Free-Form Geometric Modeling by Integrating Parametric and Implicit PDEs

Haixia Du and Hong Qin, *Member*, *IEEE*

Abstract—Parametric PDE techniques, which use partial differential equations (PDEs) defined over a 2D or 3D parametric domain to model graphical objects and processes, can unify geometric attributes and functional constraints of the models. PDEs can also model implicit shapes defined by level sets of scalar intensity fields. In this paper, we present an approach that integrates parametric and implicit trivariate PDEs to define geometric solid models containing both geometric information and intensity distribution subject to flexible boundary conditions. The integrated formulation of second-order or fourth-order elliptic PDEs permits designers to manipulate PDE objects of complex geometry and/or arbitrary topology through direct sculpting and free-form modeling. We developed a PDE-based geometric modeling system for shape design and manipulation of PDE objects. The integration of implicit PDEs with parametric geometry offers more general and arbitrary shape blending and free-form modeling for objects with intensity attributes than pure geometric models.

Index Terms—PDE techniques, geometric modeling, solid models, implicit models, free-form deformation, shape blending.

1 Introduction and Motivation

Geometric modeling is fundamental for visual computing because it provides shape representation and manipulation for geometric objects. Different from surface modeling techniques that are extensively used to define geometric shapes, solid modeling provides a geometrically unambiguous and topologically consistent representation for 3D objects with interior geometry. It greatly enhances existing surface modeling techniques. Popular solid modeling techniques [1], [2] include: constructive solid geometry (CSG), boundary representation (B-rep), cell decomposition, and free-form parametric solids, etc. The CSG approach exploits semi-algebraic sets and Boolean operations on simple primitives, such as cubes, spheres, cylinders, etc., to construct complex solid models. The B-rep technique typically defines a solid object via a set of boundary surfaces with extra topological information. The cell decomposition method usually uses 2D cross-sectional slices or cubical units (e.g., voxels) to approximate complicated solids with hierarchically structured octree schemes. Free-form solid modeling techniques use free-form splines such as B-splines, Hermite splines, and NURBS, to define solid objects that combine the benefits of free-form boundary surfaces and interior geometry in a unified framework.

On the other hand, parametric PDE models define geometric objects as solutions of certain partial differential equations with only a few boundary conditions [3], [4], [5], [6], [7]. In particular, trivariate PDEs can also be used to define parametric solid objects [8], [9]. In comparison with conventional geometric modeling techniques, PDE models have many advantages:

- The behavior of a PDE object is governed by boundary-value differential equations. Geometric models with high-order continuity requirements can be readily defined through high-order PDEs.
- In principle, PDE objects can be reconstructed from a small set of boundary conditions. Their interior information will be automatically recovered by solving given PDEs. Hence, PDE models require fewer parameters than free-form solids.
- In particular, PDE solids have the advantage of several conventional solid modeling techniques, such as spline-based behavior, boundary surface representations, and underlying parameterization for (generalized) cell decomposition in the interior. Therefore, they have the potential to integrate CSG, B-rep, and cell decomposition into a single framework.
- Parametric PDEs offer mapping between parametric and physical space. Hence, such PDEs, especially trivariate PDEs, can provide natural free-form deformation (FFD) operations for embedded objects inside the PDE models.
- PDE objects can unify both geometric and physical aspects for real-world models. Various heterogeneous requirements can be enforced and satisfied simultaneously.

In addition, PDEs are also be used to model implicit solids because implicit models have the advantage of representing arbitrary topological objects as level sets of certain scalar functions [10].

However, both parametric techniques and implicit models have their own strengths and limits. For example, parametric models provide explicit shape descriptions that are missing

- *H. Du is with the Office of High Performance Computing and Communications, National Library of Medicine, National Institutes of Health, 8600 Rockville Pike, Bldg. 38A, Room B1N30, Bethesda, MD 20894. E-mail: haixia.du@gmail.com.*
- *H. Qin is with the Department of Computer Science, Stony Brook University, Room 2426, Computer Science Building, Stony Brook, NY 11794-4400. E-mail: qin@cs.sunysb.edu.*

Manuscript received 6 Apr. 2006; revised 4 Aug. 2006; accepted 6 Oct. 2006; published online 2 Jan. 2007.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-0040-0406.
Digital Object Identifier no. 10.1109/TVCG.2007.1004.

from implicit representations, but parametric techniques have difficulties with shape blending and collision detection, which, in contrast, can be easily achieved by implicit functions. Therefore, a unified approach offering advantages of both categories will be more desirable for arbitrary geometric modeling purposes. Moreover, the previous mentioned techniques mostly focus on pure geometric models. To simulate real-world objects, it's better to incorporate material and physical properties such as density into geometric representations. Because many material attributes can be synthesized by scalar values, implicit functions will be ideal candidates to model these physical properties. Therefore, by integrating implicit models with geometric representations, one can possibly achieve more realistic simulation of real-world models. Since the PDE formulations for parametric and implicit PDE solids have similar forms, we propose an integrated PDE modeling framework which unites parametric and implicit PDEs so that it can simultaneously model geometric and material attributes including shape and intensity distributions for geometric objects. The PDE formulation is an integration of trivariate parametric PDEs that govern the geometric solid shape which can further be treated as the FFD deforming space and implicit elliptic PDEs that model its scalar intensity field. It can be viewed as a 4D formulation that extends the traditional 3D geometry by another dimension for possible physical material properties.

Our framework provides powerful modeling techniques such as free-form and direct manipulation, arbitrary shape blending, and intensity-based deformation for geometric objects with parametric geometry and implicit intensity distributions. The integration of implicit and parametric PDE techniques can inherit modeling advantages of both techniques while compensating each other's limitations. It offers an intuitive shape design and manipulation environment to model geometric objects with arbitrary topology.

2 Prior Work

Parametric PDEs were first employed by Bloor et al. for surface blending, free-form surface design, solid modeling, functional design, and interactive surface sculpting [4], [5], [6], [8]. The geometric objects are defined as solutions of given parametric PDEs with very few parameters, such as boundary-value conditions and blending coefficients associated with the equations. In the past several years, we [11], [12], [13] presented a physics-based PDE surface modeling technique that facilitated direct manipulation and interactive sculpting of physics-based PDE surfaces and displacements. These techniques allow PDE surfaces of diverse types of topology to be defined through general, flexible boundary constraints and operations such as trimming, merging, manipulating of isoparametric curves and/or arbitrary curve networks, editing user-specified subsurfaces, etc. The PDE techniques can also model surface displacements to manipulate existing parametric surface objects and facilitate the data exchange between PDE surfaces with other parametric surface models.

Later on, by expanding the coverage of parametric PDE techniques to trivariate solids, we [9] employed trivariate elliptic PDEs to model parametric solid geometry with physical properties. PDE solids can be defined by more flexible boundary constraints, including boundary surfaces or a set of boundary curve networks. Our PDE solid modeling system provides solid manipulations through boundary surface sculpting, as well as local control and trimming operations using simple CSG tools and user-specified data sets to obtain arbitrary topological shapes. The interactive solid sculpting and manipulation are accomplished by integrating PDE solids with physics-based modeling techniques with intuitive editing toolkits. Furthermore, since a trivariate parametric PDE provides a mapping between the parametric and physical space, it can be used as an FFD scheme.

In essence, an FFD scheme involves a mappixng from the 3D parametric domain to the physical domain through a certain trivariate function. The trivariate function provides a parameterization for the shape to define its position in the space. When the space is deformed, the embedded shape is deformed according to its parameterization. FFD can be applied to arbitrary geometric objects since the embedding space is independent of the geometric representation and topological structure of embedded targets. There are various FFD schemes that use splines [14], [15], [16], subdivision volumes [17], implicit functions [18], [19], etc. Despite of flexible topological coverage of geometric models, FFD has difficulty in supporting direct manipulation of solid objects in general.

Another way to define objects with arbitrary topology is using implicit functions. To take advantage of both PDE techniques and implicit models, we [20], [21] proposed using elliptic implicit PDEs for arbitrary shape design, reconstruction, recovery, blending, and manipulation.

To further explore the potential of PDE techniques for arbitrary shape modeling with features and functionalities, we propose an integrated formulation that incorporates parametric and implicit PDEs to model four-dimensional objects containing three-dimensional geometric shape information plus a one-dimensional material attribute such as intensity distribution. It offers more general direct and free-form modeling functionalities for objects. This unified framework will forge ahead toward the realization of the full potential of PDE techniques in geometric modeling.

