure under some

important operations (Hoffmann 1993). Typical operations of this kind are the offsetting and blending (Ricci 1973; Middleditch and Sears 1985; Hoffmann and Hopcroft 1987; Hoffmann 1989; Rockwood 1989; Warren 1989).
Sclaroff and Pentland (1991) have generalized the implicit function representation by introducing a deformation defined with a matrix of free vibrations. Such a generalization provides modeling deformations based on physical laws and collision detection for animated objects (Essa et al. 1992). Collision detection for implicit surfaces is also discussed by Baraff (1990), Gascuel (1993) and Snyder et al. (1993).
Several implicit functions have been proposed on the basis of a field generating skeleton (Blinn 1982; Wyvill et al. 1986a; Bloomenthal and Shoemake 1991). The field defined by superquadrics (Barr 1981) can increase a complexity of the resultant object significantly. These functions are useful in interactive modeling (Bloomenthal and Wyvill 1990) and animation (Wyvill et al. 1986b). Muraki (1991) has applied Blinn's blobby model to approximate the shape of the object defined by sample surface points. Model deformations and displacement maps (Sclaroff and Pentland 1991), as well as polynomial functions (Bajaj et al. 1993), have also been applied for solving the surface reconstruction problem.
Many authors have presented algorithms of polygonization of implicitly defined surfaces or isosurfaces of trivariate functions (Wyvill et al. 1986a, Bloomenthal 1988, Pasko et al. 1988, Schmidt 1993). A similar algorithm was proposed by Lorensen and Cline (1987) for extracting polygonal surfaces from volumetric data. In fact, objects with implicit surfaces and voxel-based objects have unified models. The only difference is that a tabulated function of three variables is used for voxel-based objects. For example, Hughes (1992) used such a viewpoint to implement a metamorphosis between two voxelized solids.
Thus, function representation is widely used in geometric modeling and computer graphics in several forms. However, these models are not closely related to each other or to such well-known representations as sweeping, boundary representation (B-rep), and CSG (Requicha 1980). This obviously retards further research. The motivation of our work is the need to fill these gaps by constructing as rich as possible a system of operations dealing with functionally represented objects. Descriptions that are independent of dimension lead to the possibility of including 4D and other multidimensional volumes in the set of objects treated. The application of real functions of several variables provides a good base for an interactive modeling system that can be extended by symbolic descriptions of primitives and operations.
This paper presents a state-of-the-art report of our project, the main objectives of which are:

- Categorization and summary of the geometric concepts required in a functionally based modeling environment
- Elaboration of a rich system of geometric operations closed on functionally represented objects
- Treatment of multidimensional and particularly space-time objects in a uniform manner
- Specification, implementation, and application of an interactive geometric modeling system based on function representation

In Sect. 2, we consider geometric concepts as a triple (objects, operations, relations) with their descriptions in terms of real functions of several variables and present new descriptions of some geometric operations. Section 3 deals with interactive geometric modeling based on function representation. Section 4 is devoted to application examples. Section 5 summarizes the paper and discusses future work.

2 Geometric concepts and function representation

This section provides a summary of geometric concepts of the functionally based modeling environment. We also discuss new descriptions of several geometric operations that we have recently developed.
Let us describe the geometric concepts as a triple (M, Φ, W) where M is a set of geometric objects, Φ is a set of geometric operations, and W is a set of relations for the set of objects. Mathematically, this triple is an algebraic system. Its application to multidimensional and time-dependent geometric modeling have been studied by Pilyugin et al. (1988), Pasko (1988) and Sourin (1988). We now consider the main parts of this triple.

2.1 Objects

We consider geometric objects as closed subsets of n-dimensional euclidean space E^n with the definition

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

where f is a real continuous function defined on E^n. We call f a *defining function*. We call Eq. 1 a *function representation* (or *F-rep*) of a geometric object. In the 3D case, the boundary of such an object is a so-called "implicit surface." Note that the definition of an object is Eq. 1 with the explicit function of n variables $f = f(x_1, x_2, \ldots, x_n)$, but not the implicit function of $n-1$ variables $f(x_1, x_2, \ldots, x_n) = 0$. The function can be defined analytically, or with a function evaluation algorithm, or with tabulated values and an appropriate interpolation procedure. The major requirement of the function is at least C^0 continuity. There is a classification of points in E^n associated with the closed n-dimensional object with function representation. If $\mathbf{X} = (x_1, x_2, \ldots, x_n)$ is a point of E^n, then:

$f(\mathbf{X}) > 0$ if $\mathbf{X}$ is inside the object;

$f(\mathbf{X}) = 0$ if $\mathbf{X}$ is on the boundary of the object;

$f(\mathbf{X}) < 0$ if $\mathbf{X}$ is a point outside the object.

However, after applying operations to such an object, the constructed function may have points with $f = 0$ inside and on the boundary of the object. In some practical applications it can require special treatment. On the other hand, it gives much more freedom for designing operations on objects. Generally speaking, geometric objects defined by Eq. 1 are not regularized solids required in CSG. The object can have a boundary with dangling portions that are not adjacent to the interior. We define objects in multidimensional space for choosing a space of arbitrary dimension in each specific case. For example, if $n = 4$, then (x_1, x_2, x_3) can be space coordinates and x_4 can be interpreted as time.

Two major types of elements of the set M are basic geometric objects (primitives) and complex geometric objects. A geometric primitive is described by a specific instance of a function chosen from a finite set of possible types. A complex geometric object is a result of operations on primitives. In the modeling system, the finite set of primitives can be defined. However, it should be possible to extend this set in a symbolic manner. In this approach, the modeling system can be "empty" initially, while the user is responsible for an application-oriented filling of the primitive set. We consider this flexibility as one of the major advantages of the F-rep-based geometric modeling system.

2.2 Operations

The set of geometric operations Φ includes such operations as:

$$\Phi_i\colon\ M^1 + M^2 + \cdots + M^n \to M$$

where n is the number of operands of an operation. We consider only unary and binary operations in this paper. The result of the operations is also an object of the set M that ensures the closure property of the F-rep. Let object G_1 has the definition $f_1(\mathbf{X}) \geq 0$. For unary operations the object G_2 is said to be derived from G_1 as $G_2 = \Phi_i(G_1)$ and is defined by $f_2 = \Psi(f_1(\mathbf{X})) \geq 0$, where Ψ is a continuous real function of one variable. Examples of unary operations are bijective mapping, affine mapping, projection, offsetting. For binary operations, the object G_3 is said to be derived from G_1 and G_2 as $G_3 = \Phi_i(G_1, G_2)$ and is defined by $f_3 = \Psi(f_1(\mathbf{X}), f_2(\mathbf{X})) \geq 0$, where Ψ is a continuous real function of two variables. Examples of binary operations are set-theoretic operations, blending operations, cartesian product, and metamorphosis. As with the objects, the user of an F-rep-based modeling system can introduce any desired operation by its analytical or procedural description in symbolic form and thus extend the list of operations. In this connection, we shall not attempt to specify the complete set of possible operations on objects, but only introduce the most commonly used types.

The transformations of the function representation associated with operations on an object are described here. We use the logic of a step-by-step extension of the modeling environment from well-known set-theoretic operations to a less familiar metamorphosis.

