Interactive Shape Design Using Volumetric Implicit PDEs

Haixia Du Hong Qin*
Department of Computer Science
State University of New York at Stony Brook

ABSTRACT

Solid modeling based on Partial Differential Equations (PDEs) can potentially unify both geometric constraints and functional requirements within a single design framework to model real-world objects via its explicit, direct integration with parametric geometry. In contrast, implicit functions indirectly define geometric objects as the level-set of underlying scalar fields. To maximize the modeling potential of PDE-based methodology, in this paper we tightly couple PDEs with volumetric implicit functions in order to achieve interactive, intuitive shape representation, manipulation, and deformation. In particular, the unified approach can reconstruct the PDE geometry of arbitrary topology from scattered data points or a set of sketch curves. We make use of a fourth-order elliptic PDE to define the volumetric implicit function. The proposed implicit PDE model has the capability to reconstruct a complete solid model from partial information and facilitates the direct manipulation of underlying volumetric datasets via sketch curves, iso-surface sculpting, deformation of arbitrary interior regions, as well as a set of CSG operations inside the working space. The prototype system that we have developed allows designers to interactively sketch the curve outlines of the object, define intensity values and gradient directions, and specify interpolatory points in the 3D working space. The governing implicit PDE treats these constraints as generalized boundary conditions to determine the unknown scalar intensity values over the entire working space. The implicit shape is reconstructed with specified intensity value accordingly and can be deformed using a set of sculpting toolkits. We use the finite-difference discretization and variational interpolating approach with the localized iterative solver for the numerical integration of our PDEs in order to accommodate the diversity of generalized boundary constraints.

*{dhaixia,qin}@cs.sunysb.edu

SM'03 June 16–20, 2003, Seattle, Washington, USA.

Categories and Subject Descriptors

I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*Curve, surface, solid, and object representations*; G.1.8 [**Numerical Analysis**]: Partial Differential Equations—*Finite difference methods*

General Terms

Algorithms, Design

Keywords

PDE Techniques, Implicit Functions, Volume Graphics, Shape Design, Geometric Constraints, Scattered Data Fitting

1. INTRODUCTION AND MOTIVATION

PDE techniques are very popular in many applications of graphics, animation, and visualization, such as nature phenomena simulation and animation [17], variational fairing[33], image inpainting[2], etc. They also provide an alternative way for geometric design [5, 6, 7, 43]. Different from the traditional geometric representations, the PDE methods model graphical objects as the solution of certain elliptic PDEs with boundary constraints inside the parametric domain. Although the parametric PDE model simplifies the geometric design process by using only boundary conditions to recover the whole interior information and offers high-order continuity as well as energy minimization properties, like traditional parametric approaches, it's extremely difficult to model arbitrary shapes of general topology, because the PDE is defined over regular parametric domain.

In contrast, instead of constructing the mapping between parametric and physical spaces, implicit functions use the level-set of certain field functions in the physical domain directly to design, model, and interact with 3D objects. They offer a fundamentally different yet convenient and natural design paradigm (in comparison with parametric representations) in visual computing fields such as graphics, animation, and geometric design. This is because of their unique properties such as arbitrary topology, collision detection, free of parametric correspondence, etc. Applications of implicit functions include shape blending, surface reconstruction from scattered data points, shape transformation, and interactive modeling [3, 4, 8, 9, 10, 12, 13, 18, 21, 22, 26, 27, 29, 31, 34, 37, 38, 41, 42, 45].

In essence, implicit functions offer several modeling advantages such as flexible topology, simple data structure, efficient storage, volumetric data information, unbounded geometry, etc. Nonetheless, most of implicit functions focus

on surface models. The previous techniques for interactive implicit volume sculpting have certain modeling limitations. Recently, Cutler *et.al.*[11] presented a procedural framework for specifying layered solid models and applying a series of simulation operations (serving as sculpting tools described by a script language) to complex models. Bærentzen and Christensen [1] developed an interactive volume sculpting system using level-set method. Museth *et.al.*[28] proposed level-set-based editors for CSG operations, blending, embossing, and smoothing for implicit surfaces. However, these tools are associated with the specification of speed functions for the evolving level-set, which are non-intuitive for common users. Turk and O'Brien [39] presented interactive implicit surface sculpting via particles, but each operation requires reformatting and recalculation of the entire system, which is difficult to model large datasets. In general, the modeling potential of implicit functions has not been fully explored yet and there is a lack of systematic toolkits to design, reconstruct, and sculpt implicit models.

To maximize the modeling capabilities of PDE techniques and implicit functions in geometric and visual computing areas, we propose a PDE-based modeling paradigm which integrates the PDE techniques with implicit functions into one single framework for interactive shape design and manipulation on PDE-based volumetric implicit models. We develop an implicit modeling system governed by a fourth-order elliptic PDE of scalar intensity fields. In particular, our prototype system can reconstruct implicit objects and the embedding implicit 3D working space as the solution of PDE by specifying a set of curve outlines or scattered data points of certain intensity values as *general* boundary constraints with the assistance of variational interpolating approaches. Because the curves and datasets are not required to be closed, open surfaces can be modeled within our system. Moreover, it offers a set of sculpting toolkits to manipulate implicit objects, such as interactively modifying the geometric shape, intensity value, and gradient direction of selected sketch curves, directly changing intensity values of selected regions in the working space, as well as deforming iso-contours at specified intensity values of the objects. Because the working space is governed by the PDE, any missing information inside the space can be recovered by the PDE according to the given constraints. Our system is able to recover damaged datasets using partial information, smooth the intensity distribution of volume data, and smoothly blend objects inside the working space. In general, our system allows intensity manipulation anywhere in the implicit working space to model the implicit objects at any iso-value either directly or indirectly, which offers users both local and global control of the implicit PDE model.

This implicit PDE approach offers advantages of both parametric PDE techniques and implicit functions. First, the behavior of the implicit PDE model is governed by differential equations. Solving the PDEs results in both boundary and interior information simultaneously, which offers an alternative way to model implicit objects by using only boundary information. This property makes PDE methods extremely suitable for shape blending process. Second, many natural physical processes are characterized by differential equations in principle [19, 20, 35]. Hence, PDE models are natural and close to the real world. They are potentially ideal candidates for design, simulation, and analysis tasks. Furthermore, geometric objects with high-order continuity requirements can be readily defined through high-order PDEs because of their differential properties. Third, smooth objects that minimize certain energy functionals are the solutions of differential equations from the variational analysis point of view, so optimization techniques can be unified with PDE models. In addition, because the implicit PDE is formulated on a scalar intensity field and defines objects by collecting points of certain iso-values, it is capable of designing arbitrary topological shapes and recovering the full information from partial input, which reduces the burden of specifying the large quantity of constraints for complete datasets. It offers users a natural way to design objects easily with general non-isoparametric arbitrary curve outlines, reconstruct objects from scattered data points, and recover the damaged datasets.

The remainder of the paper is structured as follows. Section 2 reviews the related work of PDE techniques and implicit models. We detail the PDE formulation and present our integrated approach for implicit PDE objects in Section 3. We introduce possible applications of our implicit PDE model by enforcing different types of boundary and additional constraints in Section 4. Section 5 discusses techniques of directly manipulating implicit PDE objects with constraints to construct more flexible topological shapes, such as sketch sculpting and local region manipulations. We outline the system implementation in Section 6. Section 7 concludes the paper.