3 Integrated PDE Formulation for Geometric Objects with Intensity

In order to model both the geometry and intensity material attributes of an object simultaneously, we propose a unified formulation by integrating the trivariate parametric and implicit PDEs into an elliptic PDE defined over parametric u, v, w space. Equation (1) is a fourth-order formulation:

$$\left(\mathbf{a}^2\frac{\partial^2}{\partial u^2}+\mathbf{b}^2\frac{\partial^2}{\partial v^2}+\mathbf{c}^2\frac{\partial^2}{\partial w^2}\right)^2\mathbf{P}(u,v,w)=\mathbf{0}, \qquad (1)$$

where $\mathbf{P}(u,v,w)=[\mathbf{X}(u,v,w),d(u,v,w)]^\top$, $\mathbf{X}(u,v,w)=[x(u,v,w)\quad y(u,v,w)\quad z(u,v,w)]^\top$ defines PDE solid coordinates in 3D physical space, $d(u,v,w)$ represents corresponding intensity field, $\mathbf{a}(u,v,w)$, $\mathbf{b}(u,v,w)$, and $\mathbf{c}(u,v,w)$ are blending coefficient functions that control contributions from u, v, w directions. To provide more degrees of freedom when modeling integrated PDE objects, we allow blending

coefficient functions to have different controls on geometry and intensity attributes, i.e., the coefficient functions can be defined as follows:

$$\mathbf{a}(u,v,w)=\begin{pmatrix}\alpha_g(u,v,w) & 0\\ 0 & \alpha_d(u,v,w)\end{pmatrix},$$
$$\mathbf{b}(u,v,w)=\begin{pmatrix}\beta_g(u,v,w) & 0\\ 0 & \beta_d(u,v,w)\end{pmatrix},$$
$$\mathbf{c}(u,v,w)=\begin{pmatrix}\gamma_g(u,v,w) & 0\\ 0 & \gamma_d(u,v,w)\end{pmatrix}.$$

Furthermore, we also use a second-order equation (2) for better time performance with less continuity requirements because, although higher order PDEs provide higher geometric continuity, they also require a longer time to obtain a solution than lower order equations.

$$\left(\mathbf{a}^2\frac{\partial^2}{\partial u^2}+\mathbf{b}^2\frac{\partial^2}{\partial v^2}+\mathbf{c}^2\frac{\partial^2}{\partial w^2}\right)\mathbf{P}(u,v,w)=\mathbf{0}. \tag{2}$$

Note that there are special cases that the integrated equations can reduce to either parametric geometric models or implicit intensity formulations. For example, when the intensity distribution across the entire parametric space has a constant value, (1) is reduced to a trivariate parametric equation only defining PDE solid geometry:

$$\left(\alpha_g^2\frac{\partial^2}{\partial u^2}+\beta_g^2\frac{\partial^2}{\partial v^2}+\gamma_g^2\frac{\partial^2}{\partial w^2}\right)^2\mathbf{X}(u,v,w)=\mathbf{0}. \tag{3}$$

Similarly, when the mapping between the parametric space of u, v, w and the physical space of x, y, z becomes identical, (1) becomes the implicit PDE for the intensity field similar to the PDE introduced in [20], [21]:

$$\left(\alpha_d^2\frac{\partial^2}{\partial u^2}+\beta_d^2\frac{\partial^2}{\partial v^2}+\gamma_d^2\frac{\partial^2}{\partial w^2}\right)^2 d(u,v,w)=0. \tag{4}$$

With this type of formulation, we can model solid shape geometry and material properties such as intensity distribution in a single framework which provides modeling advantages of both parametric models and implicit functions.

3.1 Boundary Conditions for Integrated PDE Solids

The integrated PDE formulations are elliptic PDEs that define the geometry and intensity of PDE solid objects with the corresponding boundary conditions. The parametric domain is a cubic space of u, v, w covered by six boundary surfaces. Usually, information on these surfaces is provided to define a PDE solid, especially the geometric shape. The resulting geometric object obtained by solving the equation will interpolate these boundary surfaces and recover the interior region surrounded by these surfaces. On the other hand, we allow different types of boundary constraints to define the intensity distribution because it is not the location but the intensity value at the location that determines the implicit shape. In this paper, we restrain u, v, w to vary between 0 and 1 because reparametrization from arbitrary span $[a,b]$ can be easily obtained.

The six geometric boundary surfaces defining three surface pairs bounding the PDE solid are in the form of (5):

$$\begin{aligned}&\mathbf{X}(0,v,w)=\mathbf{U}_0(v,w),\\ &\mathbf{X}(1,v,w)=\mathbf{U}_1(v,w),\\ &\mathbf{X}(u,0,w)=\mathbf{V}_0(u,w),\\ &\mathbf{X}(u,1,w)=\mathbf{V}_1(u,w),\\ &\mathbf{X}(u,v,0)=\mathbf{W}_0(u,v),\\ &\mathbf{X}(u,v,1)=\mathbf{W}_1(u,v).\end{aligned} \tag{5}$$

These surfaces share corresponding boundary curves with each other. More generally, we can give a set of surfaces along the three parametric directions inside the solid space, like cross-sectional lattices to define a PDE solid. This type of generalized boundary constraints has the form of $\mathbf{X}(u_i,v,w)=\mathbf{U}_i(v,w)$, $\mathbf{X}(u,v_j,w)=\mathbf{V}_j(v,w)$, or $\mathbf{X}(u,v,w_k)=\mathbf{W}_k(v,w)$.

Furthermore, because any isoparametric surface with a given value of u, v, or w inside the solid geometry is a PDE surface, we also use a bivariate PDE surface formulation with the same order of the trivariate equation to model boundary surfaces, which can be derived from a set of boundary curves. Therefore, a PDE solid geometry can also be defined by a set of boundary curve networks. For example, the PDE boundary surface formulation for a fourth-order PDE solid can have the following form:

$$\left(a_1^2\frac{\partial^2}{\partial u_1^2}+a_2^2\frac{\partial^2}{\partial u_2^2}\right)^2\mathbf{X}(u_1,u_2)=\mathbf{0}, \tag{6}$$

where $(u_1,u_2)\in\{(u,v),(u,w),(v,w)\}$. The boundary surface formulation for second-order PDE solids has a similar form:

$$\left(a_1^2\frac{\partial^2}{\partial u_1^2}+a_2^2\frac{\partial^2}{\partial u_2^2}\right)\mathbf{X}(u_1,u_2)=\mathbf{0}. \tag{7}$$

The boundary curves to define a PDE solid can be defined as follows:

$$\begin{aligned}&\mathbf{X}(u,v_i,w_j)=\mathbf{U}_{ij}(u),\\ &\mathbf{X}(u_k,v,w_l)=\mathbf{V}_{kl}(v),\\ &\mathbf{X}(u_r,v_s,w)=\mathbf{W}_{rs}(w).\end{aligned} \tag{8}$$

Such general and arbitrary boundary conditions give users more flexibility to design solid shapes with fewer parameters and are capable of modeling solids that must pass through a set of curves as general constraints.

Besides the boundary constraints for PDE solid geometric shapes, we need to define the corresponding intensity distributions. The simplest way is to set the intensity values to be constant throughout the solid space. Then, the defined PDE solids can be treated as pure geometric objects. Users can also use certain implicit functions to assign intensity values for the solids. In addition, the intensity material distribution can be defined by boundary constraints. Besides the traditional boundary conditions that contain information at boundaries of the parametric working space, we also allow generalized implicit boundary constraints such as iso-surfaces/curves and volumetric data sets, etc. Refer to [21] for more details about boundary conditions for implicit PDEs.

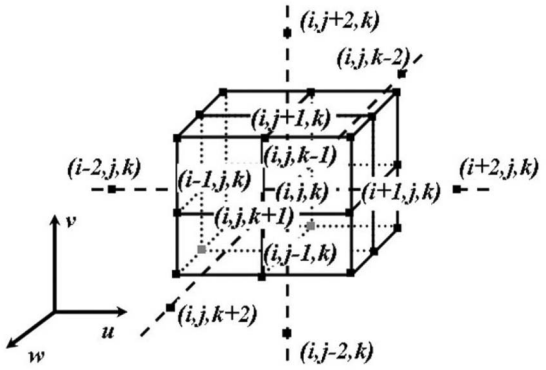

Fig. 1. The point discretization of part of a PDE solid.

3.2 Numerical Simulation

There are various techniques that can be used to solve parametric PDEs. Although analytic techniques offer accurate and fast solutions for PDEs with certain boundary conditions, they cannot provide satisfying results from more general constraints. In contrast, numerical techniques can guarantee solutions for PDEs, especially when additional constraints are enforced. Among many matured numerical techniques, we resort to the finite-difference method (FDM) and iterative techniques for linear equations because they are simple, easy to implement, and suitable for various general and additional constraints in the solid working space. To improve the system performance, we also employ a multigrid subdivision method starting from coarse resolution and refining to finer grids.

The FDM first divides the parametric space into discrete grids along parametric directions, then, for each grid point, the partial derivatives in the equation are replaced by finite difference approximations. By collecting the finite difference equations at the grid points, we can transform a continuous PDE into an algebraic equation system. This system can then be solved numerically either through a direct procedure or an iterative process for an approximate solution.