2.2.1 Set-theoretic operations

The analytical definitions of the set-theoretic operations on functionally described objects have been introduced and studied by Rvachev (1963, 1974) for solving problems of mathematical physics in areas of complex shapes. Rvachev proposed these definitions to transform set-theoretic operations on areas described by Eq. 1 to operations on the defining functions. The resultant object is defined as follows:

$f_3 = f_1 | f_2$ for the union

$f_3 = f_1 \& f_2$ for the intersection

$f_3 = f_1 \backslash f_2$ for the subtraction

where f_1 and f_2 are defining functions of initial objects and |, &, \ are signs of the so-called R functions. One of the possible analytical descriptions of the R functions is as follows:

$$f_1 | f_2 = \frac{1}{1+\alpha}\left(f_1 + f_2 + \sqrt{f_1^2 + f_2^2 - 2\alpha f_1 f_2}\right)$$

$$f_1 \& f_2 = \frac{1}{1+\alpha}\left(f_1 + f_2 - \sqrt{f_1^2 + f_2^2 - 2\alpha f_1 f_2}\right) \quad (2)$$

where $\alpha = \alpha(f_1, f_2)$ is an arbitrary continuous function satisfying the following conditions:

$$-1 < \alpha(f_1, f_2) \leq 1,$$

$$\alpha(f_1, f_2) = \alpha(f_2, f_1) = \alpha(-f_1, f_2) = \alpha(f_1, -f_2) .$$

The expression for the subtraction operation is $f_1 \backslash f_2 = f_1 \& (-f_2)$.
Note that with this definition of subtraction, the resultant object includes its boundary.
If $\alpha = 1$, Eqs. 2 become

$$f_1 | f_2 = \max(f_1, f_2)$$

$$f_1 \& f_2 = \min(f_1, f_2) . \quad (3)$$

This is the particular case described by Ricci (1973). Equations 3 are very convenient for calculations but have C^1 discontinuity when $f_1 = f_2$. If $\alpha = 0$, Eqs. 2 take the most useful practical form:

$$f_1 | f_2 = f_1 + f_2 + \sqrt{f_1^2 + f_2^2}$$

$$f_1 \& f_2 = f_1 + f_2 - \sqrt{f_1^2 + f_2^2} . \quad (4)$$

Equations 4 have C^1 discontinuity only in points where both arguments are equal to zero. If C^m continuity is to be provided, one may use another set of R functions:

$$f_1 | f_2 = \left(f_1 + f_2 + \sqrt{f_1^2 + f_2^2}\right)\left(f_1^2 + f_2^2\right)^{\frac{m}{2}}$$

$$f_1 \& f_2 = \left(f_1 + f_2 - \sqrt{f_1^2 + f_2^2}\right)\left(f_1^2 + f_2^2\right)^{\frac{m}{2}} . \quad (5)$$

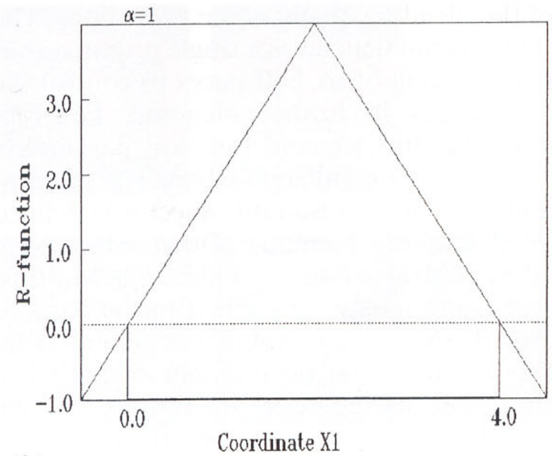
Generally, R functions correspond to standard, but not regularized, set-theoretic operations. They can result in "interior zeroes" of a defining function. Thus, in a general case, one cannot distinguish between a point of "interior zero" and a boundary point. If regularity of resulting objects is required, a special method of constructing the defining functions can be applied (Shapiro 1994). Note that the R functions can be used for describing geometric primitives. For example, the description of a segment in E^1 can be obtained from the descriptions of two rays as follows:

$$f(x) = (x_1 - b_1) \& (b_2 - x_1) .$$

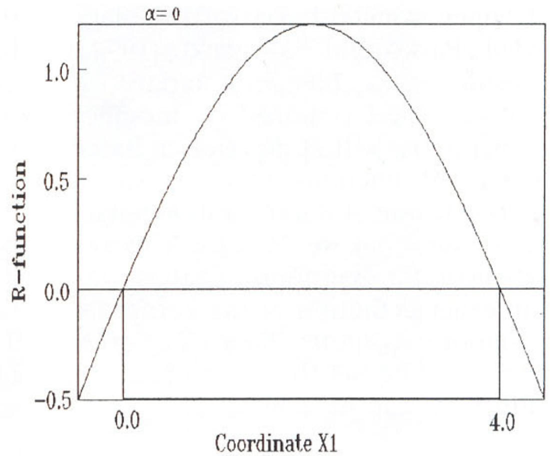
The plot of this function for Eqs. 3 is shown in Fig. 1a with $\alpha = 1$; Fig. 1b corresponds to Eqs. 4 with $\alpha = 0$, and Fig. 1c corresponds to Eqs. 5 with $m = 1$. It is important to point out that the function in Fig. 1b and c does not have points in its domain where the derivative is discontinuous. Such points can cause problems is subsequent operations on objects, especially when blending is used (Rockwood 1989). Figure 2 illustrates the process of construction of a 3D solid with the union and intersection operations.

2.2.2 Blending

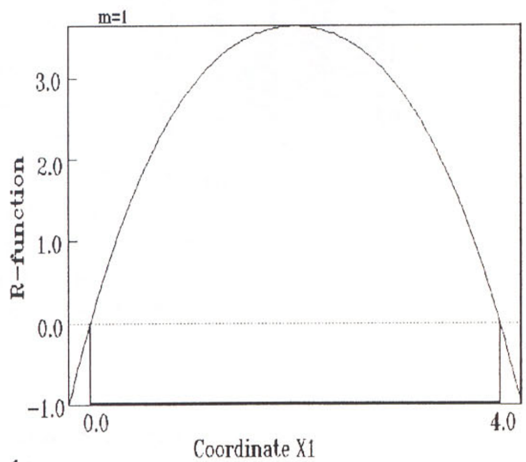
Although blending has been studied by many researchers, the continuous analytical description



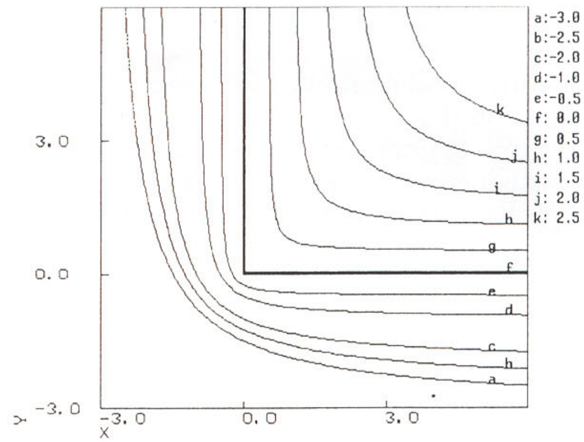
1a



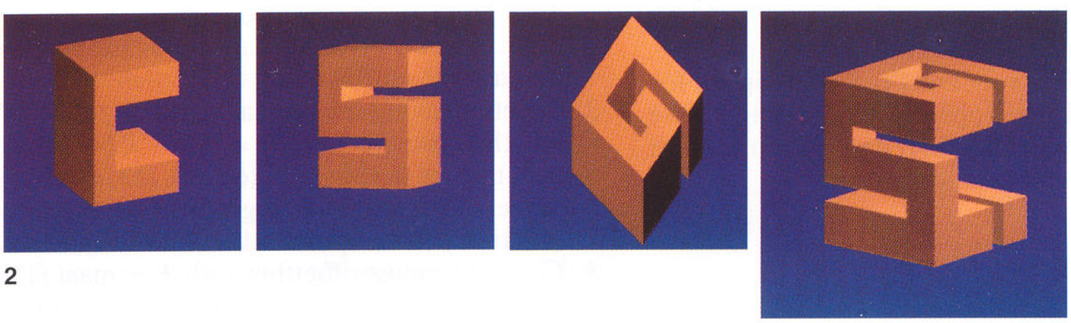
1b



1c



3



2

Fig. 1a–c. A segment in E^1 described as an intersection of two rays. The intersection operation is defined by different R functions: **a** $\alpha = 1$; **b** $\alpha = 0$; **c** m = 1

Fig. 2. Construction of a constructive solid geometry (CSG) solid. Three initial solids "C", "S", and "G" are defined as a union of blocks. The final solid is defined as an intersection ("C"$\cap$"S")$\cap$"G"

Fig. 3. The contour map of the intersection of two 2D halfspaces $x \geq 0$ and $y \geq 0$ represented by the R function with $\alpha = 0$. The contour with $f = 0$ is drawn with a *bold line*

of set-theoretic operations with controllable blend shape is needed. Pasko and Savchenko (1994a) proposed considering a blending surface as a boundary of an object obtained by modified set-theoretic operations with a description based on the technique of R functions. This approach is based on the observation of the contour behavior shown in Fig. 3. Note that we use Eqs. 4 here to define the set-theoretic operation. That is why $f = C$ gives an exact definition of the corner for $C = 0$ and smooth contour lines for other C values. The blending set-theoretic operation based on R functions can be defined as:

$$F(f_1, f_2) = R(f_1, f_2) + d(f_1, f_2) \tag{6}$$

where R is a corresponding R function, d is a displacement function that has a maximal absolute value $d(0, 0)$ and asymptotically approximates a zero value with increasing absolute values of the arguments. We have found that, for example, the following simple form of the displacement function is suitable for blending:

$$d(f_1, f_2) = \frac{a_0}{1 + (\frac{f_1}{a_1})^2 + (\frac{f_2}{a_2})^2}\,. \tag{7}$$