2. PRIOR WORK

Different from traditional free-form spline-based modeling techniques, Bloor and Wilson [5] introduced a method that defines a smooth surface as a solution of *elliptic* PDEs. Since its initial application on surface blending, PDE approach has broadened its uses in free-form surface design, solid modeling, and interactive surface editing [6, 7, 40] during the past decade. In principle, the PDE-based method has the advantage that most of the information defining an object comes from its boundaries. This permits an object to be generated and controlled by a very few parameters such as boundary-value conditions and global coefficients associated with an elliptic PDE. This PDE technique was then used for modeling parametric surfaces and solids with global geometric features. To obtain interactive sculpting and local control, we [14, 15] proposed an integrated model which combined the PDE surfaces and the physics-based modeling techniques to offer users direct manipulation for the PDE surfaces with generalized boundary constraints and user-specified features. We [16] extended the PDE techniques coverage from surfaces to solids in order to provide users a set of direct editing toolkits to model the real-world objects with interior material distribution. Zhang and You [43] investigated three different orders, i.e., second, mixed, and fourth order of PDEs as surface representation techniques and demonstrated the use and effectiveness of the PDE method for free-form surface design.

However, because the aforementioned PDE methods define objects over the regular parametric domain, they (like other parametric representation techniques) have limitations in handling arbitrary topological shapes, which can be easily achieved by implicit functions.

Implicit functions offer a different way for shape modeling by using certain scalar field functions to define geometric entities. In the past several years, implicit functions

have been widely developed as a powerful design and manipulation tool for graphical models. In 1994, Witkin and Heckbert [42] introduced an approach using particles to sample and control implicit surfaces. Ferley *et.al.* [18] presented a *sculpture metaphor* for rapid shape prototyping. These techniques only provide interactive and practical sculpting tools for implicit surfaces. As for implicit solids, Savchenko *et.al.* [32] introduced a novel approach to the reconstruction of geometric models from given point sets using volume splines. Raviv and Elber [31] presented an interactive sculpting technique that uses the zero level set of the scalar, tensor-product, uniform trivariate B-spline functions to represent 3D objects. The trivariate functions have a control volume that consists of scalar control coefficients. Users can indirectly sculpt the object by modifying relevant scalar control coefficients of the trivariate B-spline functions in different levels of details. Hua and Qin [23, 24] developed interactive solid sculpting toolkits with haptics on implicit B-spline solids defined through the use of B-spline control coefficients over the intensity field. However, the control of B-spline coefficients is less intuitive to ordinary users in general. Implicit functions can also be used for shape reconstruction and 3D morphing process. Turk and O'Brien [38] used variational implicit functions to achieve shape morphing and surface reconstruction. They employed the Radial Basis Function (RBF) method to construct an implicit function which interpolates the given dataset and minimizes the thin-plate energy. Yet since the RBF method is a global variational interpolating approach, any changes in the dataset will cause recalculation of the entire system. It's time-consuming for direct manipulation and not applicable for local sculpting of complex implicit surfaces. Level-set method is another popular technique to model implicit surfaces. Zhao *et.al.* [45] proposed a *weighted* minimal surface model based on variational formulations and PDE techniques to construct a surface from scattered data. They used the level-set method as a numerical technique to evolve the implicit surface continuously following the gradient descent of the energy functional for the final reconstruction. Their level-set model is governed by a time evolution PDE with velocity at the level-sets given by the motion equation of the original surface. The level-set method is based on a continuous formulation using PDEs and deforms an implicit surface according to various equations of motion depending on geometry, external forces, or certain energy minimization. It can easily handle topological changes and reduce noises in the dataset. The level-set method mainly focused on implicit objects reconstructed from scattered datasets. Problems for interpolating curve sketches, especially open curve sketches have not been addressed. The shape deformation using the level-set method is often obtained by manipulating the speed functions in the level-set formulation [1, 28], which is non-intuitive for general users.

Despite the modeling advantages of implicit functions, there is a lack of systemic modeling toolkits for design and direct manipulation of implicit surfaces and solids in general. We integrate the implicit functions with the parametric PDE to offer users modeling advantages of both types of techniques. Instead of time evolution PDEs used in the level-set method, we employ static elliptic PDEs for boundary value problems. In particular, we introduce a novel technique which defines the volumetric implicit objects as the solution of the fourth-order elliptic PDE of scalar intensity fields under *generalized* boundary constraints, including sketch curves, scattered data points, as well as volumetric datasets. The constraints may be associated by intensity values different from each other, which offers more degrees of freedom than the previous implicit techniques. This method can be used for geometric shape design, object reconstruction, damaged data recovery, and shape blending. The implicit PDE objects can be manipulated by modifying the initial constraints or directly changing intensity values in the interior of the volumetric space. Using the implicit PDE, not only the objects, but also the entire working space can be recovered by the given information. Our system does not require the constraints to be closed datasets, which provides modeling potentials for open surfaces. To visualize the scalar intensity field, we can either use the Marching Cube method [25] which calculates the triangulated iso-surface at selected intensity value on discretized sampling grids, or output the volume data to other volume rendering systems such as Pov-Ray and Vol-Vis systems.

3. FORMULATING IMPLICIT PDE

This section formulates the fourth-order elliptic PDE of scalar intensity fields in 3D, and outlines major properties of the unified principle of PDE techniques. We also propose the constrained implicit PDE for direct manipulation and the numerical techniques discretizing and solving the equations.

3.1 Implicit Elliptic PDE

The implicit PDE formulation employed in this paper is founded upon the parametric PDE solid models[16]. In order to take advantage of the interactive feature associated with the parametric PDE modeling techniques, we use elliptic PDEs to define scalar intensity field for modeling implicit objects. Because higher-order PDEs can provide higher-order continuity for the scalar intensity value distribution, we use the fourth order elliptic PDE to model the scalar field to obtain smooth results with tangential continuity, especially when dealing with shape blending and damage data recovery in which most of information are specified as constraints. In particular, we formulate the unknown function as the intensity field function $d(x, y, z)$ defined in the physical space of x, y, and z. The corresponding implicit PDE is formulated as follows:

$$(a^2\frac{\partial^2}{\partial x^2} + b^2\frac{\partial^2}{\partial y^2} + c^2\frac{\partial^2}{\partial z^2})^2 d(x, y, z) = 0, \qquad (1)$$

where x, y, and z are coordinate variables of 3D physical space varying from 0 to 1, respectively, which form a unit cube as the working space; a, b, and c are arbitrary blending coefficient functions of x, y, z defining material properties of the implicit space, which are initially defined as constants throughout the entire working space.

Because $a(x, y, z)$, $b(x, y, z)$, and $c(x, y, z)$ are allowed to vary across $d(x, y, z)$, i.e., different locations in the physical domain may have different smoothing coefficient values, local control on implicit PDE objects can be easily achieved.