We use the central-difference scheme to approximate partial derivatives in the trivariate PDE by dividing the u, v, w domain into l, m, and n discretized points (Fig. 1), respectively. Using (3) as an example, given a a grid point (i, j, k), the second and fourth-order partial derivatives of $\mathbf{X}$, $\frac{\partial^2 \mathbf{X}_{i,j,k}}{\partial u^2}$, $\frac{\partial^4 \mathbf{X}_{i,j,k}}{\partial u^4}$, and $\frac{\partial^4 \mathbf{X}_{i,j,k}}{\partial u^2 \partial v^2}$ at $\{i, j, k\}$ can be approximated by:

$$\frac{\partial^2 \mathbf{X}_{i,j,k}}{\partial u^2} = \frac{\mathbf{X}_{i-1,j,k} + \mathbf{X}_{i+1,j,k} - 2\mathbf{X}_{i,j,k}}{(\Delta u)^2},$$

$$\frac{\partial^4 \mathbf{X}_{i,j,k}}{\partial u^4} = \frac{\mathbf{X}_{i-2,j,k} + \mathbf{X}_{i+2,j,k} - 4\mathbf{X}_{i-1,j,k} - 4\mathbf{X}_{i+1,j,k} + 6\mathbf{X}_{i,j,k}}{(\Delta u)^4},$$

$$\frac{\partial^4 \mathbf{X}_{i,j,k}}{\partial u^2 \partial v^2} = \frac{\frac{\mathbf{X}_{i-1,j-1,k} + \mathbf{X}_{i-1,j+1,k} + \mathbf{X}_{i+1,j-1,k} + \mathbf{X}_{i+1,j+1,k}}{(\Delta u)^2 (\Delta v)^2} + \frac{-2\mathbf{X}_{i-1,j,k} - 2\mathbf{X}_{i+1,j,k} - 2\mathbf{X}_{i,j-1,k} - 2\mathbf{X}_{i,j+1,k} + 4\mathbf{X}_{i,j,k}}{(\Delta u)^2 (\Delta v)^2}}{}.$$

Other partial derivatives along the v and w directions can be computed similarly.

After replacing the partial derivatives by their finite-difference approximations at discretized grid points, (3) can be rewritten as:

$$\mathbf{AX} = \mathbf{z}, \tag{9}$$

where $\mathbf{A}$ is a discretized differential operator in $(l \times m \times n) \times (l \times m \times n)$ matrix form and each row in $\mathbf{A}$ consists of coefficients of the difference equation for its corresponding grid point. $\mathbf{A}$ is also controlled by the coefficient functions.

$$\mathbf{X} = [\, \mathbf{X}_{0,0,0} \quad \mathbf{X}_{0,0,1} \quad \cdots \quad \mathbf{X}_{l-1,m-1,n-1} \,]^\top$$

and

$$\mathbf{z} = \left[\mathbf{z}_0 \quad \mathbf{z}_1 \quad \cdots \quad \mathbf{z}_{(l-1)\times(m-1)\times(n-1)} \right]^\top.$$

More detailed information about $\mathbf{A}$ can be found in [21].

Similarly, (4) can be approximated by

$$\mathbf{Bd} = \mathbf{e}, \tag{10}$$

as well as the second-order equations.

By putting these together, we get an approximation for (1):

$$\mathbf{MP} = \mathbf{f}. \tag{11}$$

Note that a PDE solid is open along all of the u, v, and w directions, so the computation of partial derivatives near to the six boundary surfaces requires a forward/backward difference approximation scheme. Arbitrary boundary conditions can be easily enforced using FDM by formulating the boundary conditions as linear equations and incorporating them into the finite difference equation system. Note that, despite certain combinations of constraint imposition shown in our experiments, in general this type of elliptic PDEs allow boundary conditions to be explicitly formulated in arbitrary form. This permits designers to choose (various) constraints based on diverse designing tasks.

3.3 Multigrid Iterative Approximation

The approximate difference equations form an algebraic equation system which can be easily solved by either direct methods or iterative methods and is suitable for parallel computing. For high resolution of domain discretization, the number of difference equations will increase dramatically, which indicates iterative solvers are more realistic choices than direct methods.

The iterative methods make use of the structure of the sparse matrix on the left-hand side of the finite-difference equation system. Using (9) as an example, the matrix $\mathbf{A}$ is split into two parts

$$\mathbf{A} = \mathbf{A}_d - \mathbf{A}_r, \tag{12}$$

where $\mathbf{A}_d$ consists of the diagonal elements of $\mathbf{A}$ and zeros everywhere else, $\mathbf{A}_r$ is the remainder. Then, (9) becomes

$$\mathbf{A}_d\mathbf{X} = \mathbf{A}_r\mathbf{X} + \mathbf{z}. \tag{13}$$

The iterative methods start from choosing an initial guess $\mathbf{X}^{(0)}$ and then solving the equations successively by iterating $\mathbf{X}^{(s)}$ from

$$\mathbf{A}_d\mathbf{X}^{(s)} = \mathbf{A}_r\mathbf{X}^{(s-1)} + \mathbf{z}. \tag{14}$$

Given boundary conditions, one can compute an initial guess of the PDE solid by linear interpolations based on given constraints. The iteration will stop at $\mathbf{X}^{(s)}$ for an approximate solution when the difference between consecutive iterating results $\mathbf{X}^{(s)}$ and $\mathbf{X}^{(s-1)}$ is less than a threshold.

Certain variants of iterative techniques exist [22]. In this paper, we employ the Gauss-Seidel iteration, which applies the iteration result at a grid point to the right-hand side of (14) as soon as it becomes available. To further speed up the converging rate of Gauss-Seidel iteration, the error factor characterized by the difference between the approximation and the real solution is considered. This leads to the method of Successive Over-Relaxation (SOR) iteration.

The large number of sample points of a PDE surface/solid results in the slow convergence of iterative techniques. The multigrid approximation based on simple subdivision schemes is used to improve the computation performance. When solving the PDE solid geometry, since there are two types of boundary conditions, i.e., curve network and surfaces, different multigrid approximation schemes are employed to handle these two types of boundary constraints, respectively.

If boundary conditions for solid geometry are curve networks, boundary surfaces shall first be computed based on the PDE surface formulation. This can be done by starting with a small number of sample points at the coarsest resolution of the PDE boundary surfaces and the approximate PDE surfaces can be easily derived after several iterations. Then, the corresponding PDE solid is solved. Users can refine the coarse boundary mesh through subdivision and use the new subdivided mesh as an initial guess for subsequent iteration steps. The finer resolution is then computed iteratively to achieve a more accurate and smoother solution of boundary PDE surfaces as well as the PDE solid. For further refinement over the finest resolution, the multigrid approximation starts with the up-sampling of all boundary curves through the use of a four-point interpolatory subdivision scheme [23] in order to guarantee the smoothness requirement of refined curves.

If boundary conditions are connected surfaces, the approximation scheme should be slightly modified. The process starts with the coarsest resolution of boundary surfaces through down-sampling to obtain a coarse solution of the solid. Then, during the refining process, more points are sampled over boundary surfaces until it reaches the finest resolution. After that, the subdivision process may continue to reach even finer resolution. In this scenario, the given boundary surfaces are considered as constrained PDE surfaces, requiring four curves as boundary conditions and the originally defined surface sample points as hard constraints. Then, the four-point interpolatory subdivision scheme is used to subdivide boundary curves and compute unknown surface points by solving the surface PDE subject to the subdivided boundary curves and original surface points as hard constraints.

4 Interactive Editing Toolkits for Free-Form Geometric PDE Objects

Our PDE modeling system provides direct manipulations and regional operations for PDE solid models as well as free-form deformation of arbitrary objects by embedding them into PDE solids.

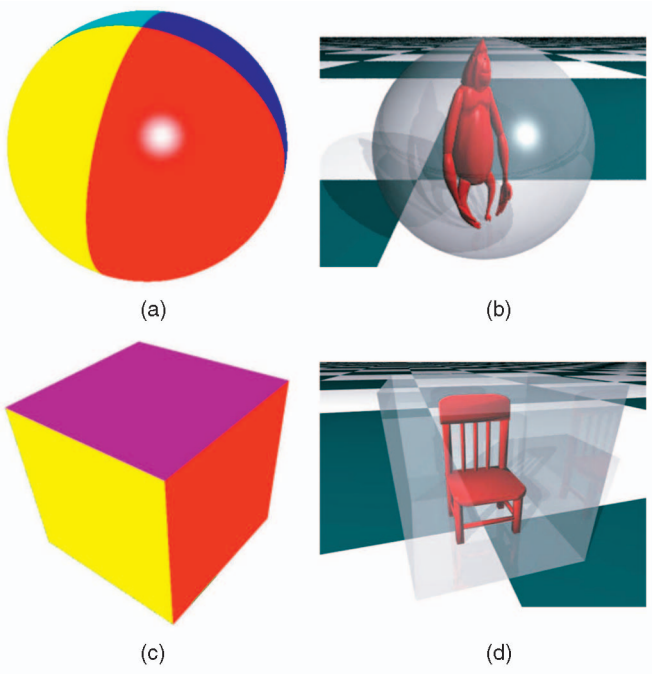

Fig. 2. PDE solids generated from given boundary surfaces. (a) and (c) are two sets of boundary surfaces; (b) and (d) are the corresponding PDE solids (displayed using transparent color) subject to (a) and (c) with embedded data sets, respectively.

4.1 Defining PDE Solid Geometry

The PDE solid geometry can be defined by either boundary surfaces or boundary curves. At first, users must specify the boundary type, i.e., predefined surfaces, or connected boundary curve network for the PDE solid.

For predefined boundary surfaces, the system can obtain the already defined surfaces that form the outline of PDE solid. Then, using these surfaces as boundary conditions, the system can recover the interior information of the PDE solid bounded by these surfaces as the solution of (9). Fig. 2 shows two examples. We put some data sets in the PDE solids to illustrate their inside structures.

If a curve network is used as boundary conditions, the PDE solid can be generated through two steps: first, generating the boundary surfaces from given curves by solving (6) or (7); second, solving (9) for the corresponding PDE solid using the results from the previous step as boundary constraints. Because every two neighboring boundary surfaces share one boundary curve, the shared curves need to be defined in the curve network. The boundary surfaces can be even defined more precisely by adding more curves as boundary conditions. Fig. 3 shows examples of using curve networks to define PDE solids.