It is assumed that the defining functions for both objects have the distance property. The proposed displacement function is not the only one possible. Other gaussian-like functions can be designed for specific applications. Applying Eqs. 4, 6, and 7, we can get, for example, the final description form of the blending intersection operation:

$$F(f_1, f_2) = f_1 + f_2 - \sqrt{f_1^2 + f_2^2} + \frac{a_0}{1 + (\frac{f_1}{a_1})^2 + (\frac{f_2}{a_2})^2}\,. \tag{8}$$

This definition provides highly intuitive shape control of added material, subtracted material, and variable radius blends. It was used to generate aesthetic blends defined by hand-drawn strokes (see Sect. 4). Constant-radius blending is connected with the offsetting operation and is discussed later. Figure 4a illustrates an application of the blending set-theoretic operations. The basic block is not defined as a single primitive, but as an intersection of six halfspaces to control the shape of edges in further blending. Different values of the displacement function parameters were assigned to the different edges. The prominent front edge of the resultant object was defined by added material blending. Other edges were defined by subtracted material blending. Multiple blending operations were applied to the corners. The small cylindrical hole was made after blending to show that no problem occurs when applying the set-theoretic operations to the blended object. This is ensured by the continuity of Eq. 8.

2.2.3 Offsetting

To generate expanded or contracted versions of an initial object, one can apply a positive or a negative offsetting operation, respectively. The descriptions of the following three offsetting operations have been proposed by Pasko and Savchenko (1994b):

1. Iso-valued offsetting with $F = f(\mathbf{X}) + C$; where the negative constant C defines the negative offset, and the positive C defines the positive offset. Figure 4b shows an application of this operation to a 3D solid.
2. Offsetting along the normal with $F = f(\mathbf{X} + D\mathbf{N})$ for the positive and $F = f(\mathbf{X} - D\mathbf{N})$ for the negative offsetting, where D is the given distance value, and $\mathbf{N}$ is a gradient vector of the function f in the point with $\mathbf{X}$ coordinates.
3. Constant-radius offsetting with $F = \max(f(\mathbf{X}'))$ for the positive and $F = \min(f(\mathbf{X}'))$ for the negative offsetting, where $\mathbf{X}'$ is a vector of the coordinates of points belonging to the sphere of the given radius D and the center at $\mathbf{X}$. Note that continuity and distance property of the defining function of the object are also used here. The procedure of the constant-radius blending of the object's convex edges includes consequently applied negative and positive constant-radius offsetting (Rossignac and Requicha 1984). This procedure applied to a 2D solid is illustrated in Fig. 4c.

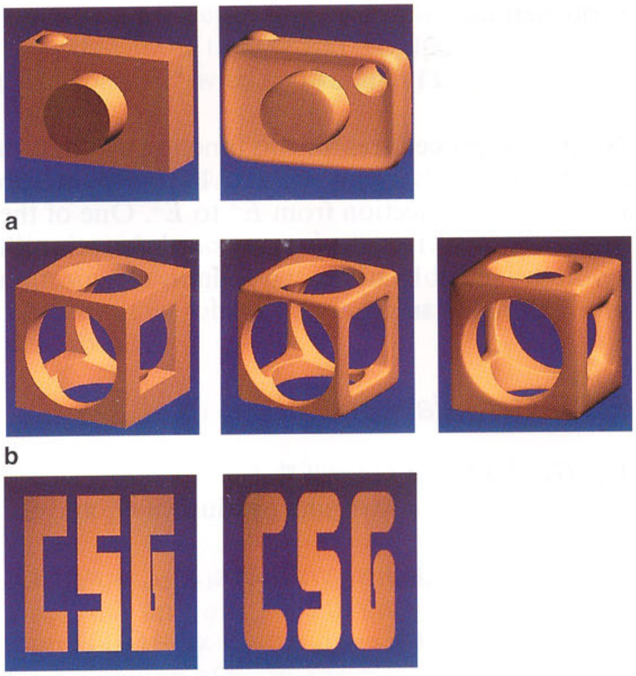


Fig. 4. **a** Blending followed by the set-theoretic subtraction (a cylindrical hole); **b** negative and positive offset solids obtained by the isovalued offsetting operation; **c** rounding convex vertices of a 2D solid by blending based on the constant-radius offsetting

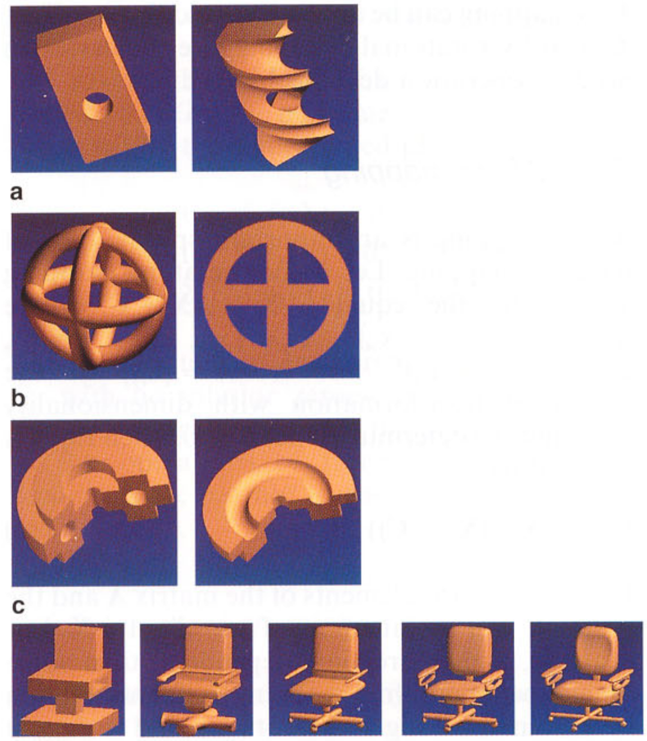


Fig. 5. **a** Twisting a constructive solid with the bijective mapping; **b** a union of three tori and a projection of the object to a plane orthogonal to an axis of one of tori; **c** the application of the cartesian product and bijective mapping: a 3D solid defined by the rotational sweeping; **d** several steps of time-dependent metamorphosis of 3D solids

2.2.4 Bijective mapping

Let Φ_i be defined by the coordinate transformations

$$x'_j = \varphi_j(x_1, x_2, \ldots, x_n), \quad j = 1, \ldots, n$$

where φ_j are continuous real functions. We assume also that inverse functions φ_j^{-1} exist. Then the resultant object is described as

$$G_2\colon f_1(\varphi_1^{-1}(x'_1, x'_2, \ldots, x'_n), \ldots, \varphi_n^{-1}(x'_1, x'_2, \ldots, x'_n)) \geq 0 \,. \tag{9}$$

The examples of such a bijective mapping are tapering, twisting, bending, (Barr 1984) and modal deformations (Sclaroff and Pentland 1991). The twisting of a solid constructed by set-theoretic operations is shown in Fig. 5a. Another example is the mapping of an object, defined in an arbitrary coordinate system, to a cartesian coordinate system. The inverse functions φ_j^{-1} from Eq. 9 for the mapping of an object from the cylindrical coordinate system to the cartesian one is defined as:

$$x_1 = \varphi_1^{-1}(x'_1, x'_2, x'_3) = \sqrt{x_1'^2 + x_2'^2}$$

$$x_2 = \varphi_2^{-1}(x'_1, x'_2, x'_3) = \arctan(x'_2/x'_1)$$

$$x_3 = \varphi_3^{-1}(x'_1, x'_2, x'_3) = x'_3 \tag{10}$$

where

$x_1 = \rho, x_2 = \theta, x_3 = z$ are cylindrical coordinates,

and

$x'_1 = x, x'_2 = y, x'_3 = z$ are cartesian coordinates.

This mapping can be applied to describe an object defined by rotational sweeping (see the cartesian product operation described later).