To obtain direct and local manipulation on the implicit PDE objects, we solve (1) using numerical methods based on finite-difference approximations of the PDE, which require at least six boundary conditions at $x = 0, x = 1, y = 0, y = 1, z = 0, z = 1$ defining the intensity values at three boundary surface pairs of the 3D physical working space in order to derive a unique solution. However, in most applications,

there are no such boundary conditions available for modeling implicit objects, especially in the case of using implicit functions for shape reconstruction, where the constraints are usually defined by certain contouring sketch curves or scattered points assigned with specified intensity values inside the 3D working space. In such cases, the intensity distributions on the boundaries are unknown. Thus, such problems cannot be solved by traditional finite-difference methods directly. We can solve this type of problems by first finding an initial guess of the volumetric working space, then approximating the solution of the implicit PDE using the guessed boundary values with the given constraints. After that, we can enforce direct manipulations inside the working space by adding additional constraints to the PDE. Variational interpolating approaches are good candidates for shape reconstruction from scattered points, such as the RBF method [26, 38] which creates a 3D implicit function to give an approximation interpolating the given constraints by minimizing certain energy functional. We employ the RBF method to compute the initial guess of the implicit PDE objects. We can also calculate the intensity values on sampled grids using their distance to the constraints, because the implicit objects can be defined by distance functions. The algorithm we use to compute the distance field is the fast-tagging approach proposed by [44].

3.2 Radial Basis Function

The RBF approach is commonly used for scattered data interpolation, which is to generate a smooth surface that passes through a given set of (unorganized) data points. Scattered data interpolation sometimes can also be addressed using variational analysis where the desired solution is a function, $f(\vec{x})$, that will minimize certain energy functionals. In principle, the energy functional measures the quality of interpolation subject to the interpolatory constraints $f(\vec{c}_i) = h_i$. It can be solved by a weighted sum of certain radial basis functions (note that, we use $\phi(\vec{x}) = |\vec{x}|^3$ in this paper). Then the interpolation function can be formulated as:

$$f(\vec{x}) = \sum_{i=1}^{n} w_i \phi(\vec{x} - \vec{c}_i) + P(\vec{x}), \tag{2}$$

where $\vec{c}_i$'s are the coordinate vectors of the constraints, the w_i's are the weights, and $P(\vec{x})$ is a polynomial only consisting of the linear and constant portions of f. According to the properties of the appropriate radial basis functions, the interpolation function minimizes the thin-plate energy while satisfying the data interpolation requirement. By applying the constraints to (2), we can obtain a linear equation system whose unknowns are the weights and coefficients of the polynomial P. This system can be solved using standard solvers of linear equations.

However, the RBF method requires gradient information of the datasets, and space complexity of the equation system depends on the number of constraints, so it's not suitable for reconstruction of arbitrary scattered datasets and interactive sculpting of large number of constraints. Because our goal here is simply making an initial guess for our implicit PDE shape, fast-tagging algorithm to approximate distance functions gives satisfactory results for such input.

3.3 Numerical Simulation

In order to easily apply additional constraints for direct manipulation of the implicit objects, we resort to the numerical techniques based on the finite-difference approximation and iterative method for linear equations to solve the implicit PDE with predefined boundary values or approximated initial guess from sketch curves/scattered points and arrive at an approximated solution with user-specified error tolerances. Numerical algorithms also facilitate the material modeling of anisotropic distribution. A multi-grid-like iterative solver is used to improve the system performance.

The finite-difference method divides the working space into discrete grids along x, y, z directions and transforms a continuous PDE into a set of simultaneous algebraic equations by sampling the partial derivatives in differential equations for each grid point with their discretized approximations. The algebraic equation system can be solved numerically either through a direct procedure or an iterative process for an approximated solution of the continuous PDE.

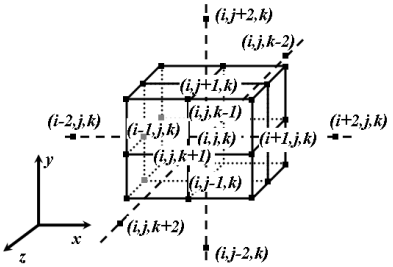


Figure 1: The point discretization of an implicit function.

Based on Taylor's expansion, the derivatives of a univariate function can be approximated using the central-difference approximation $f'(x) = (f(x+h) - f(x-h))/2h$, $f''(x) = [f(x+h) - 2f(x) + f(x-h)]/h^2$, where h denotes the spatial step. This can be generalized to all partial derivatives on trivariate implicit geometry, by dividing the x, y, z domain into l, m, and n discretized grids respectively. We represent the function $d(x, y, z)$ by its values at the discrete set of points $(x_i = i\Delta x, y_j = j\Delta y, z_k = k\Delta z)$, $i = 0, 1, ..., l-1$, $j = 0, 1, ..., m-1$, and $k = 0, 1, ..., n-1$. $\Delta x, \Delta y, \Delta z$ are the *grid spacing* along x, y, z directions. We write $d_{i,j,k}$ for $d(x_i, y_j, z_k)$ and $\{i, j, k\}$ for grid point (x_i, y_j, z_k) for sake of simplicity (Fig. 1). We use the finite-difference representation of fourth order partial derivatives $\frac{\partial^4 d_{i,j,k}}{\partial x^4}$ and $\frac{\partial^4 d_{i,j,k}}{\partial x^2 \partial y^2}$at $\{i, j, k\}$ as examples:

$$\frac{\partial^4 d_{i,j,k}}{\partial x^4} = \frac{d_{i-2,j,k} + d_{i+2,j,k} - 4d_{i-1,j,k} - 4d_{i+1,j,k} + 6d_{i,j,k}}{(\Delta x)^4},$$

$$\frac{\partial^4 d_{i,j,k}}{\partial x^2 \partial y^2} = \frac{\frac{d_{i-1,j-1,k} + d_{i-1,j+1,k} + d_{i+1,j-1,k} + d_{i+1,j+1,k}}{(\Delta x)^2 (\Delta y)^2} + \frac{-2d_{i-1,j,k} - 2d_{i+1,j,k} - 2d_{i,j-1,k} - 2d_{i,j+1,k} + 4d_{i,j,k}}{(\Delta x)^2 (\Delta y)^2}}{}.$$

Substituting by the finite-difference representation at grid points, (1) can be rewritten as:

$$\mathbf{AD} = \mathbf{b}, \tag{3}$$

where $\mathbf{A}$ represents the discretized differential operator in $(l \times m \times n) \times (l \times m \times n)$ matrix form, where each row in $\mathbf{A}$ is corresponding to the difference equation at a grid point. $\mathbf{A}$ is also controlled by the blending functions $a(x, y, z)$, $b(x, y, z)$, and $c(x, y, z)$. $\mathbf{D}$ collects the unknown intensity values at the

grid points, and $\mathbf{b}$ is defined by the value of constraints:

$$\mathbf{A} = \left[\mathbf{A}_{(0,0,0)}, \mathbf{A}_{(0,0,1)}, \cdots, \mathbf{A}_{(l-1,m-1,n-1)}\right]^\top,$$
$$\mathbf{A}_{(i,j,k)} = \left[A_{(i,j,k),(0,0,0)}, \cdots, A_{(i,j,k),(l-1,m-1,n-1)}\right],$$
$$\mathbf{D} = \left[d_{(0,0,0)}, d_{(0,0,1)}, \cdots, d_{(l-1,m-1,n-1)}\right]^\top,$$
$$\mathbf{b} = \left[b_{(0,0,0)}, b_{(0,0,1)}, \cdots, b_{(l-1,m-1,n-1)}\right]^\top.$$