4.2 Boundary Manipulation

Users can modify the global shape of a PDE solid through boundary manipulations. Our system permits users to directly modify boundary surfaces, then eventually deform the PDE solid. To modify a PDE solid through boundary conditions, users first select a boundary surface to be edited, then use a set of sculpting toolkits that can manipulate points/curves/regions on a PDE surface to modify the selected boundary surface. Details about PDE surface sculpting can be found in [13]. Fig. 4 has two examples of boundary surface manipulation with curve sculpting.

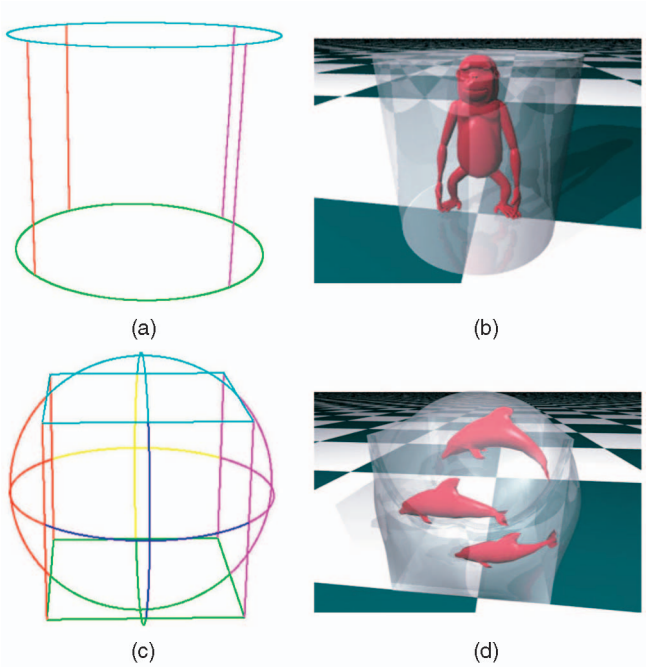

Fig. 3. Examples of PDE solids subject to boundary curve networks. (a) Coons-like boundary curves, (b) the corresponding PDE solid, (c) Gordon-like boundary conditions, and (d) the PDE solid subject to (c). The PDE solids are displayed using transparent color with embedded data sets.

4.3 Direct Solid Manipulations

One advantage of PDE solids is that the solid interior is controlled by PDEs without the need for specification on interior information. PDE solids provide an integrated scheme that not only expands the B-rep method to cover the solid interior, but also supports Boolean operations associated with CSG models. More importantly, with a finite-difference scheme, users can deform the interior of a PDE solid by enforcing additional constraints inside the solid without changing boundary conditions. This can be done by replacing several equations in (9) by equations obtained from additional constraints to form a constrained system:

$$\mathbf{A}_c\mathbf{X} = \mathbf{z}_c. \tag{15}$$

Our system provides a set of interactive operations inside a PDE solid including trimming, local region sculpting, and deformation.

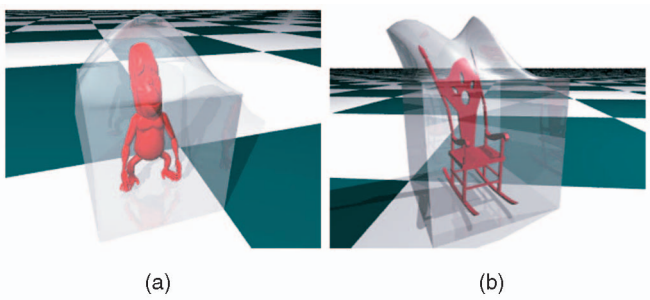

Fig. 4. Modifying PDE solids via curve constraints of boundary surfaces.

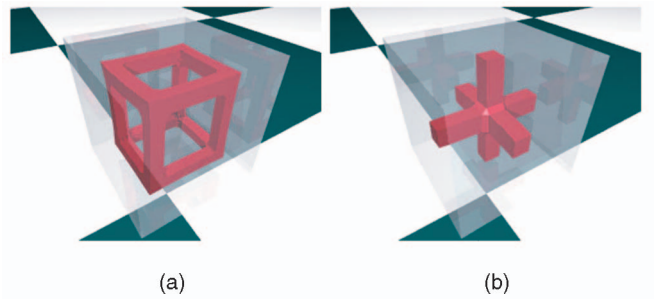

Fig. 5. Examples using CSG operations to trim PDE solids. The trimmed parts are shown in red covered by transparent original solids.

4.3.1 Solid Trimming

One of the disadvantages of parametric solids is that it is difficult to model objects of arbitrary topology. According to the idea of CSG models, the trimming operation offers an alternative way to model objects with irregular shape. The system provides trimming functionalities on a PDE solid for sculpting of arbitrary topological shapes. First, users can select regions of interest by specifying the parametric coordinates of the boundary of the region in a pop-up dialog (like the one shown in Fig. 17). Then, they can indicate removing material from the PDE solid either inside or outside those regions. Furthermore, simple shape primitives, such as sphere, cube, or cylinder, can be placed at any position inside the parametric domain as trimming tools. Again, users just need to specify the type, size, and position, and Boolean operation type for the desired tool, then they can move the tool along the u, v, or w directions using keypads and all of the regions covered by the navigating path will be chosen/discarded according to the specified Boolean operations. Such tools allow the CSG construction of complex objects based on PDE solids. Fig. 5 shows trimming examples.

4.3.2 Geometric Free-Form Deformation

Because the trivariate PDE solid formulation provides a mapping between the parametric space and physical space (illustrated in Fig. 6), it's straightforward to use the PDE solid for FFD. The parametric space can be viewed as the original space, and the PDE solid will be the deforming space bounded by the boundary constraints. Given an arbitrary data set, its original coordinates can be viewed as

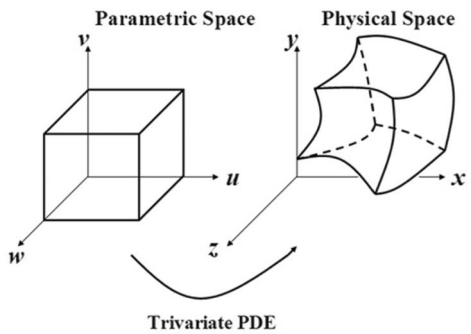

Fig. 6. PDE solid geometry from the parametric space to the physical space.

Fig. 7. Free-form deformation based on PDE solids. In each figure, the object on the left is obtained by embedding it into a PDE cube and the object on the right is obtained from a PDE sphere.

parametric coordinates after embedding it into the PDE parametric domain, then, by mapping the parametric space to the PDE solid space, it can be deformed according to the shape of the PDE solid. The mapping of embedded data sets to different PDE solids will result in different deformed shapes. In essence, this is analogous to the principle of FFD, where the transformation between the parametric space to the physical space is governed by an elliptic PDE. The free-form deformation based on PDE solids can greatly expand the coverage of PDE solid applications, making it possible to obtain PDE-governed free-form modeling for arbitrary topological objects. Fig. 7 shows some examples.

The PDE-governed FFD is different from other FFD techniques because the deformed space is a PDE solid whose interior is governed by the PDE. It allows both global indirect and local direct modification and deformation. We will further discuss this issue in Section 6.3.

4.3.3 Local Region Manipulation

Traditional PDE models only support boundary manipulations which lead to global deformation throughout the entire model. It is more desirable to have local editing functionalities on arbitrary interior regions. This can be easily done by using the finite-difference solver for PDE solids. Our system provides a set of toolkits that allow designers to specify any local regions of a PDE solid, and only enforces deformations either inside or outside selected regions. Users can define the region of interests by: 1) interactively specifying a region in $[u, v, w]$ domain, 2) employing some basic CSG-based tools such as spheres and cubes to navigate the entire parametric domain to define the region of interest, or 3) embedding data sets in the PDE solid space in order to define the particular region. Subsequently, any changes within the region will not propagate to outside areas. The localized deformation can be achieved easily because only those equations corresponding to the points of specified regions will be solved. In principle, all additional constraints regarding geometric properties at selected locations can be viewed as certain types of local deformation. Fig. 8 shows examples of local deformation.

5 Intensity-Based Free-Form Modeling

In contrast to parametric modeling techniques, implicit models offer a different way to define geometric objects by

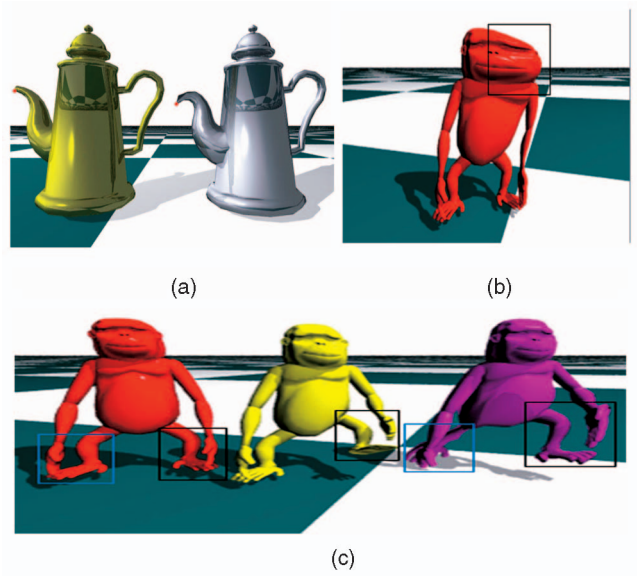

Fig. 8. Direct modification of embedded objects in PDE solids. (a) Directly modifying a point on an embedded object in the PDE solid, (b) a deformed object obtained by moving a specified region inside the PDE solid, and (c) the deforming sequence of objects by rotating selected PDE solid regions. The red point in (a) and rectangles in (b) and (c) show the selected parts for modification.

level sets of scalar intensity field instead of explicit locations. Comparing with parametric techniques, implicit models have several unique properties such as free of parametric correspondence, easy collision detection, etc. Since we incorporate the implicit PDE into our integrated formulation, we can take advantage of implicit models and obtain more free-form modeling features for arbitrary objects with intensity attributes. The combination of parametric and implicit modeling based on PDEs can potentially be used for arbitrary modeling and deformation of objects with material properties.