2.2.5 Affine mapping

Affine mapping is an important specific case of bijective mapping. Let Φ_i be the affine mapping defined by the equality $\mathbf{X}' = \mathbf{AX} + \mathbf{C}$, where $\mathbf{X} = (x_1, x_2, \dots, x_n)^T$, $\mathbf{X}' = (x'_1, x'_2, \dots, x'_n)^T$, $\mathbf{C} = (c_1, c_2, \dots, c_n)^T$, and let $A = \{a_{ij}\}$ be the matrix of transformation with dimensionality $n \times n$ and the determinant $\det \mathbf{A} \neq 0$. Then Eq. 9 is changed to:

$$G_2\colon f_1(\mathbf{A}^{-1}(\mathbf{X}' - \mathbf{C})) \,. \tag{11}$$

In a general case, elements of the matrix **A** and the vector **C** can be functions of coordinates **X**. For example, if they are time dependent, then some time-dependent affine mapping is defined. It can be movement along a line, rotation and scaling in time, delay and bringing forward, complex movement with time mapping, etc. The unification of mappings of space coordinates and time allows an effective description of complex geometric processes, combining models used in a description of geometric volumes, and complex movements in animation.
Next we consider the operation of projection where $\det \mathbf{A} = 0$.

2.2.6 Projection

R functions can also be applied for approximately describing the projection operation from E^n to E^{n-1} that does not have an inverse transformation Φ^{-1}. Let us define

$$G_1 \subset E^n\colon f_1(x_1, x_2, \dots, x_i, \dots, x_n) \geq 0$$

$$G_2 \subset E^{n-1}\colon f_2(x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots, x_n) \geq 0,$$

and G_2 is a projection of G_1 to E^{n-1}. The object G_2 can be defined as a union of sections of G_1 by hyperplanes $x_i = C_j$ where $C_{j+1} = C_j + \Delta x_i$, $j = 1, N$ and $C_1 = x_{i\,\min}$.
Let $f_{1j} = f_1(x_1, x_2, \dots, x_{i-1}, C_j, x_{i+1}, \dots, x_n)$ be a defining function for a section. Then the defining function for the projection with $\Delta x_i \to 0$ can be expressed as:

$$f_2 = f_{11} \,|\, f_{12} \,|\, \dots \,|\, f_{1j} \,|\, \dots \,|\, f_{1N} \,. \tag{12}$$

Numerical procedures for this function evaluation are discussed elsewhere (Pasko 1988). Figure 5b illustrates a projection from E^3 to E^2. One of the applications of this operation is a description of a volume swept by a moving solid as a projection of a 4D object to E^3.

2.2.7 Cartesian product

Let $G_1 \subset E^k$ and $G_2 \subset E^m$.
We define G_3 as a cartesian product of G_1 and G_2:

$$G_3 = G_1 \times G_2 = \{(x_1, x_2, \dots, x_n) \colon (x_1, x_2, \dots, x_k) \in G_1, \\ (x_{k+1}, x_{k+2}, \dots, x_n) \in G_2\}$$

where $G_3 \subset E^n$ and $n = k + m$. The defining function for G_3 can be obtained with R functions:

$$f_3(x_1, x_2, \dots, x_n) = f_1(x_1, x_2, \dots, x_k) \\ \& \, f_2(x_{k+1}, x_{k+2}, \dots, x_n) \,. \tag{13}$$

Swept objects can be defined with the help of the cartesian product and the bijective mapping. A 3D object defined by the rotational sweeping is shown in Fig. 5c. Firstly, the 3D object was constructed as a cartesian product of a 2D solid and a 1D segment. Then bijective mapping (Eq. 10) was applied to this 3D object. In other words, its x coordinate was interpreted as the angle, and its y coordinate as the radius of a cylindrical coordinate system. Another application is a definition of time-dependent geometric objects on some time interval by the cartesian product of a static object and a time segment or a ray. For example,

$G_1 \subset E^3\colon f_1(x, y, z) \geq 0$ is a static object in E^3

$G_2 \subset E^1\colon (t - t_1) \& (t_2 - t) \geq 0$ is a time segment $[t_1, t_2]$.

$G_3 \subset E^4$,

$$G_3 = G_1 \times G_2\colon f_1(x, y, z) \,\&\, ((t - t_1) \,\&\, (t_2 - t)) \geq 0. \tag{14}$$

G_3 is a time-dependent geometric object being activated at the time t_1 and terminated at the time t_2 in space E^4 with coordinates (x, y, z, t) .

2.2.8 Metamorphosis

We consider metamorphosis as a binary operation on two objects G_1 and G_2 defined in E^{n-1}. The resultant object G_3 is defined in E^n and is described as

$$f_3(x_1, x_2, \ldots, x_n) = f_1(x_1, x_2, \ldots, x_{n-1}) \cdot (1 - g(x_n)) + f_2(x_1, x_2, \ldots, x_{n-1}) \cdot g(x_n) \quad (15)$$

where $g(x_n)$ is a positive continuous function, $g(x_n^0) = 0$, and $g(x_n^1) = 1$. This means that G_1 is a section of G_3 by the hyperplane $x_n = x_n^0$, and G_2 is a section of G_3 by the hyperplane $x_n = x_n^1$ in E^n. For $n = 4$, an object G_3 can be thought of as a time-dependent object reconstructed from its two instances at different moments. Figure 5d shows several time steps of the metamorphosis of a 3D solid. For $n = 3$, this operation generates a 3D solid reconstructed from its two planar cross-sections. This can be useful in tomography and range-data processing.

2.3 Relations

We consider only binary relations as the subsets of the set M^2. The examples of binary relations are inclusion, point membership and intersection. Similarly, as with objects and operations, we allow the user to extend the set of relations by symbolically defining their predicates.

2.3.1 Inclusion relation

This relation is described as $G_2 \subset G_1$, and means that the object G_2 is included in G_1. If G_2 is the point P, the relation can be described by the following bivalued predicate:

$$S_2(P, G_1) = \begin{cases} 0 & \text{if } f_1(\mathbf{X}) < 0 \text{ for } P \notin G_1 \\ 1 & \text{if } f_1(\mathbf{X}) \geq 0 \text{ for } P \in G_1 \end{cases} . \quad (16)$$

2.3.2 Point membership relation

Let iG_1 be the interior of G_1 and bG_1 be the boundary of G_1. The point membership relation is described by the three-valued predicate:

$$S_3(P, G_1) = \begin{cases} 0, & \text{if } f_1(\mathbf{X}) < 0 \quad \text{for } P \notin G_1 \\ 1, & \text{if } f_1(\mathbf{X}) = 0 \quad \text{for } P \in bG_1 \\ 2, & \text{if } f_1(\mathbf{X}) > 0 \quad \text{for } P \in iG_1 \end{cases} \quad (17)$$

This predicate can be correctly evaluated for G_1 with no interior zeroes. Note that the set-theoretic operations with R functions correspond to the operations of three-valued logic over predicates S_3, but not to the boolean logic over predicates S_2.

2.3.3 Intersection relation

The relation is defined by the two-valued predicate:

$$S_c(G_1, G_2) = \begin{cases} 0, & \text{if } G_1 \cap G_2 = \emptyset \\ 1, & \text{if } G_1 \cap G_2 \neq \emptyset \end{cases} \quad (18)$$

A function $f_3(\mathbf{X}) = f_1(\mathbf{X}) \;\&\; f_2(\mathbf{X})$ defining the result of the intersection can be used to evaluate S_c. It can be stated that $S_c = 0$ if $f_3(\mathbf{X}) < 0$ for any point of E^n. This definition leads to the collision detection algorithm discussed in Sect. 4.

3 Interactive geometric modeling based on the function representation

3.1 A machine representation and a user representation

In the previous section, we introduced F-rep as a mathematical notion of an abstract nature to define basic geometric concepts – objects, operations, and relations – in terms of real functions of several variables. This section is devoted to describing our experience of building an interactive geometric modeling system based on the F-rep notion. Accordingly, the very notion "representation" is treated in a more applied sense, as concerned with the computer's and user's manipulation and interpretation of it in our current modeling system.