$\mathbf{A}$ and $\mathbf{b}$ are defined as follows: Given a grid point $\{i, j, k\}$, let its index $d = i \times l \times m + j \times m + k$ be represented as (i, j, k). If it's a constraint point, all elements in $\mathbf{A}_{(i,j,k)}$ have value 0 except $A_{(i,j,k),(i,j,k)} = 1$, and $b_{(i,j,k)}$ is set to be the intensity value defined by the constraint. If it's free, the value of $A_{(i,j,k),(i',j',k')}$ depends on contribution of $\{i', j', k'\}$ in the difference equation at $\{i, j, k\}$, and $b_{(i,j,k)} = 0$. Fig. 1 shows the grid points contributing for $\{i, j, k\}$ in the d^{th} row of $\mathbf{A}$, $\mathbf{A}_{(i,j,k)}$. All the values of $A_{(i,j,k),(i',j',k')}$ are set to be 0 except:

$$\begin{aligned}
A_{(i,j,k),(i,j,k)} &= 6\left(\frac{a^4_{i,j,k}}{\Delta x^4} + \frac{b^4_{i,j,k}}{\Delta y^4} + \frac{c^4_{i,j,k}}{\Delta z^4}\right) + \\
&\quad 8\left(\frac{a^2_{i,j,k}b^2_{i,j,k}}{\Delta x^2\Delta y^2} + \frac{a^2_{i,j,k}c^2_{i,j,k}}{\Delta x^2\Delta z^2} + \frac{b^2_{i,j,k}c^2_{i,j,k}}{\Delta y^2\Delta z^2}\right), \\
A_{(i,j,k),(i\pm1,j,k)} &= -4\left(\frac{a^4_{i,j,k}}{\Delta x^4} + \frac{a^2_{i,j,k}b^2_{i,j,k}}{\Delta x^2\Delta y^2} + \frac{a^2_{i,j,k}c^2_{i,j,k}}{\Delta x^2\Delta z^2}\right), \\
A_{(i,j,k),(i,j\pm1,k)} &= -4\left(\frac{b^4_{i,j,k}}{\Delta y^4} + \frac{a^2_{i,j,k}b^2_{i,j,k}}{\Delta x^2\Delta y^2} + \frac{b^2_{i,j,k}c^2_{i,j,k}}{\Delta y^2\Delta z^2}\right), \\
A_{(i,j,k),(i,j,k\pm1)} &= -4\left(\frac{c^4_{i,j,k}}{\Delta z^4} + \frac{a^2_{i,j,k}c^2_{i,j,k}}{\Delta x^2\Delta z^2} + \frac{b^2_{i,j,k}c^2_{i,j,k}}{\Delta y^2\Delta z^2}\right), \\
A_{(i,j,k),(i\pm2,j,k)} &= \frac{a^4_{i,j,k}}{\Delta x^4}, \\
A_{(i,j,k),(i,j\pm2,k)} &= \frac{b^4_{i,j,k}}{\Delta y^4}, \\
A_{(i,j,k),(i,j,k\pm2)} &= \frac{c^4_{i,j,k}}{\Delta z^4}, \\
A_{(i,j,k),(i\pm1,j\pm1,k)} &= \frac{2a^2_{i,j,k}b^2_{i,j,k}}{\Delta x^2\Delta y^2}, \\
A_{(i,j,k),(i\pm1,j,k\pm1)} &= \frac{2a^2_{i,j,k}c^2_{i,j,k}}{\Delta x^2\Delta z^2}, \\
A_{(i,j,k),(i,j\pm1,k\pm1)} &= \frac{2b^2_{i,j,k}c^2_{i,j,k}}{\Delta y^2\Delta z^2}.
\end{aligned}$$

The matrix $\mathbf{A}$ is called "tridiagonal with fringes" [30].

Our implicit PDE is open along all of x, y, and z directions, so forward/backward difference approximations shall be utilized for the computation of partial derivatives near the six boundaries instead. Arbitrary boundary/additional constraints can be easily enforced by finite-difference method. In our system, after making the initial guess of the intensity values, we fix the intensity values at those boundaries, so that the manipulations on the implicit objects can be performed using the finite-difference iterative solver. In general, this type of elliptic PDEs allows designers to choose (various) constraints based on diverse design tasks.

3.4 Constrained System

The PDE modeling techniques have an attractive advantage that the interior of the objects is controlled by PDEs without the need of extra specification for interior material distribution. More importantly, users can modify an implicit PDE object by enforcing additional hard constraints of desired intensity values anywhere inside the working space without violating all the previously defined conditions. Additional hard constraints inside the working space introduce a set of new equations into the system to replace the corresponding original difference equations. For example, if we want to set the intensity value $d_{i,j,k}$ as a particular constant value d_0, the equation $d_{i,j,k} = d_0$ will be used to replace the equation the discretized difference equation approximating the PDE at the point $\{i, j, k\}$, i.e. $A_{(i,j,k)(i,j,k)} = 1$, all other $A_{(i,j,k)(i',j',k')} = 0$, and $b_{(i,j,k)} = d_0$. After replacing all the equations according to the constraints, (3) becomes

$$\mathbf{A}_c\mathbf{D} = \mathbf{b}_c, \tag{4}$$

where $\mathbf{A}_c$ and $\mathbf{b}_c$ are obtained by replacing $k(k > 0)$ equations in the original system with those derived from additional k constraints at the corresponding coordinate positions.

3.5 Iterative Method

With predefined or approximated boundary conditions from initial guess, (3) and (4) are solved using finite-difference-based iterative techniques. These methods make immediate use of the structure of the sparse matrix on the left-hand side of the equations, e.g., $\mathbf{A}$ in (3). The matrix is split into two parts

$$\mathbf{A} = \mathbf{A}_d - \mathbf{A}_r, \tag{5}$$

where $\mathbf{A}_d$ consists of the diagonal elements of $\mathbf{A}$ and zeros elsewhere, $\mathbf{A}_r$ is the remainder. Then (3) becomes

$$\mathbf{A}_d\mathbf{D} = \mathbf{A}_r\mathbf{D} + \mathbf{b}. \tag{6}$$

The iterative methods start from choosing an initial guess $\mathbf{D}^{(0)}$ and then solving successively for iterates $\mathbf{D}^{(s)}$ from

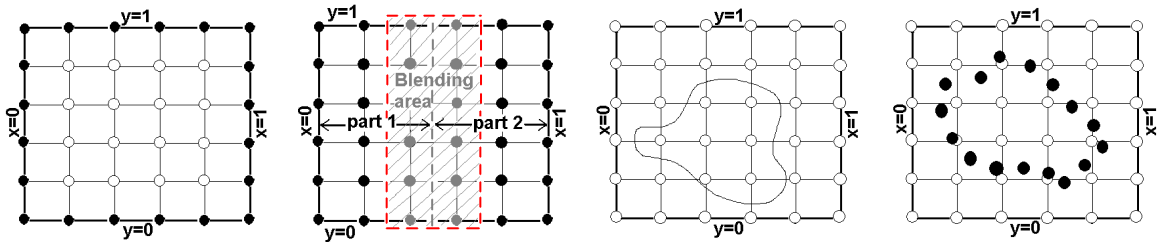
$$\mathbf{A}_d\mathbf{D}^{(s)} = \mathbf{A}_r\mathbf{D}^{(s-1)} + \mathbf{b}. \tag{7}$$

The same idea can be applied to the constrained system (4).