5.1 Initialization

For intensity-based free-form deformation, the system needs an initial PDE solid space with both geometry information and intensity distribution throughout the space. The PDE solid geometry can be defined using initialization techniques introduced in previous sections. As for the initial intensity distribution, it can be an arbitrary intensity function defined over the parametric domain to maximize the modeling potential of the implicit PDE. In particular, one possible choice to initialize the intensity field is using implicit PDEs to calculate intensity distribution of the working space based on embedded data sets, which unifies the geometric and intensity properties for the PDE solid. Other predefined intensity functions or volumetric data sets can also be used as initial intensity distributions. Fig. 9 shows some examples. With finite-difference techniques, intensity attributes can be directly manipulated using certain implicit PDE modeling toolkits introduced in [20], [21]. When the intensity distribution is obtained by solving implicit PDEs using assigned intensity values on embedded objects in the PDE solid space, shape blending operations can be easily achieved through smooth intensity blending by implicit PDE techniques.

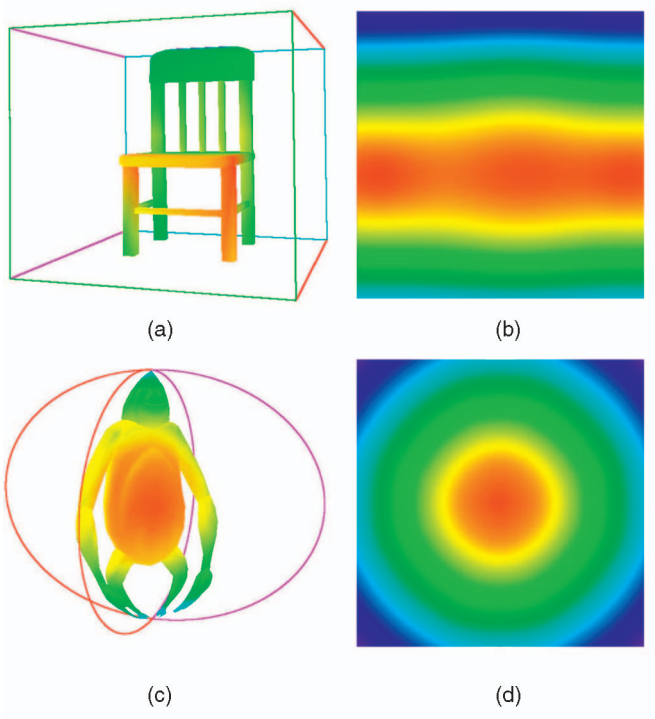

Fig. 9. Examples of intensity initialization of PDE solids. The objects (a) and (c) are embedded in the PDE solid working space with a color map of intensity distribution; (b) and (d) are cross-section views of intensity distributions in the parametric space from the w direction.

5.2 Isosurface Deformation

From the implicit modeling point of view, the integrated solid can also be treated as a deformed implicit space. After the initialization of intensity field for a PDE solid working space, shape deformation related to both intensity and geometry can be obtained. If the intensity distribution represents certain implicit shape, an isosurface of this shape can be generated using the Marching Cubes [24] at any user specified intensity value in the parametric domain. The isosurface can then be treated as an embedded data set for the parametric PDE solid and all of the editing toolkits for parametric PDE solids can be employed for further direct sculpting and free-form deformation for the isosurface. Using this feature, the system provides geometric FFD and direct manipulation for implicit objects. Fig. 10 shows an example.

5.3 Free-Form Shape Blending and Deformation

Shape blending between arbitrary geometric objects are not easy for explicit models because it is hard to construct correspondence between the blending parts. However, the implicit PDE model provides a natural way to blend implicit objects by embedding objects into the implicit PDE working space. Arbitrary shape blending can be easily achieved by the integration of PDE solid geometry and implicit PDEs. Moreover, it can unify shape blending based on implicit PDEs and PDE-based FFD for more flexible shape blending and deformation of arbitrary objects.

To blend arbitrary geometric shapes, the system first constructs an embedding geometric space for each shape to be blended and calculates intensity distributions for the embedding spaces by implicit PDE techniques introduced

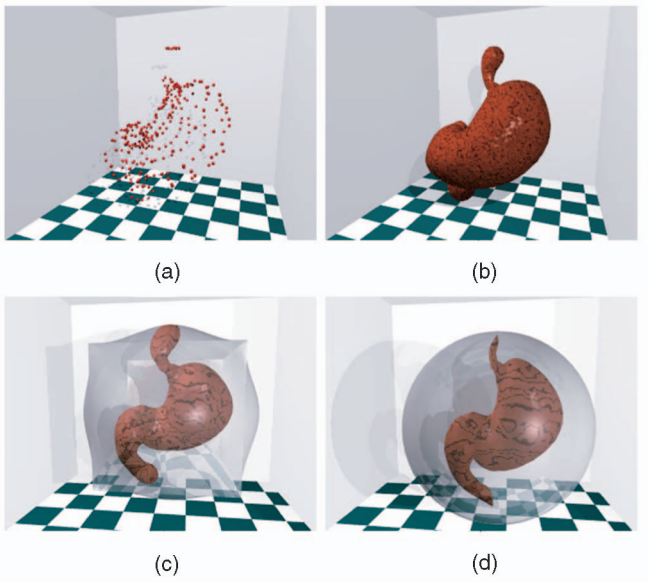

Fig. 10. An example for isosurface deformation. (a) A set of scattered points, (b) implicit isosurface obtained from (a), and (c) and (d) are deformed isosurfaces in different PDE solids.

in [20], [21]. Then, the embedding geometric spaces and intensity distributions can be used as boundary constraints for the blended PDE solid geometry and intensity distribution, respectively. Solving (11) with these boundary constraints will result in a single geometric PDE solid blending the original shapes with a smoothly blended intensity distribution for the entire solid working space. The blended intensity field will provide a smoothly blended intensity transition between original shapes. With any specific isovalue, an isosurface for the blended shape can be reconstructed accordingly. At the same time, the constructed PDE solid obtained from the geometric embedding spaces as boundary constraints offers the blended shape geometry (Fig. 11). Moreover, because blending operations are performed based on intensity distributions in the parametric domain, the system can provide users with different blended shapes for objects embedded in deformed PDE solid working space. This allows users to obtain shape blending and deformation at the same time. It allows more freedom of shape manipulation for objects with arbitrary topology. Refer to Fig. 12 and Fig. 13 for examples.

5.4 Intensity-Based Shape Deformation

The geometric shape of an embedded object in the PDE solid can be deformed by modifying the associated intensity distribution. When modifying the intensity distribution, intensity values of the embedded object will be changed accordingly. Because of the correspondence between intensity values and geometric coordinates, to preserve their original intensity values in the modified intensity field, vertices on the object will follow the intensity modifications to new locations in the working space, which will deform the object's geometric shape. We incorporate the implicit PDE modeling toolkits introduced in [21] into the system to change the intensity values of the working space in order to deform embedded objects. The procedure is as follows: First, by initializing the intensity field of the PDE solid space, the intensity values of an embedded data set are