Snyder (1992) believes "the representation to be the part of a geometric modeling system which most determines its quality." We agree, and consider F-rep essentially as a "user representation" for specifying the user's geometric model in a symbolic form. We explicitly distinguish it from a lower-level "machine representation" that can be present inside the system. This lower level can be in the form of a generalized CSG-type tree with, in turn, other levels, such as collections of polygons. The corresponding set of procedural tools forms a kernel geometric modeler. Pasko et al. (1993a) give the VDM (Vienna Development Method) specification of such a modeler that formally defines principal data structures of a "machine representation" as well as operations over them. This modeler, together with a visualization subsystem, serves as a basis for building an interactive geometric modeling system.

3.2 Geometric language and an example of modeling

F-rep as a user representation serves as a base for a high-level geometric language that is the user's instrument for modeling specifications. To see how significant features of F-rep are reflected in the geometric language, let us consider a corresponding modeling program (Fig. 6). This program creates a model of cycled metamorphosis of the following four geometric objects that are normally modeled in different representational styles:

1. A constructive object defined with the help of set-theoretic operations on primitives.
2. A swept object defined by the cartesian product and subsequent bijective mapping.
3. A voxel-based object built by manual sculpting similar to that of Galyean and Hughes (1991) with subsequent trilinear interpolation providing C^0 continuity to make this volumetric teapot a "legal" F-rep object.
4. A "blobby" object as a representative of objects with analytically defined implicit surfaces.

Finally, the metamorphosis itself is modeled with using the objects described as "key volumes" and eventual getting necessary intermediate volumes in accordance with Eq. 15. Figure 7 represents the frames of the computer film corresponding to the specification of the resultant 4D geometric object *gob_metamorp* from Fig. 6.

The program interactively introduced during a modeling session consists of the following parts:

- Geometric model
- Geometric types
- Environment

Each part can be defined and changed irrespective of others in the interactive modeling work. A brief description of these parts is given here.

3.2.1 Geometric model

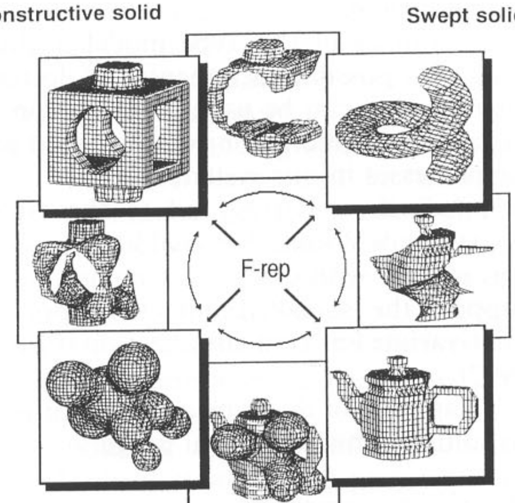
This is a parametrized specification of the geometric objects themselves that is intimately related to the F-rep. Geometric objects are given by their defining functions. In keeping with the mathematical framework described in Sect. 2, each function defining a complex geometric object *gob_*$\langle name \rangle$ is built as a rather traditional mathematical expression with symbols of coordinate variables x_i, geometric primitives *pob_*$\langle name \rangle$, numerical constants and parameters, arithmetic operations, standard algebraic functions (sin, cos, log, min, max, etc.), and built-in set-theoretic operations implemented by R functions (| for union, & for intersection, \ for subtraction, and @ for cartesian product). One can choose the type of R function by setting the *r_alpha* parameter. If the built-in *pob_block_3D* was defined with the help of a "min-max" system of R functions with *r_alpha* $= 1$ (Eqs. 3), the user can introduce a new one defined with the help of the system of R functions (Eqs. 4) with better continuity properties. Such operators of structured programming as "*if-then-else*" and "*while-do*", together with a complete set of logical functions, provide proper programming flexibility for defining complex and nontraditional geometric transformations and relations.

Note that, besides the built-in conventional primitives and operations, the user can introduce new ones during a modeling session. For instance, if there is a need for a "block" with blended edges, one can define the following "blending

```
{ Metamorphosis }
{ "Geometric model" section }

{ Setting the system of R-functions ("|", "&", "\", "@") }
r_alpha = 1.
{ Primitive "block" }
pob_block_3D(a0,b0,c0,a,b,c,)=  (x1-a0 & a0+a-x1)& (x2-b0 & b0+b-x2)& (x3-c0 &c0+c-x3);

{ CSG object }
{ Basic block }
gob_bl1 = p_block_3D(-9.,-8.,-9.,18.,16.,18.);

{ Vertical cylinder }
gob_cyl1 = (r1**2-x1**2-x3**2) & 10.5-x2) & x2+10.5;

{ Horizontal infinite cylinder }
gob_cyl2 = r2**2-x1**2-x2**2;

{ Smaller block with infinite x1-dimensionality }
gob_bl2 = (x2+d1 & d1-x2) & (x3+d1 & d1-x3);

{ final CSG object }
gob_csg = ( ( gob_bl1 | gob_cyl1 ) \ gob_cyl2 ) \ gob_bl2;

{ Swept object built with help of cartesian product and bijective mapping }
gob_swept = ( r3**2 - (x1-c1*SIN(-1+x2/c2)**2 - (x3-c1*COS(-1+x2/c2)**2 ) @ ( x2+10. & 10.-x2);

{ Voxel-based object defined by 3D array in a file with subsequent interpolation }
gob_vox = INTERPOLATE("trilinear", "teapot.dat", x1,x2,x3);

{ blobby object }
gob_blobby = b;
WHILE ( i <= 8 ) DO
gob_blobby = gob_blobby + a[i]*EXP(-SQRT( (x1-px[i])**2+ (x2-py[i])**2+ (x3-pz[i])**2);

{Metamorphosis between all objects }
gob_metamorp =      IF ( 0 <= x4 < = 1 )
                    THEN gob_csg*(1-x4) + gob_swept*x4;
                    ELSE IF ( 1 < x4 <= 2 )
                              THEN gob_swept*(2-x4) + gob_vox*(x4-1);
                              ELSE IF ( 2 < x4 <= 3 )
                                        THEN gob_vox*(3-x4) + gob_blobby*(x4-2);
                                        ELSE IF ( 3 < x4 <= 4 )
                                                  THEN gob_blobby*(4-x4) + gob_csg*(x4-3);


{ "Geometric types" section }
x1 : X;
x2 : Y;
x3 : Z;
x4 : T;
{ "Environment" section }
{ modeling space }
x1min = -11.;
x1max = 11.;
x2min = -11.;
x2max = 11.;
x3min = -11;
x3max = 11.;
x4min = 0.;
x4max = 4.;
{ incremental interval for x4 with geometric type "T" }
x4_delta = 0.5;
{ "Parameters" }
r1 = 4.;
r2 = 6.;
d1 = 5.;
r3 = 3.5;
e1 = -8.;
e2 = 2.5;
b  = -0.07;
a = [1.5, 1., 1., 0.8, 0.5, 0.3, 0.4, 0.35 ];
px = [0., 6., -4., -5., 5., 6., 5., 8. ];
py = [0., 6., -4., 3., -5., -7., -6., -9. ];
pz = [0., 6., -4., 7., 5., 9., -3., -4. ];
```

6

Fig. 6. Metamorphosis modeling program in a high-level geometric language



7

Fig. 7. Frames of a metamorphosis process of "key volumes" reflecting different representational styles: constructive geometry, sweeping, soft objects, and voxel-based objects

intersection" transformation based on Eq. 8:

$$tr\_bl\_int((gob\,|\,pob)\,g1, (gob\,|pob)\,g2, (\text{real})\,a1, (\text{real})\,a2, (\text{real})\,a3) = g1 + g2 - \text{sqrt}(g1^2 + g2^2) + a1/(1 + (g1/a2)^2 + (g2/a3)^2).$$

This operation can be applied to halfspaces to get a block with smooth edges and corners.

3.2.2 Geometric types

Each coordinate variable x_i can be associated with a certain "geometric type." These types establish conventions governing their semantics by giving a geometric interpretation of x_i. This interpretation can be important while exploring the geometric model, particularly in visualization. The following geometric types in the current version of our system are:

- "*constant*": variables of this type are assigned numerical values to define a single cross-section, such as $x_i = const$.
- "g": these variables define a group of constants corresponding to several cross-sections.
- "x", "y", "z": these are coordinate variables corresponding to axes in the 3D cartesian coordinate system.
- "t": variables of this type model a course of time with possible incremental or decremental changes that can be used in animation. Time-dependent geometric objects are called geometric processes in our system.
- "v", "w": these correspond to the additional V-axis and W-axis for building a geometric spreadsheet with elementary images in cells to support the so-called inductive approach to multivariate function visualization (Pasko et al. 1992).
- "c": variables of this type are used for mapping to colors within a spectral range.