In the case of predefined boundary conditions, we compute the initial guess by simple linear interpolation based on the constraints. The iteration will stop at $\mathbf{D}^{(s)}$ for an approximated solution when the difference between $\mathbf{D}^{(s)}$ and $\mathbf{D}^{(s-1)}$ is less than a threshold (we use 10^{-9} in this paper). Certain variants of iterative techniques exist for solving the aforementioned linear equations [36]. In this paper, we employ the Gauss-Seidel iteration which uses the updated value of the iteration result at a grid point on the right-hand side of (7) as soon as it becomes available. To further speed up the converging rate of Gauss-Seidel iteration, we take into account the error factor which is characterized by the difference between the approximation and the real solution. This leads to the method of Successive Over-Relaxation iteration, or SOR iteration. Nonetheless, the discretization of volumetric implicit PDE space results in a very large number of linear equations. This causes the slow convergence of iterative methods. To achieve a solution faster, we start solving the equations at a coarse grid with down-sampled constraints and interpolate the solution at finer grids to compute the initial guess for the iterative methods at the finer resolution. The convergent rate of the iterative solvers can be greatly increased.

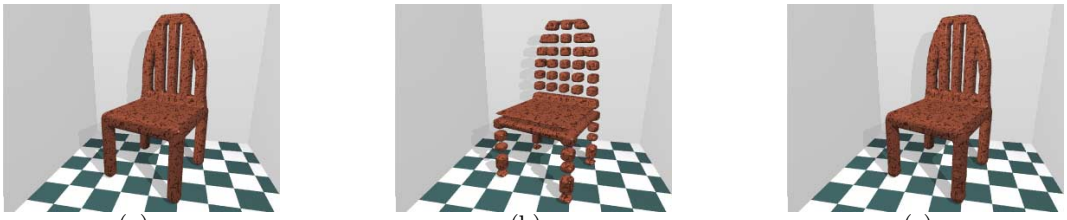
4. BOUNDARY CONDITIONS FOR DIFFERENT APPLICATIONS

To construct an implicit PDE object, first we need to outline the rough shape of the object, which can be defined through boundary conditions or special constraints such as curve contours and scattered data points in the working space that the object interpolates. The form of boundary constraints varies for different applications. Our implicit PDE techniques cover the boundary conditions for applications such as shape blending, shape reconstruction from sketch curves and scattered data points. Fig. 2 illustrates different types of boundary conditions in simplified 2D cases.



(a) (b) (c) (d)

Figure 2: 2D illustrations of different types of boundary conditions. (a) Traditional boundary constraints; (b) boundary conditions for shape blending; (c) sketch-curve constraints; (d) scattered-point constraints.



(a) (b) (c)

Figure 3: An implicit PDE object generated from cross-sectional boundary conditions: (a) Original object; (b) cross-sectional boundary conditions by removing several data slices along the y-direction from the original data; (c) is the generated implicit object from (b).

4.1 Shape Design Using Traditional Boundary Constraints

The implicit PDE can model geometric shapes by computing the information of the whole working space based on traditional boundary constraints with optional cross-sectional details inside the working space. Such boundary conditions are defined as intensity values sampled at certain resolution from input or use some analytic functions to generate implicit boundary functions $d(0, y, z)$, $d(1, y, z)$, $d(x, 0, z)$, $d(x, 1, z)$, $d(x, y, 0)$, $d(x, y, 1)$ and a collection of cross-sectional scalar intensity functions $d(x_i, y, z)$, $d(x, y_j, z)$, or $d(x, y, z_k)$, where $x_i, y_j, z_k \in (0, 1)$ are constants. These functions are sampled at specified resolution to provide a set of intensity values inside the working space. Using these values as generalized boundary conditions, we introduce certain number of new equations and the linear equation system has the form of (4) which can be solved using above mentioned techniques. Fig. 3 shows an example.

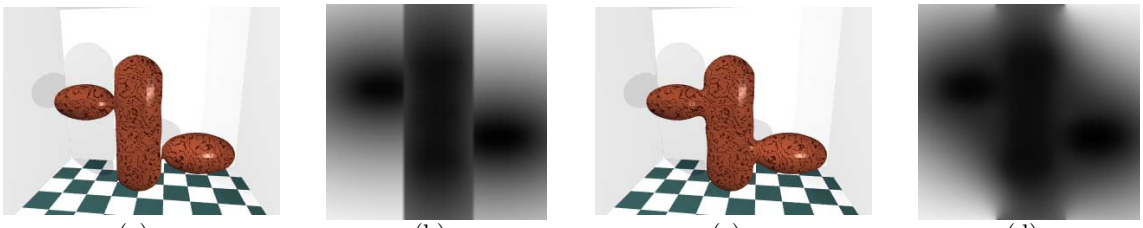
4.2 Shape Blending

Our PDE formulation defines the interior information of the implicit object via the differential properties, which means that it is possible to automatically recover the missing information from partial data with our prototype system and guarantee intensity continuity of non-constrained parts of the working space. This feature can be applied for shape blending process by placing the objects to be blended into the working space and the system will compute the connecting parts between those objects. Such kind of datasets form another type of initialization with pre-defined boundary constraints, which gives most of the information and only a small portion of the working space is missing. The missing information of the working space can be approximated based on the remaining part using our PDE formulation. An example of shape blending is shown in Fig. 4. The above two types of boundary conditions allow our system to model volumetric datasets.

4.3 Shape Reconstruction from Sketch Curves and Scattered Data points

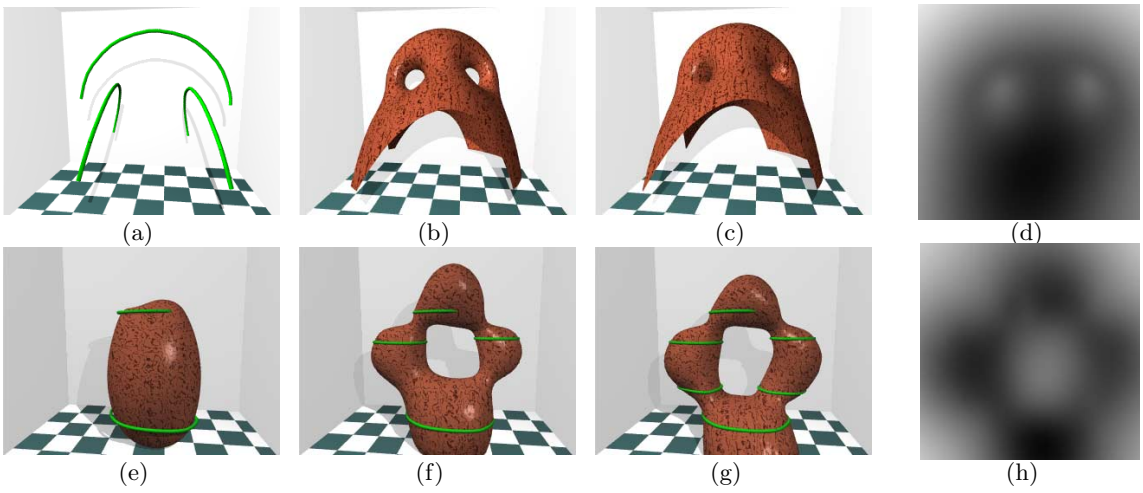
The implicit functions are very useful for shape reconstruction from scattered data points. To maximize the modeling potential of implicit PDEs, we develop a set of toolkits using PDE techniques to reconstruct objects from spatial sketch curves or scattered data points of specified intensity values. Because with this type of constraints, the boundary information around the working space is missing, it's extremely difficult to directly solve the implicit PDE under such constraints. Therefore, we employ techniques such as the RBF method for the interpolation problems and signed distance field approximation to obtain an initial guess for the implicit PDE shapes subject to those constraints. We then use the iterative solver to get a smooth solution. When performing the RBF method, the gradient information indicating the change of the intensity values around the constraints will be needed to define the inside and the outside of the reconstructed shape. If the gradient information is not provided by users, our system calculates the gradient at each sample point of the constraints according to the normal of the local tangent plane of the curve at that point, as explained in Fig. 6. Our system also allows designers to interactively input certain sketch curves such as B-spline curves with specified intensity value, which permits the initial sketch curves being modified directly. Note that, the sketch curves are not required to be planar curves. Moreover, they can even be open curves, which may result in open iso-surfaces instead of solid objects. Fig. 5 shows examples using sketch curves.