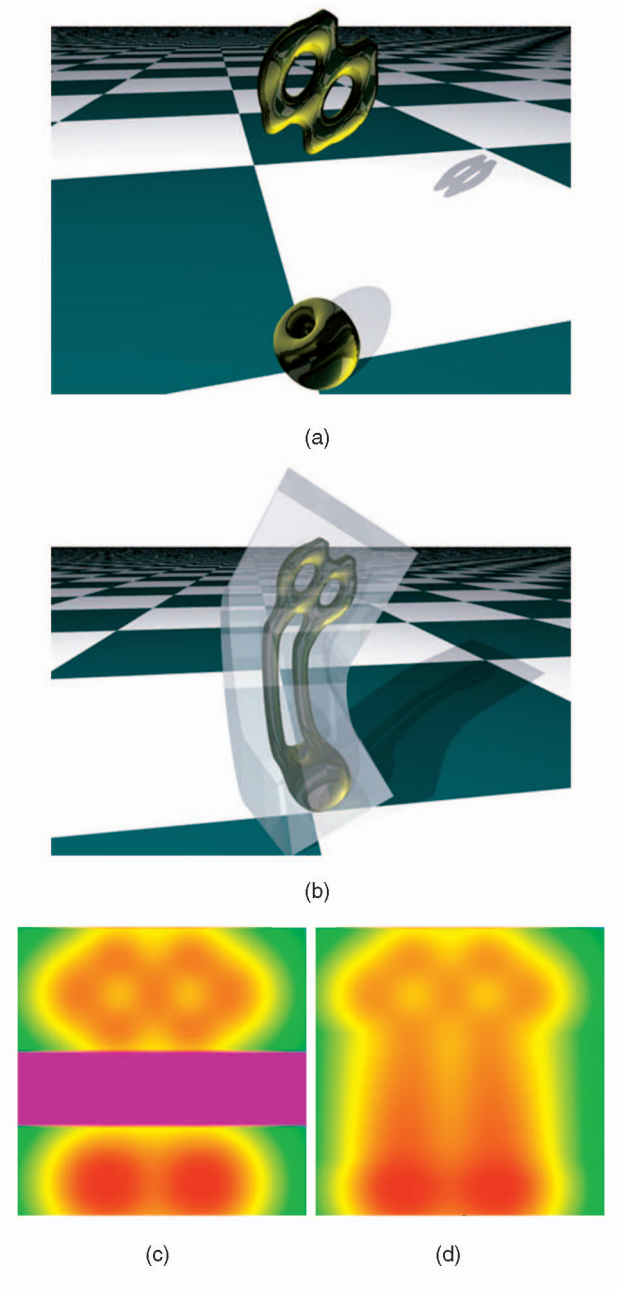

Fig. 11. An example for shape blending by integrated PDE solid with intensity. (a) Two objects to be blended, (b) the blended object embedded in the reconstructed PDE solid, and (c) and (d) are cross-section views of intensity distributions of (a) and (b) along the w direction in the parametric domain, respectively.

obtained. Second, users can modify the intensity distribution of the solid space and new intensity values of the data set as well as the gradient information are calculated according to the modified intensity field. Third, to preserve the original intensity values of the data set, vertices on the data set are allowed to move along their intensity gradient directions to new locations that have their original intensity values in the modified intensity field. As a result, the geometric shape of the embedded object is deformed. The system allows users to change the intensity distribution of the working space locally in selected regions and keep intensity values of other parts untouched. This will provide local implicit deformation of the object. The initial intensity

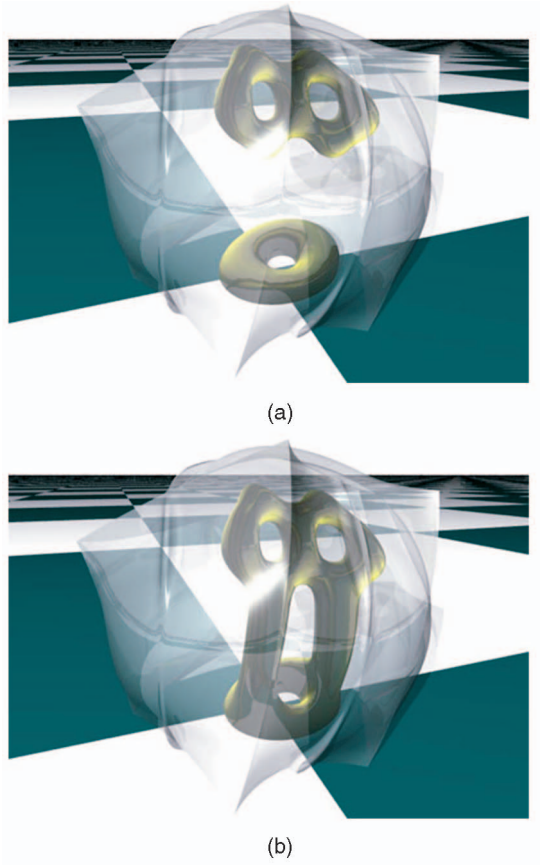

Fig. 12. An example of PDE-based free-form shape blending and deformation. (a) A set of embedded objects to be blended in a PDE solid and (b) the blending result.

distribution of the working space can either be constructed from the embedded data set using implicit PDE techniques or obtained from an arbitrary implicit function for arbitrary shape deformation. Fig. 14 and Fig. 15 have corresponding shape deformation examples of these two cases. In Fig. 15, the implicit function has the following form:

$$d(u,v,w) = e^{-(\sigma_u(u-u_0)^2+\sigma_v(v-v_0)^2+\sigma_w(w-w_0)^2)}.$$

6 System Structure and Results

The integrated PDE-based geometric modeling system offers PDE-based solid shape design from geometric boundary surfaces or curve network and arbitrary shape blending, deformation, and direct manipulation based on objects' geometric and intensity attributes. It allows users to manipulate PDE objects by enforcing various local/global constraints on geometry and intensity properties via boundary conditions and interior operations. Fig. 16 illustrates the outline of the system architecture.

6.1 Modeling Toolkits

We developed a set of direct manipulation toolkits for solid objects in the PDE-based modeling system. Fig. 17 shows the user interface of the system.

Geometric Boundary Representations. Users can interactively input and edit boundary surfaces and curves by

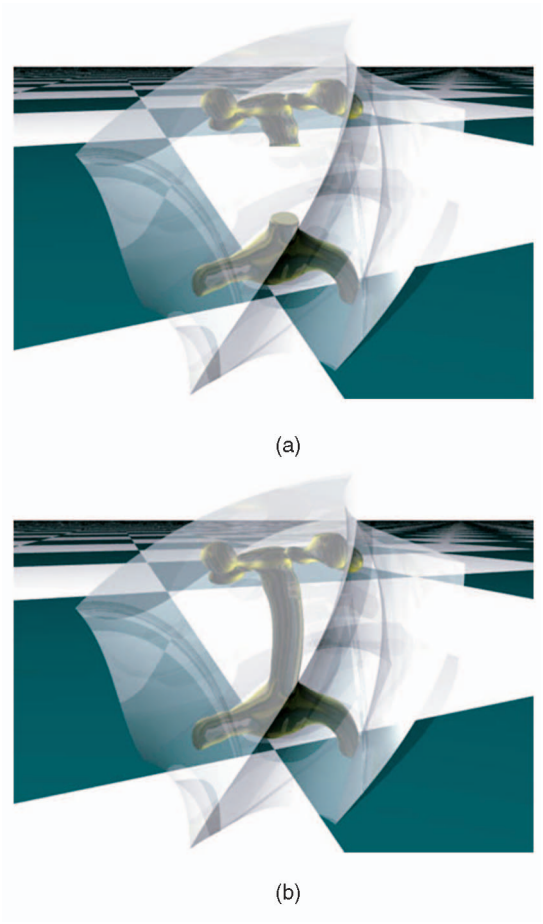

Fig. 13. Another example of PDE-based free-form shape blending and deformation. (a) Two objects before blending embedded in a PDE solid and (b) the result after blending.

selecting the boundary of interests and obtain PDE solids satisfying these conditions. The system also provides various manipulation toolkits to deform boundary surfaces and the solid geometry will be modified accordingly.

Geometric Interior Operations. Users can also work directly inside the PDE solid space through: 1) interior deformation with additional constraints inside the solid, 2) trimming specified regions for complex geometry and arbitrary topology, and 3) free-form deformation based on the mapping from the 3D parametric domain to physical space through trivariate PDEs.

Shape Modeling Based on Intensity Fields. We further integrate scalar intensity properties with PDE solid geometry for arbitrary shape modeling: 1) implicit PDE-based shape modeling through free-form deformation and direct manipulation, 2) arbitrary shape blending by integrating geometric and intensity properties in the working space, and 3) intensity-based shape deformation by changing the intensity distribution globally/locally in the PDE solid space.

Besides traditional boundary conditions of PDE techniques, the system allows users to specify and enforce a large variety of additional constraints on a set of points, cross-sectional curves, and surface areas on boundary surfaces. These constraints provide more freedom to designers, making the design process of PDE solids more cost-effective. The curve-based boundary conditions make it

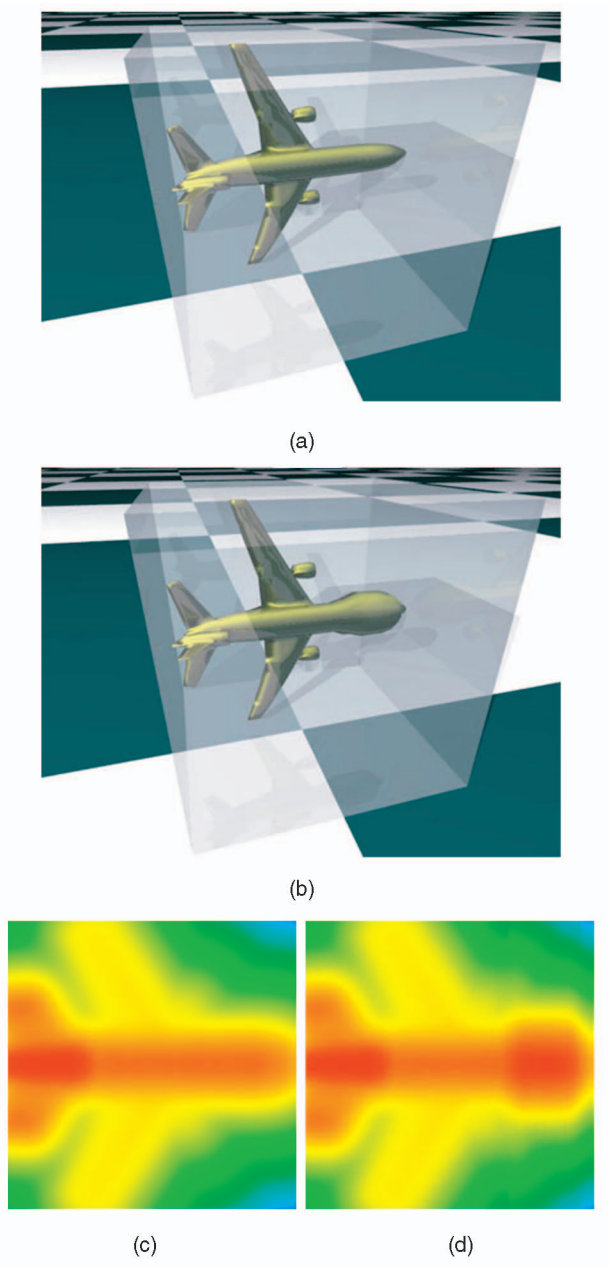

Fig. 14. An example for PDE-based FFD by intensity changes. (a) An embedded shape, (b) the deformed object to preserve its original intensity value in the locally modified intensity field, and (c) and (d) are corresponding cross-section views of the intensity fields along the w direction for (a) and (b), respectively.