Note, that the geometric types "*constant*" and "g" can also be assigned to the defining function of the resultant geometric object. Normally, the zero value sets the boundaries of a geometric object. If type "g" is assigned, the visualization of corresponding isolines and isosurfaces can be very useful for exploring the features of the defining function (Fig. 3).

3.2.3 Environment

This section is intended for concretizing those parameters present in the "Geometric Model" that were denoted by their abstract names. The first subsection defines the ranges $[x_{i\,\min}, x_{i\,\max}]$ of coordinate variables to set boundaries of a modeling space. Then an incremental interval x_i_delta for a coordinate variable of geometric type "t" must be defined. In the example, the value $x_i\_delta = 0.5$ lets us get one intermediate frame between each pair of "key volumes." The next subsection deals with assigning the values of numerical parameters, including arrays. The user can specify the necessary visualization parameters that essentially depend on the assignment of geometric types to coordinate variables.

3.3 Advantages of function representation as a base for the interactive modeling system

We use (with some modifications) the following three criteria for evaluating F-reps that were introduced by Synder (1992) for user representation as a special adaptation of the categories from Requicha (1980).

3.3.1 Ease of specification

This is the first and the most crucial criterion for evaluating the quality of a user representation, which basically assesses how efficiently the user can define and change his geometric model.
The F-rep is *closed*, meaning that further operations can be applied to results of previous ones. This representation is also *uniform* in the sense that it supports combinations of representational styles to describe objects of a traditionally different nature. These two properties have already been substantiated.
This results in the conclusion about the *higher abstraction level* regarding many other representations. Uniform definitions for objects in spaces

of various dimensions, as well as uniform representation of static and time-dependent objects, are especially attractive. Moreover, *multidimensionality* can be interpreted in the interesting and natural way through the concept of geometric types.

It is obvious that F-rep, because of its analytical nature, provides categories important for the user's modeling work such as the *compactness* and *accuracy* of a model.

Because of the thousand-year tradition of analytical description in geometry, we consider F-rep as *natural* for our users. Even elementary knowledge of analytical geometry and brief training lead users to connect the way they think about geometric shapes to symbolic descriptions in the form of analytical expressions. The very nature of F-rep lets users easily change some parameters in the analytical model and observe the visual or computational effect that sets the understandable correspondence of an analytical description and the model's behavior. Thus, *editability* and *controllability* of description are naturally provided.

Last, but not the least, one should mention *extendibility*, meaning the option of introducing new primitives and operations, not only by procedural definition, but also by symbolic definition, during a modeling session. This means that one can create a specific modeling system for particular application areas and even for particular users who can change details of the system if they wish. This is a base for the "Empty Case" technology of geometric modeling (Pasko et al. 1993b), which supposes a working process of an "absolutely first user" who must create his personal geometric modeling system by defining the necessary primitives and transformations in a symbolic manner. This seems to be especially valuable in an educational perspective, and a corresponding project is being realized at the University of Aizu.

3.3.2 Renderability

We consider F-rep suitable for providing fast visual feedback to the user. This is achieved by conversion to a polygonal mesh. The original algorithm of polygonization of an isosurface $f(x_1, x_2, x_3) = 0$ (Pasko et al. 1988) was used to generate the frames in Fig. 7. This algorithm can easily be decomposed to map on a parallel computer architecture. Higher-quality rendering is also possible with ray casting, which also undergoes parallelization very well. The compactness of F-rep allows to run rendering software, even on parallel computers with little distributed memory (e.g., transputer networks).

3.3.3 Analyzability

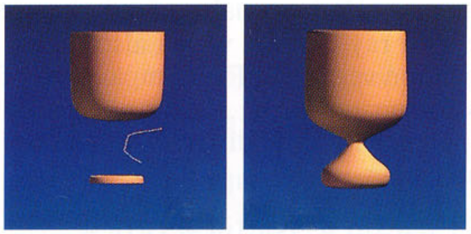
Geometric queries, such as point inclusion, are easily handled in the F-rep, which makes it easier to compute physical quantities about the shape (volume, moments of inertia) with well-known algorithms. Collision detection can be realized by a maximum search procedure (see Sect. 2.3 and the example in Sect. 4). However, finding curves of surface intersections requires slow numerical algorithms.

3.4 Perspective interactive environment

The F-rep as a high-level user representation fits in very well with "exploratory geometric modeling." The properties and features of a geometric model are initially given, but the model is not simply described. Rather, new geometric objects and transformations are introduced. Subsequently, their characteristics and behavior are explored interactively. This creative process is similar to a traditional scientific investigation of a physical phenomenon when one experiments with the model created and observes its behavior under changeable conditions.

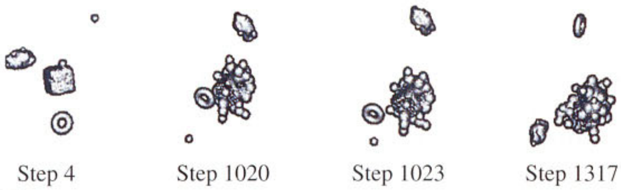
We think the corresponding interactive environment can be built on the basis of the "definitively based" programming paradigm (Beynon 1989) and the agent-oriented framework, which provides easy interactivity for modifying the specification (of both the parameters in defining functions and the functions themselves) with indivisible propagation of changing any dependent entities.

The corresponding implementation work is in progress. Adzhiev et al. (1994) propose a LSD specification of a geometric modeling system. This specification describes, within an agent-oriented modeling framework, both the interactions of the user with the system and the interactions between the principle components of the system itself.

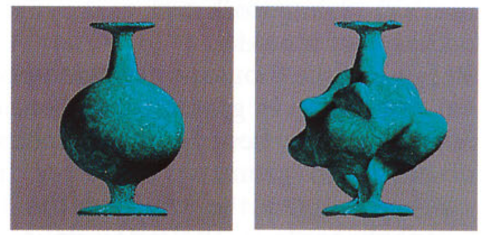


8a b

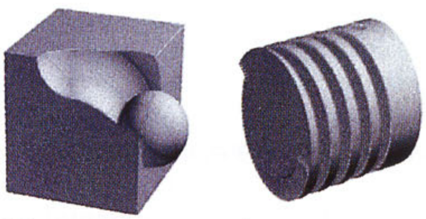
11



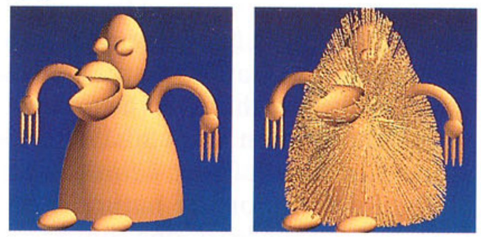
9



12a



10a b



b

Fig. 8. **a** The body and the bottom of a wine glass to be connected with an aesthetic blend defined by the hand-drawn stroke; **b** The result of blending with the estimated parameters

Fig. 9. Simulation of colliding particles sticking together

Fig. 10. Application of the set-theoretic operations for numerically controlled (NC) machining: **a** the set-theoretic substraction between two moving solids; **b** the cut object achieved as a result of the subtraction of the solid swept by a linearly moving cutter from the rotating workpiece

Fig. 11. Two functionally represented cross-sections and a solid reconstructed by the metamorphosis operation

Fig. 12. **a** Metamorphosis of a vase and solid noise produces a noisy vase; **b** a furry object defined by the offsetting and intersection with fur strands described with solid noise

This specification can serve as a basis for the parallel implementation of the interactive modeling system with an advanced graphical user interface.

4 Application examples

Here we give several examples of applications of geometric modeling software based on the F-rep principles.

4.1 Aesthetic design

Although the blending operations described in Sect. 2.2 provide easy shape control, they seem too indirect for aesthetic blending. Rather than changing numeric parameters, a designer prefers defining the shape of an aesthetic blend with a single hand-drawn stroke. These points are then used to estimate the values of parameters that define the blend as close to this stroke as possible. This is illustrated in Fig. 8. The stroke belongs to a certain plane. Points of hand-drawn strokes are assumed to belong to the blend surface. These constraints determine a system of nonlinear equations:

$$F(f_1(x_i, y_i, z_i), f_2(x_i, y_i, z_i), a_0, a_1, a_2) = 0,\ i = 1, N$$

where f_1 and f_2 are defining functions of initial objects, F defines the blending set-theoretic operation (Eq. 6), and N is a number of points. To find the best estimation of a_0, a_1, and a_2 in the sense of least squares, we applied a simple quadratic search and used random points for the initial estimate.