When modeling more complex shapes from sketch, usually there are a large number of sketch curves need to be enforced, which will increase the number of calculations dra-



(a) (b) (c) (d)

Figure 4: An example of shape blending. (a) Original dataset; (b) cross-section view of the working space for (a) where darkness increases with intensity; (c) blended object from (a); (d) cross-section view of (c).



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 5: Examples of shape reconstruction from sketch curves. (a) is a set of open curves without specified gradient information; (b) and (c) are iso-surfaces at different iso-values, respectively; (d) is a cross-section view of the implicit shape; (e), (f), and (g) show an example of generating implicit shapes by incrementally defining a set of B-spline curves; (e) is an object defined by two curves; (f) is the refined object by adding two additional sketch curves; (g) is the shape reconstructed from six B-spline sketch curves; and (h) is a cross-section view.

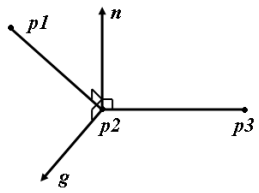


Figure 6: Illustration of computing the gradient direction.

matically. Moreover, sometimes the sketch curves are only designed to model the local area they resides, so their global contribution are not desirable. To address such issues, our system allows users to compute the initial guess of implicit PDE objects using the RBF method for selected subset of sketch curves at any local region of the working domain without disturbing the outside areas. At the initialization stage, when using RBF method to compute the initial guess of the implicit shape, users are prompted to select interested curves, define the region in the working space to reconstruct the subset of the object, as well as to indicate if curves that only part of them inside the specified area can make contribution to the reconstruction. After all the sampled intensity values in each of the sub-regions of the working space are computed, our system can perform a global blending process to put sub-regions together. This feature can reduce the number of calculations of the RBF method, and provide fast reconstruction by sculpting sketch curves. Moreover, CSG sculpting tools can be easily enforced accordingly. Fig. 7 shows an example. To reconstruct shapes from scattered data points where the number of constraints is extremely large and there is no gradient information available, we use the signed distance field approximation to compute the initial guess. The initial intensity value on the sample grids are computed by the fast-tagging algorithm introduced by [44] based on their signed distance to the data point constraints and we then use iterative solver to conduct a smoothing task. Two examples are shown in Fig. 8.

5. INTERACTIVE SCULPTING TOOLKITS FOR IMPLICIT PDES

This section details manipulating techniques for implicit PDE modeling. Fig. 9 shows a snapshot of our prototype system while manipulating a selected sketch curve. Besides the constraints, the coefficient functions $a(x, y, z)$, $b(x, y, z)$, and $c(x, y, z)$ can also be modified locally to deform the shape. They control the relative intensity blend-

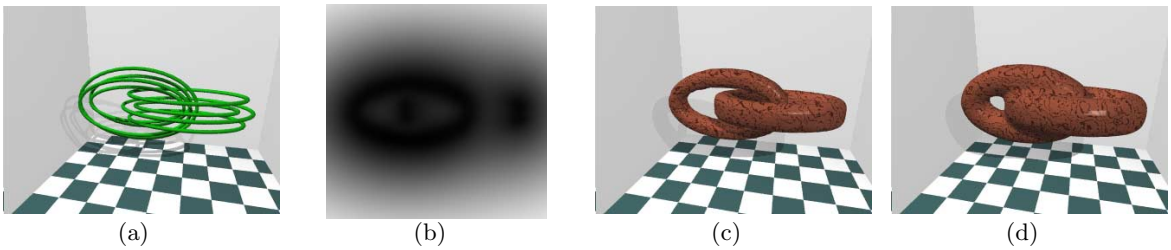


Figure 7: Example for performing the RBF initialization locally. (a) Two set of sketch curves; (b) a cross-section view; (c) and (d) are reconstructed implicit shapes at different iso-values.

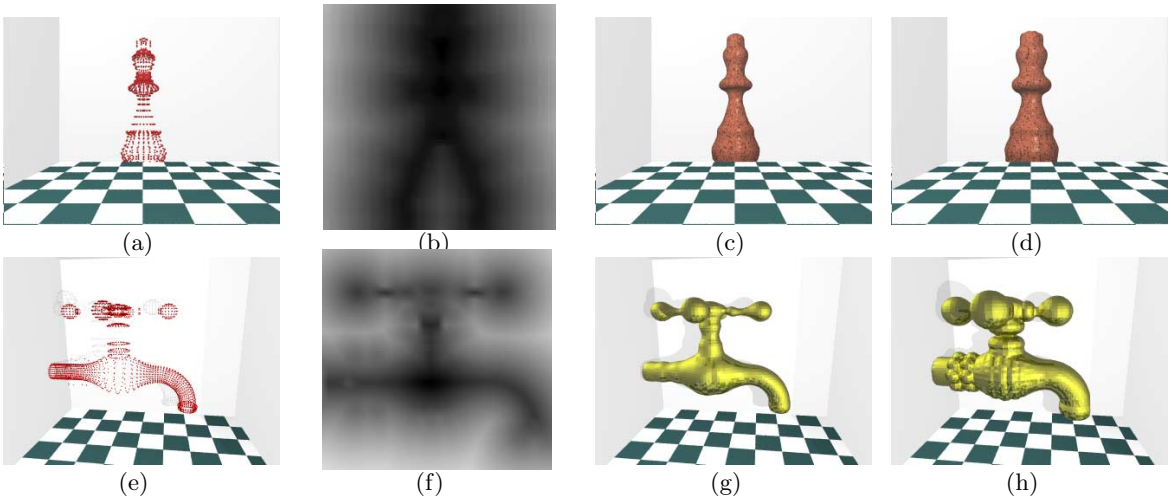


Figure 8: Examples of shape reconstruction from scattered data points. (b) and (f) are cross-section views; (c) and (d) are iso-surface at different intensity values of the object reconstructed from point set (a); (g) and (h) represent the reconstructed shape from data set (e) at different iso-values.

ing and the level of variable dependence among x, y, and z directions, thus they can be treated as generalized material properties over the volumetric working space. Consequently, users can control how the boundary and additional conditions influence the interior intensity distribution by modifying the length scale at arbitrary location (i.e., $a_{i,j,k}$, $b_{i,j,k}$, and $c_{i,j,k}$).