even easier for designers to achieve the desired shapes of PDE solids from curve sketches. We use finite-difference techniques because they are simple, easy to implement, and suitable for the incorporation of complicated, flexible constraints. Because of the finite-difference discretization, the system allows users to enforce additional constraints directly inside the PDE solid and apply trimming operations, which facilitate the construction of PDE objects of arbitrary topology. The direct and free-form modeling based on PDE solid geometry and intensity distributions allow arbitrary shape blending and modification functionalities for the PDE-based geometric modeling system.

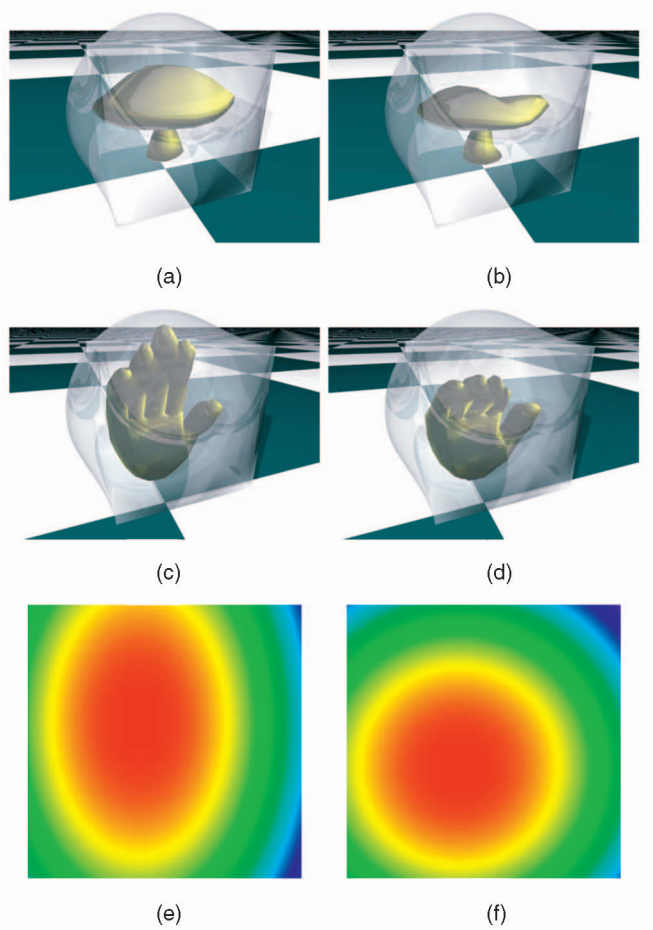

Fig. 15. Examples of PDE-based FFD due to intensity changes. (a) and (c) are embedded objects, (e) w-direction view of intensity distribution obtained from function $d(u,v,w) = e^{-(\sigma_u(u-u_0)^2+\sigma_v(v-v_0)^2+\sigma_w(w-w_0)^2)}$, (b) and (d) are deformed objects when modifying $d(u,v,w)$ by changing σ_u, σ_v, u_0, and v_0 shown in (f).

6.2 Performance

In general, the time and space complexities for the finite-difference solver will increase with higher resolution as well as increased accuracy. The multigrid subdivision method for various levels of refinement achieves anticipated results in our experiments. As an example, Table 1 shows the time performance for generating geometric PDE solids on an Intel Pentium M (1.10 GHz) laptop.

In Table 1, "Grids" stands for the resolution of the parametric space discretization,"Surfaces" represents a geometric PDE solid example obtained from a set of boundary surfaces. "Curves" stands for an example obtained from a boundary curve network. The "−4," "−2," and "−s" stand for the fourth, second order PDE, and the fourth order PDE with multigrid subdivision, respectively. We use a SOR (successive over-relaxation) version of Gauss-Seidel iteration. The multigrid subdivision can greatly improve the performance of our finite-difference solver. It is possible to obtain finer resolutions of solid objects using multigrid subdivision in reasonable time. And, the FFD feature enables our system to model complex objects in relatively coarse grids. Moreover, although solving the PDE for the entire parametric space is time consuming, direct manipulations can be much faster because only small regions of the solid space are involved. On the other hand, in recent years, GPUs have been used for linear algebra operations on sparse matrices with much faster performance [25]. We are also considering using GPUs to accelerate the convergence rate of our FDM solver for interactive manipulation and sculpting.

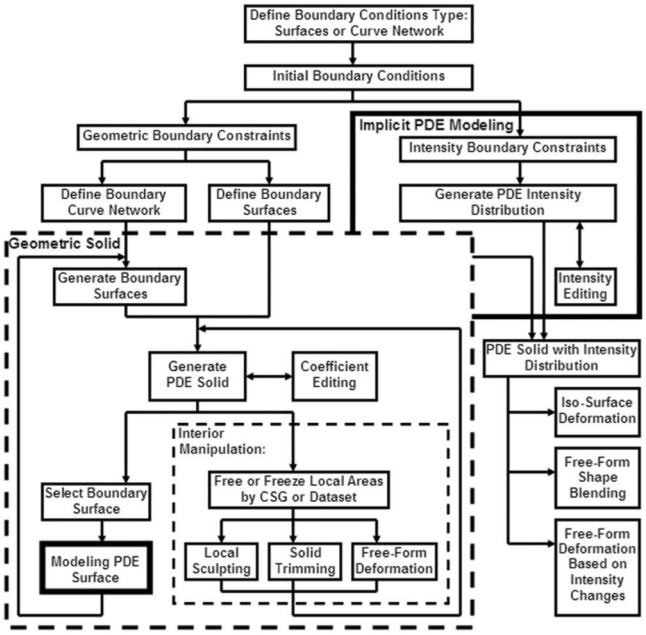

Fig. 16. System architecture.

6.3 Discussion

Our PDE-based geometric modeling system offers a free-form modeling scheme according to the mapping from parametric space to the physical space generated by the trivariate PDE. Comparing with other popular FFD techniques, our method has some unique features. First, the deformed physical space as the solution of the PDE can be viewed as a PDE space which satisfies the given equation. Different from spline-based and subdivision-based FFD techniques that require 3D control lattices/volumes to define the deforming space, only boundary surfaces are

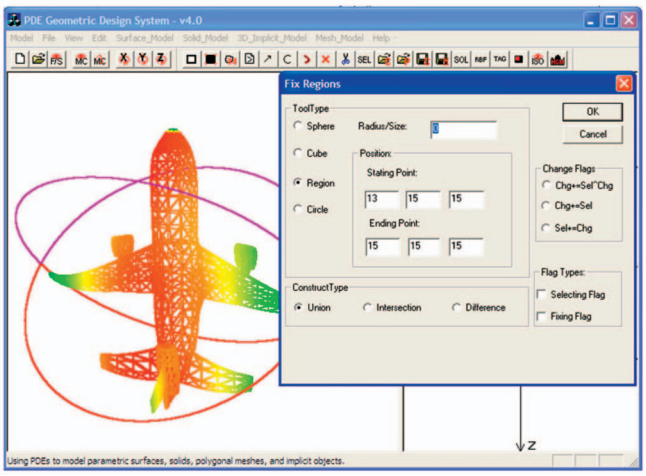

Fig. 17. Screen shot of the PDE-based geometric modeling system. The pop-up dialog is brought out to specify trimming tools.

TABLE 1
CPU Time (in Seconds) for Generating PDE Solid Geometry with Different Types of Boundary Conditions

Model	Grids	CPU-time
Surfaces-4	$65 \times 65 \times 65$	257.790
Surfaces-2	$65 \times 65 \times 65$	16.404
Surfaces-s	$65 \times 65 \times 65$	28.626
Curves-4	$65 \times 65 \times 65$	369.157
Curves-2	$65 \times 65 \times 65$	34.078
Curves-s	$65 \times 65 \times 65$	122.375

necessary to form the PDE space, and the interior information will be recovered and governed by the equation itself. Second, the PDE scheme also provides a continuous deforming space by default and modifications of any region in the space will be smoothly propagated into its neighborhood. Therefore, any object embedded in the PDE space will follow the continuous geometric distribution inside the solid space and avoid self-intersection. Third, we also provide sculpting of arbitrary interior regions for local deformation and direct manipulation similarly to other FFD techniques. Unlike most spline-based FFD techniques that have to modify the control lattices based on the direct manipulations first and then deform the rest of the object accordingly, any sculpting operation on local regions in our PDE model will directly affect neighboring areas according to the equation. However, because of the nature of FDM used in our system, interactivity is a concern when manipulating large regions. For improvement, we consider using GPUs to obtain interactive operations.