4.2 Simulation of collisions

Several application problems (e.g., air and water quality control, collision dynamics of bodies in celestial mechanics, computer games) deal with irregularly shaped interacting solids. Figure 9 illustrates the simulation of colliding particles sticking together. It presents collisions of a noisy block, small spheres, a torus, and a noisy ellipsoid. The spheres colliding with the noisy block are stuck to it by blending union. The surfaces of the block and the ellipsoid were generated with solid noise (discussed later). All bodies are variably oriented in space and interact with one another as rigid bodies. The collision-detection algorithm is based on the intersection relation defined in Sect. 2.3. To find a collision point of two particles, we apply the maximum searching algorithm to the defining function of the particle intersection. The admissible domain is detected with bounding spheres. Then we use a spiral quadratic search within this domain to detect a point with a positive or zero function value. The simulating algorithm was implemented on transputers T805 in OCCAM-2 (INMOS 1988) and is described in Savchenko and Pasko (1993). It has simulated 1320 time steps for 35 small spheres. Steps 1020 and 1023 in Fig. 9 illustrate the collision between the torus and the newly formed object.

4.3 Modeling of numerically controlled (NC) machining

The set-theoretic operations between moving solids can be used to model a process of NC machining. The result can be defined with a set-theoretic subtraction between the workpiece model and the swept model of the moving tool. Sourin and Pasko (1994) have proposed the procedural function representation of a swept solid with an envelope surface for a parametrically defined trajectory. Figure 10 illustrates its application to modeling the time-dependent set-theoretic operations.

4.4 Reconstruction of solids from cross-sections

Tomography, range data processing, and other applications require the reconstruction of solids from their given cross-sections. The metamorphosis operation (Eq. 15) describes a solid using defining functions of two parallel cross-sections. The generation of a defining spline function of a cross-section given by its contour points is proposed by Savchenko et al. (1994). The reconstruction is illustrated in Fig. 11. Note that this approach is capable of generating highly concave and branching solids automatically.

4.5 Three-dimensional texture generation

To obtain 3D textures on constructive solids, Pasko and Savchenko (1993) have proposed applying set-theoretic and other operations to a solid defined by a "solid noise" function. Figure 12a shows an irregularly shaped vase. The initial vase was designed by the aesthetic blending of several ellipsoids. The irregular shape was obtained by metamorphosis of the initial vase and solid noise. Figure 12b shows fur obtained by offsetting and set-theoretic intersection applied to an initial solid and fur strands defined with solid noise.

5 Summary and future work

In conclusion, we would like to summarize the main advantages of F-rep and to discuss future research developments.
F-rep for geometric objects offers a number of advantages:

- An abstraction level higher than those of other known representations is provided. Combinations of representational styles, including constructive geometry, sweeping, soft (blobby) objects, and voxel-based objects, are supported.
- The option of the symbolically defining new primitives, operations and predicates is naturally provided. A symbolic description of a complex geometric object as a result of modeling can be generated too.

- Objects defined in spaces of various dimensions are uniformly represented. Dimension-increasing (cartesian product) and dimension-decreasing (projection) operations are supported. Static objects and time-dependent geometric processes are described uniformly, with time as one of the coordinates.
- Convenience of designing application algorithms, especially for parallel computers, is provided. Compactness of the representation allows the implementation of application algorithms, even on parallel computers with little distributed memory, for example, on a transputer network.

There are several problems in using F-rep. Evaluating a defining function in a given point is a time-consuming task. Moreover, if R functions with square roots are applied, calculations become slower. The halftone images presented in the paper have been produced on a Silicon Graphics Indigo^2 machine by ray casting. The average time for a 200×200 image calculation in double precision is 20–90 s. One can suppose that the numerical stability of nested square roots (see Eqs. 2) is questionable. Our numerical experiment has shown that R union (Eqs. 4) applied to 10 000 spheres (calculated in double precision) gives an error of $0.2 \cdot 10^{-15}$ for zero value of the defining function in a boundary point. Although graphic workstations provide acceptable response times even for ray casting, we see the final solution in parallel computing and special hardware. The parallel implementation of the polygonization algorithm (Savchenko and Pasko 1994) improved the performance of the computations and practically linearly scaled with the number of processors.
In practical systems, conversion from boundary representations (B-rep) may be required. Shapiro (personal communication, 1994) suggested that the way to convert B-rep to F-rep is first to convert it to a constructive representation with standard (nonregularized) set operations, and then to F-rep with R functions. The problem of converting B-rep to the standard set operations is similar to B-rep/CSG conversion (Shapiro and Vossler 1993), but not identical. This problem needs further investigation.
Now, there is no direct connection between F-rep and parametric representations. Because parametric surfaces are very suitable for interactive geometric design, we try to incorporate these models in the function representation. Spline-controlled deformations of constructive solids are also very attractive.
F-rep requires users to define a model in a highly abstract way. Coordinate variables, numerical constants and parameters, arithmetic operations, and standard algebraic functions must always be explicitly defined by the users. It can be difficult for general users to define objects in this way.
Although symbolic description is a powerful modeling tool, adequate graphical user interface must be specified for F-rep-based modeling. However, the facility for extending a modeling system with symbolic descriptions of new elements must be preserved. Interrelations with computer algebra systems will be researched. Now we are going to apply F-rep to virtual reality applications.

Acknowledgements: We sincerely thank Dr. V.V. Pilyugin from the Moscow Engineering Physics Institute (MEPI) and Dr. W.M. Beynon from the University of Warwick for the fruitful theoretical discussions. We thank Dr. V.A. Galatenko and V.K. Nikolaev from Scientific Research Institute of System Analysis, Russian Academy of Sciences (SRISA RAS) for software support. The authors also acknowledge the referees for their helpful comments.

References

Adzhiev VD, Beynon WM, Pasko AA (1994) Interactive geometric modeling based on R-functions: an agent-oriented approach. Set-theoretic Solid Modeling: Techniques and Applications, CSG'94 Conference Proceedings, Information Geometers, Winchester, UK, pp 253–272

Bajaj C, Ihm I, Warren J (1993) Higher-order interpolation and least-squares approximation using implicit algebraic surfaces, ACM Trans Graph 12:327–347

Barr AH (1981) Superquadrics and angle-preserving transforms. IEEE Comput Graph Appl 1:11–23

Barr AH (1984) Global and local deformations of solid primitives, Comput Graph 18:21–30

Baraff D (1990) Curved surfaces and coherence for non-penetrating rigid body simulation. Comput Graph 24:19–28

Beynon WM (1989) Evaluating definitive principles for interaction in graphics. In: Earnshaw RA and Wyvill B (eds) New Advances in Computer Graphics, Proceedings of CG International'89, Springer-Verlag, New York, 291–303

Blinn JF (1982) A generalization of algebraic surface drawing, ACM Trans Graph 1:235–256

Bloomenthal J (1988) Polygonization of implicit surfaces. Comput Aided Geom Design 5:341–355

Bloomenthal J, Shoemake K (1991) Convolution surfaces. Comput Graph 25:251–256

Bloomenthal J, Wyvill B (1990) Interactive techniques for implicit modeling. Comput Graph 24:109–116

Duff T (1992) Interval arithmetic and recursive subdivision for implicit functions and constructive solid geometry. Comput Graph 26:131–138

Essa IA, Sclaroff S, Pentland A (1992) A unified approach for physical and geometric modeling for graphics and animation. Comput Graph Forum 11:129–138

Galyean TA, Hughes JF (1991) Sculpting: an interactive volumetric modeling technique. Comput Graph 25:267–274

Gascuel MP (1993) An implicit formulation for precise contact modeling between flexible solids. SIGGRAPH'93, Computer Graphics Proceedings, Annual Conference Series 27:313–320

Hoffman CM (1989) Geometric and solid modeling. Morgan Kaufmann, San Mateo, Calif

Hoffmann CM (1993) Implicit curves and surfaces in CAGD, IEEE Comput Graph Appl 13:79–88