5.1 Sketch Curve Sculpting

Implicit objects can be defined by specifying a set of sketch curves which outline the rough shape of the objects. Moreover, our implicit PDE model provides interactive shape design toolkits to allow users manipulating the sketch curves in order to deform the underlying reconstructed implicit objects. The sketch curves defining the rough shape of the object can be obtained by either predefined curve network or B-spline curves from users' direct input. Our system allows users to modify the geometric shape, intensity value, as well as gradient directions of the sketch curves interactively in order to get the desired objects.

In order to modify the sketch curves smoothly, B-spline approximations for those curves are calculated at the initialization stage, then users can sculpt the curves interactively by manipulating the B-spline control points via translation and rotation. Because the reconstructed implicit object is required to interpolate those sketch curves which define its outlining shape approximately, it will follow the shape changes accordingly. Fig. 11 has an example of sculpting the shape of selected sketch curves. The intensity value of the sketch curves decides where the final shape of the implicit objects should pass through at the level set of its value. By modifying the intensity values of selected curves, users can manipulate the objects accordingly. Furthermore, according to the gradient definition, the intensity value increases along the gradient direction of sketch curves and decreases in the opposite direction in general. They provide information of the intensity distributions starting at the sketch curves and propagating to the neighborhood, which defines the inside and outside of the object. Without the predefinition of gradient directions, the PDE solution will be trivial. Therefore, gradient information of sketch curves are required for reconstructing a unique shape. Accordingly, changing the gradient direction at selected sketch curves means modifying directions where the intensity will increase or decrease in the implicit working space and will result in different implicit shapes. Our system allows users to specify the gradient direction of each individual sketch curve to construct different implicit PDE objects. Refer to Fig. 10 for examples of specifying and modifying gradient directions at the sketch curves. Without further specification, other examples in this paper have gradient directions pointing inward the curves by default.

5.2 Local Manipulation of Implicit PDE Solids

Usually the sketch curve sculpting will deform the entire reconstructed shape, which only offers global manip-

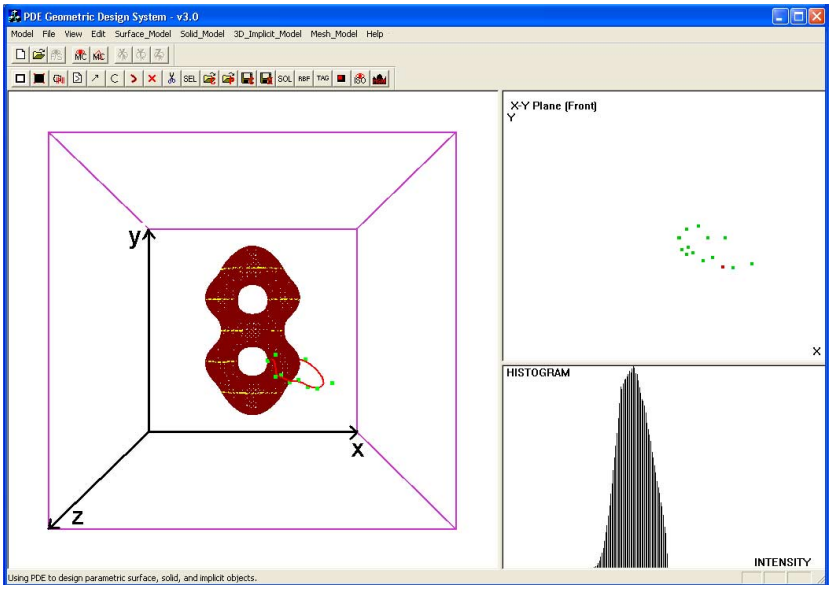


Figure 9: Snapshot of the interface of our implicit PDE system.

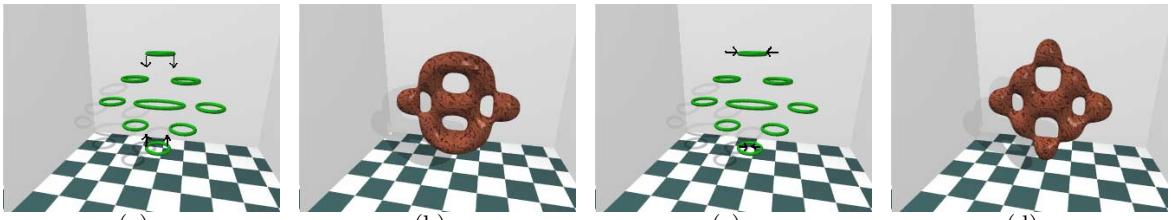


Figure 10: Examples for specifying and changing gradient directions of sketch curves. (a) and (c) are two sets of curves with same geometric shape but different gradient directions, where the arrows show intensity increasing directions; (b) and (d) are corresponding implicit objects.

ulation and is less intuitive for ordinary users to handle. Even with the specification of local areas of interests containing the sculpted sketch curve, the sculpting will affect all the points in the selected regions. Moreover, sometimes the input constraints alone can not guarantee a satisfactory solution of constructed shape. Therefore, direct modification on selected areas is desirable, especially when the overall recovered shape is satisfactory but minor changes in small localized areas are needed. Our system provides interactive tools of the intensity values modification in selected regions to sculpt the reconstructed shape. The modification will be enforced into (4). Using the aforementioned techniques, we can solve (4) to obtain the modified objects.

Traditional implicit techniques for data reconstruction does not support direct manipulations on the arbitrary locations in the volumetric working space. The changes on the predefined constraints will cause global deformation. It is more desirable to offer users editing functionalities on the interior properties with interactive interface. Besides the local RBF approximation for local sketch curve sculpting, our system also allows users to specify any interior region of the sampling grids, and only applies intensity changes within the specified region. Alternatively, we can freeze the selected region and disallow any changes in the specified region. In our system, this can be done through interactively specifying the maximum and minimum sampling grid in x, y, and z direction of the desired region in the sampling volumetric working space. Subsequently, any change within the region will have no influence on sampling points outside the region. The localized deformation can be easily achieved because only those equations corresponding to the points of the specified regions in (4) will be solved. In addition, the number of computations is reduced due to fewer number of equations involved in the local sculpting. In principle, all hard constraints can be viewed as some sort of local deformation. Fig. 11 shows an example of local deformation.