In addition, the incorporation of implicit PDEs for intensity distribution inherits modeling advantages of implicit models into our system. Thus, we can provide modeling functionalities that are difficult for parametric techniques such as arbitrary shape blending and shape deformation based on intensity manipulation. Compared to algebraic shape blending, our method offers local control and direct interference because our PDE formulation allows additional constraints directly enforced in local regions.

7 Conclusion

We present a novel geometric modeling framework that integrates PDE solids, surfaces, and implicit PDE models for general and arbitrary shape modeling including design, deformation, sculpting, and blending. The integrated model uses elliptic PDEs to govern both geometric and intensity properties of solid objects. The PDE solid geometry can be defined by either boundary surfaces or a set of curves as generalized boundary conditions governed by PDE surface formulation. The geometric PDE solid modeling techniques allow users to manipulate objects satisfying a set of design criteria and functional requirements with lots of degrees of freedom. The incorporation of intensity attributes to solid objects provides modeling advantages of implicit functions to the integrated PDE model. We developed a set of manipulation toolkits that support both global and local deformation of PDE solids subject to various constraints. Our PDE-based geometric modeling system offers PDE-governed boundary surface manipulations to deform the solid geometry, which permit users to model and edit boundary PDE surfaces and the geometric shape of corresponding PDE solid intuitively with ease. The deformation and trimming operations inside the solid space provide a natural way of free-form modeling for objects with arbitrary topology. In addition, users can manipulate intensity distributions of the PDE solid to obtain arbitrary shape blending and deformation for embedded objects. Our unified PDE-based approach greatly expands the geometric coverage and topological flexibilities of conventional PDE solids and implicit PDE models and improves the utility of PDE solids for modeling and design applications. With this integrated framework, it is possible to simulate real world objects with material properties.

Acknowledgments

This work was supported in part by US National Science Foundation (NSF) CAREER award CCR-9896123, NSF grants IIS-0082035 and IIS-0097646, NFS grant CCR-0328930, NSF ITR grant IIS-0326388, the Alfred P. Sloan Fellowship, the Honda Initiation Award, and an appointment to the NLM Research Participation Program sponsored by the National Library of Medicine and administered by the Oak Ridge Institute for Science and Education.

References

[1] C.M. Hoffmann, *Geometric and Solid Modeling: An Introduction.* Morgan Kaufmann, 1989.
[2] M. Mortenson, *Geometric Modeling,* second ed. Wiley Computer Publishing, 1997.
[3] M.I.G. Bloor and M.J. Wilson, "Generating Blend Surfaces Using Partial Differential Equations," *Computer Aided Design,* vol. 21, no. 3, pp. 165-171, 1989.
[4] M.I.G. Bloor and M.J. Wilson, "Using Partial Differential Equations to Generate Free-Form Surfaces," *Computer Aided Design,* vol. 22, no. 4, pp. 202-212, 1990.
[5] T.W. Lowe, M.I.G. Bloor, and M.J. Wilson, "Functionality in Blend Design," *Computer Aided Design,* vol. 22, no. 10, pp. 655-665, 1990.
[6] H. Ugail, M.I.G. Bloor, and M.J. Wilson, "Techniques for Interactive Design Using the PDE Method," *ACM Trans. Graphics,* vol. 18, no. 2, pp. 195-212, 1999.
[7] J.J. Zhang and L. You, "Surface Representation Using Second, Fourth and Mixed Order Partial Differential Equations," *Proc. Int'l Conf. Shape Modeling and Applications,* pp. 250-256, 2001.
[8] M.I.G. Bloor and M.J. Wilson, "Functionality in Solids Obtained from Partial Differential Equations," *Computing Supplement 8,* pp. 21-42, 1993.
[9] H. Du and H. Qin, "Integrating Physics-Based Modeling with PDE Solids for Geometric Design," *Proc. Eighth Pacific Conf. Computer Graphics and Applications (Pacific Graphics '01),* pp. 198-207, 2001.
[10] J. Bloomenthal, C. Bajaj, J. Blinn, M.-P. Cani-Gascuel, A. Rockwood, B. Wyvill, and G. Wyvill, *Introduction to Implicit Surfaces.* Morgan Kaufmann, 1997.
[11] H. Du and H. Qin, "Direct Manipulation and Interactive Sculpting of PDE Surfaces," *Computer Graphics Forum,* vol. 19, no. 3, pp. C261-C270, 2000.
[12] H. Du and H. Qin, "Dynamic PDE Surfaces with Flexible and General Constraints," *Proc. Eighth Pacific Conf. Computer Graphics and Applications (Pacific Graphics '00),* pp. 213-222, 2000.
[13] H. Du and H. Qin, "Dynamic PDE-Based Surface Design Using Geometric and Physical Constraints," *Graphical Models,* vol. 67, no. 1, pp. 43-71, 2005.
[14] T. Sederberg and S. Parry, "Free-Form Deformation of Solid Geometric Models," *Proc. SIGGRAPH '86,* pp. 151-158, 1986.
[15] S. Coquillart, "Extended Free-Form Deformation: A Sculpting Tool for 3D Geometric Modeling," *Proc. SIGGRAPH '90,* pp. 187-196, 1990.
[16] W.H. Hsu, J.F. Hughes, and H. Kaufman, "Direct Manipulation of Free-Form Deformation," *Proc. SIGGRAPH '92,* pp. 177-184, 1992.

[17] R. MacCracken and K.I. Joy, "Free-Form Deformations with Lattices of Arbitrary Topology," *Proc. SIGGRAPH '96,* pp. 181-188, 1996.
[18] K. Singh and E. Fiume, "Wires: A Geometric Deformation Technique," *Proc. SIGGRAPH '98,* pp. 405-414, 1998.
[19] J. Hua and H. Qin, "Scalar-Field-Guided Adaptive Shape Deformation and Animation," *The Visual Computer,* 2003.
[20] H. Du and H. Qin, "Interactive Shape Design Using Volumetric Implicit PDEs," *Proc. Eighth ACM Symp. Solid Modeling and Applications,* pp. 235-246, 2003.
[21] H. Du and H. Qin, "A Shape Design System Using Volumetric Implicit PDEs," *Computer-Aided Design,* vol. 36, no. 11, pp. 1101-1116, 2004.
[22] G. Strang, *Introduction to Applied Mathematics.* Wellesley-Cambridge Press, 1986.
[23] N. Dyn, D. Levin, and J.A. Gregory, "A Butterfly Subdivision Scheme for Surface Interpolation with Tension Control," *ACM Trans. Graphics,* vol. 9, no. 2, pp. 160-169, 1990.
[24] W.E. Lorensen and H.E. Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," *Proc. SIGGRAPH '87,* pp. 163-169, 1987.
[25] J. Krüger and R. Westermann, "Linear Algebra Operators for GPU Implementation of Numerical Algorithms," *ACM Trans. Graphics,* vol. 22, no. 3, pp. 908-916, 2003.

Haixia Du received the BS degree in computer science from Jilin University in Changchun, China, in 1995, the ME degree in computer science from the Institute of Mathematics, Chinese Academy of Sciences in Beijing, China, in 1998, and the MS and PhD degrees in computer science from the State University of New York at Stony Brook (Stony Brook University) in 2000 and 2004. She is a postdoctoral fellow in the 3D Informatics Group of the Office of High Performance Computing and Communications at the National Library of Medicine, National Institutes of Health. Her research interests include geometric and physics-based modeling, scientific and information visualization, and medical imaging, with emphasis on applying PDE-based techniques for geometric modeling, including shape design, reconstruction, metamorphosis, and simplification, and medical image processing including image filtering, segmentation, and registration. Detailed information about Dr. Haixia Du can be found at her Web site: http://www.haixiadu.net.

Hong Qin received the BS degree and the MS degree in computer science from Peking University in Beijing, China. He received the PhD (1995) degree in computer science from the University of Toronto. He is a full professor (with tenure) of computer science in the Department of Computer Science at the State University of New York at Stony Brook (Stony Brook University). During his years at the University of Toronto (UofT), he received the UofT Open Doctoral Fellowship. In 1997, Professor Qin was awarded the US National Science Foundation (NSF) CAREER Award. In December 2000, Professor Qin received the Honda Initiation Grant Award. In February 2001, Professor Qin was selected as an Alfred P. Sloan Research Fellow by the Sloan Foundation. In 2005, Professor Qin served as the general cochair for Computer Graphics International 2005 (CGI 2005) held on 22-24 June 2005 at SUNY Stony Brook. Currently, he is an associate editor for *IEEE Transactions on Visualization and Computer Graphics* (*TVCG*) and he is also on the editorial board of *The Visual Computer* (*International Journal of Computer Graphics*). He is the conference cochair for the ACM Solid and Physical Modeling Symposium 2007 (to be held at Tsinghua University, Beijing, China). His research interests include geometric and solid modeling, graphics, physics-based modeling and simulation, computer aided geometric design, human-computer interaction, visualization, and scientific computing. Detailed information about Dr. Hong Qin can be found from his Web site: http://www.cs.sunysb.edu/qin. He is a member of the IEEE.