Hoffmann, C, Hopcroft J (1987) The potential method for blending surfaces and corners. In: Farin G (ed) Geometric Modeling, SIAM, Philadelphia, pp 347–365

Hughes JF (1992) Scheduled Fourier volume morphing. Comput Graph 26:43–46

INMOS (1988) OCCAM-2. Reference manual, INMOS, Bristol UK

Lorensen WE, Cline HE (1987) Marching cubes: a high resolution 3D surface construction algorithm. Comput Graph 21:163–169

Middletich AE, Sears KH (1985) Blend surfaces for set-theoretic volume modeling systems. Comput Graph 19:161–170

Muraki S (1991) Volumetric shape description of range data using blobby model. Comput Graph 25:227–235

Pasko AA (1988) Conceptual and instrumental tools for a direct method of multidimensional geometric problem solving with a computer. PhD Dissertation, Moscow Engineering Physics Institute, Moscow, Russia

Pasko AA, Savchenko VV (1993) Solid noise in constructive solid geometry, EDUGRAPHICS'93 and COMPUGRAPHICS'93, In: Santa HP (ed) Combined Proceedings, ACM, Lisbon, Portugal, pp 351–357

Pasko AA, Savchenko VV (1994a) Blending operations for functionally based constructive geometry. Set Theoretic Solid Modeling: Techniques and Applications, CSG'94 Conference Proceedings, Information Geometers, Winchester, UK, pp 151–161

Pasko AA, Savchenko VV (1994b) Offsetting operations for functionally based constructive geometry. 6th International Conference on Engineering Computer Graphics and Descriptive Geometry, Tokyo, Japan 1:201–204

Pasko AA, Pilyugin V, Pokrovskiy V (1988) Geometric modeling in the analysis of trivariate functions. Comput Graph 12:457–465

Pasko AA, Adzhiev VD, Prostakov I (1992) Multivariate functions visualization: the inductive approach. Proceedings of the 3rd Eurographics Workshop on Visualization in Scientific Computing, Viareggio, Italy, pp 303–316

Pasko AA, Savchenko W, Adzhiev VD, Sourin A (1993a) Multidimensional geometric modeling and visualization based on the function representation of objects. Technical Report 93-1-008, University of Aizu, Aizu, Japan

Pasko AA, Savchenko VV, Adzhiev VD (1993b) Educational project: Empty Case technology of geometric modeling. EDUGRAPHICS'93 and COMPUGRAPHICS'93, In: Santo HP (ed) Combined Proceedings, ACM, Lisbon, Portugal, pp 6–11

Pilyugin VV, Adzhiev VD, Pasko AA, Sourin AI (1988) Theoretical issues of dynamic and multidimensional geometric modeling with a computer. Problems Informational Syst 5:3–35

Requicha AAG (1980) Representations for rigid solids: theory, methods, and systems. Comput Surveys 12:437–464

Requicha AAG, Rossignac JR (1992) Solid modeling and beyond. IEEE Comput Graph Appl 12:31–44

Ricci A (1973) A constructive geometry for computer graphics. Comput J 16:157–160

Rockwood AP (1989) The displacement method for implicit blending surfaces in solid models. ACM Trans Graph 8:279–297

Rossignac JR, Requicha AAG (1984) Constant-radius blending in solid modeling. Comput Mech Eng 3:65–73

Rvachev VL (1963) On the analytical description of some geometric objects. Reports of the Ukrainian Academy of Sciences 153:765–767

Rvachev VL (1974) Methods of logic algebra in mathematical physics. Naukova Dumka, Kiev

Savchenko VV, Pasko AA (1993) Simulation of dynamic interaction between rigid bodies with time-dependent implicitly defined surfaces. In: Arnold D (ed) Parallel Comput Transputers, IOS Press, Amsterdam, pp 122–129

Savchenko VV, Pasko AA (1994) Parallel polygonization of implicit surfaces on transputers. In: Arabnia H (ed) Transputer Research and Applications 7, IOS Press, Amsterdam, pp 22–30

Savchenko VV, Pasko AA, Okunev OG, Kunii TL (1994) Function representation of solids reconstructed from scattered surface points and contours. Technical Report 94-1-032, University of Aizu, Aizu, Japan

Schmidt MFW (1993) Cutting cubes – visualizing implicit surfaces by adaptive polygonization. Visual Comput 10:101–115

Sclaroff S, Pentland A (1991) Generalized implicit functions for computer graphics. Comput Graph 25:247–250

Shapiro V (1988) Theory of R-functions and applications: a primer, Technical Report CPA88-3, Cornell University, Ithaea, NY

Shapiro V (1994) Real functions for representation of rigid solids. Comput Aided Geom Design 11:153–175

Shapiro V, Vossler DL (1993) Separation for boundary to CSG conversion, ACM Trans Graph 12:35–55

Snyder JM (1992) Generative modeling for computer graphics and CAD. Academic Press, Boston

Snyder JM, Woodbury AR, Fleischer K, Currin B, Barr AH (1993) Interval methods for multi-point collisions between time-dependent curved surfaces. SIGGRAPH'93, Computer Graphics Proceedings, Annual Conference Series 27: 321–334

Sourin AI (1988) Mathematics and software for problems of time dependent geometry. PhD Dissertation, Moscow Engineering Physics Institute, Moscow, Russia

Sourin AI, Pasko AA (1995) Function representation for sweeping by a moving solid. In: Hoffmann C and Rossignac J (eds) Third ACM Symposium on Solid Modeling and Applications, Salt Lake City, USA, ACM, pp 383–391

Wang WP (1984) Solid geometric modeling for mold design and manufacture. Technical Report 44, Cornell university, Ithaca, NY

Warren J (1989) Blending algebraic surfaces. ACM Trans Graph 8:263–278

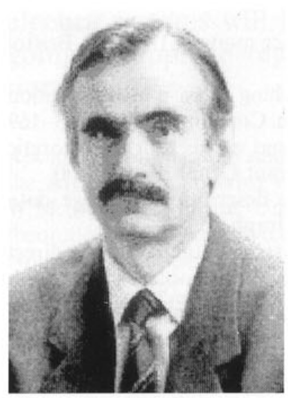
Wyvill G, McPheeters C, Wyvill B (1986a) Data structure for soft objects. Visual Comput 2:227–234

Wyvill B, McPheeters C, Wyvill G (1986b) Animating soft objects. Visual Comput 2:235–242

ALEXANDER A. PASKO is an Assistant Professor at the Shape Modeling Laboratory, Computer Software Department, University of Aizu, Japan. He received his MSc and PhD degrees in Computer Science from the Moscow Engineering Physics Institute (MEPI), Russia, in 1983 and 1988. He was a researcher at MEPI from 1983 to 1992. His research interests include solid modeling, animation, multidimensional visualization, experimental data processing, and computer art. He is a member of ACM, IEEE, and Eurographics Association.

VALERY D. ADZHIEV is a Research Fellow at the Department of Computer Science, University of Warwick, Coventry, UK. He received his MSc and PhD degrees in Computer Science from the Moscow Engineering Physics Institute, Russia, in 1980 and 1992. He was a senior scientist at the Flight Research Institute (Russia) from 1985 to 1993. His research interests include geometric modeling, advanced interactive environments, geometric specification and programming languages, and intelligent CAD.

ALEXEI I. SOURIN received his MS Diploma in Computer Engineering in 1983 from the Moscow Engineering Physics Institute (MEPI), Russia. For five years he has been involved in developing computer graphics systems as an engineer and later as a research assistant at the Department of Physics of Superconductivity at MEPI. In 1988 he finished his dissertation on mathematics and software for problems of time-dependent geometry and received his PhD degree in Computer Science. Since 1993, he has been a lecturer at the Nanyang Technological University, Singapore. He is interested in various branches of computer animation, solid modeling, and virtual reality.



VLADIMIR V. SAVCHENKO is a Professor at the Shape Modeling Laboratory, Computer Software Department, University of Aizu, Japan. He received his MSc in Mechanical Engineering from Moscow Aviation Institute in 1971 and his PhD in Theoretical Mechanics from the Institute of Applied Mathematics in Moscow in 1985. He was Head of the Computational Mechanics Department at the Scientific Research Institute of System Analysis, part of the Russian Academy of Sciences, from 1989 to 1992. His research interests include parallel processing, geometric modeling, computer-aided design, and artificial life.