Users can also specify an iso-surface at a particular intensity value and use a cutting plane inside the volumetric working space to get a 2D iso-contour on the plane, then stretch, push, rotate the contour, as well as add desired intensity values at specified locations to modify the shape of the iso-surface and the intensity distribution of the interested areas. Refer to Fig. 11 for an illustrative example. We also offer several CSG sculpting tools such as sphere and cube to trim/extrude/sculpt the implicit objects by adding more constraints on the sampling grids of the working space. This is extremely useful for such situations when there are some minor changes needed to be done in some local small regions. Such sculpting tools make our system compatible with CSG-based implicit models by treating those models

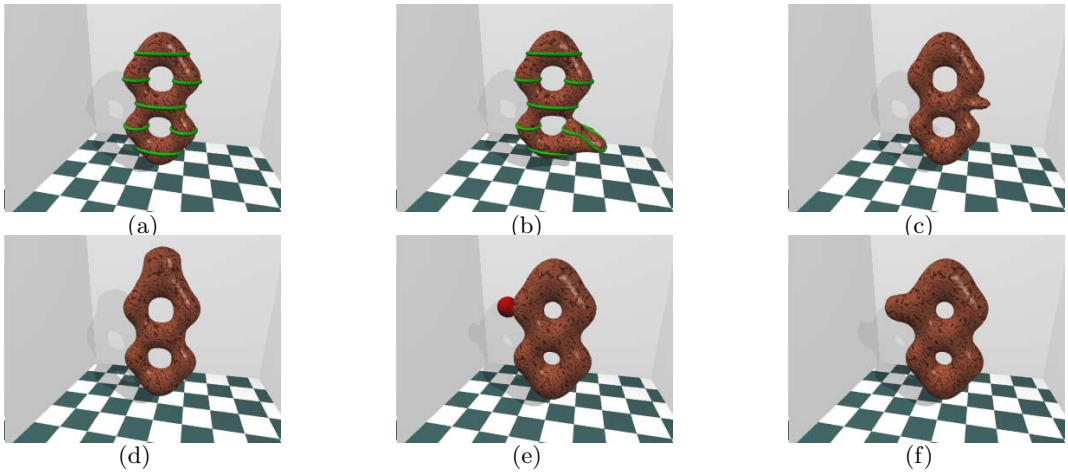


Figure 11: Examples of enforcing curve and direct manipulation constraints. (a) Original object with sketch curves; (b) deformed object by sculpting a selected curve; (c) changing an iso-contour; (d) deformed object subject to local region constraints; (e) adding a sphere in the working space; and (f) is the corresponding deformed object subject to (e).

as modeling tools. An example is shown in Fig. 11.

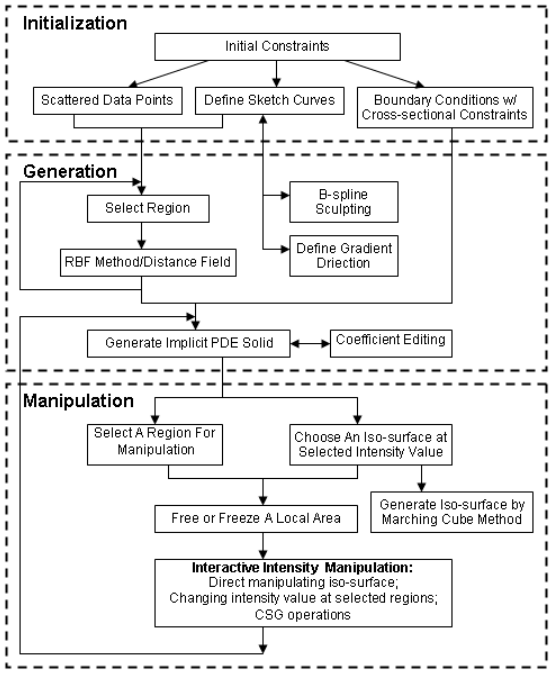


Figure 12: System architecture and functionalities.

6. IMPLEMENTATION AND DISCUSSION

We develop a prototype software that permits users to reconstruct geometric shapes defined by PDE-based implicit functions from a set of sketch curves or scattered data points. Our system also allows interactive manipulation of reconstructed implicit PDE objects with various intensity constraints in the volumetric working space. The interactive sculpting of implicit PDE objects can be obtained via modification of predefined conditions and interior operations. The system is written in Visual C++ and runs on Windows95/98/NT/2000/XP. Fig. 12 illustrates the architecture of our modeling environment for implicit PDE objects. In particular, our system provides the following functionalities:

Shape Reconstruction. Users can interactively input and edit scattered data points or sketch curves with specified intensity value, then the system uses the RBF method or distance field approximation to calculate intensity values on the sampling grids within the volumetric working space as the initial guess for the iterative solver of the discretized implicit PDE to obtain an approximated solution for implicit PDE objects satisfying these conditions. Our system can model both close and open implicit shapes.

Missing Information Recovery and Shape Blending. The underlying implicit PDE of our system provides a simple yet systematic mechanism to obtain the volumetric information satisfying specified constraints automatically. Such an advantage makes it possible to recover missing information of input datasets with our system. It can also be used to compute the connecting part between different objects in the working space which leads to shape blending.

Discrete Models. Our system supports implicit PDE objects obtained from solving the fourth order elliptic PDE using: (1) finite-difference discretization for the numerical solution of the fourth order elliptic PDE in 3D working space; and (2) RBF approximation at arbitrary sub-regions in the working space for modeling localized details and performance speedup.

Interactive and Direct Operations. Users can also work directly on the implicit PDE objects through: (1) sketch curve sculpting using B-spline manipulation; (2) gradient specification of selected curves; (3) local RBF approximation for improved time performance and interactive CSG manipulation; (4) interior deformation with additional constraints inside the working space; (5) iso-surface manipulation and direct manipulation of iso-contours at selected intensity values; and (6) local modification of blending coefficient functions.

We employ two iterative techniques (Gauss-Seidel and SOR) with multi-grid like techniques to solve the implicit

PDE subject to various constraints. Besides original datasets or predefined sketch curves, our system allows users to interactively define and sculpt sketch curves through B-spline manipulation and specify gradients at selected curves. These constraints provide more freedom to designers and make interactive design of implicit objects more cost-effective. We also enforce additional constraints directly inside the volumetric working space, apply local operations, and provide sculpting toolkits for the implicit objects, which facilitate the construction of implicit PDE objects of arbitrary topology. The PDE is solved by finite-difference techniques because they are simple, easy to implement, and suitable for complicated, flexible constraints. In general, the time and space complexity are increased with higher resolution as well as increased accuracy. Examples in this paper are rendered by POV-RAY.

Examples	Constraints	Initial	G-S	M-G
Fig. 3	169888	N/A	15.801	7.992
Fig. 4	840	23.855	60.048	32.134
Fig. 5 (a)	180	5.889	739.063	379.766
Fig. 5 (g)	720	18.872	760.244	416.312
Fig. 8 (a)	1219	267.925	N/A	113.432
Fig. 8 (e)	3154	359.657	N/A	148.283

Table 1: CPU-time (seconds) of different solvers for several examples of implicit PDE objects with different number of constraints.

Table 1 summarizes the numbers of constraints and CPU time of different numerical solvers for the implicit PDE examples when running on a Pentium 4 1.4GHz PC. The resolution of the working space is $64 \times 64 \times 64$ for Fig. 3 and $65 \times 65 \times 65$ for other examples. The stopping threshold (difference between two iteration steps) is 10^{-9}. "Initial" stands for the initial guess where we use RBF method for sketch curve datasets and fast-tagging approximation for the scattered data points input. G-S and M-G indicate the CPU time for solving the entire implicit PDE working space based on the initial guess using Gauss-Seidel iteration and multigrid improvement.

Although the initialization of the implicit models are time-consuming because of the approximation of the entire working space, the local sculpting afterwards will be interactive because only small number of sampling grids are involved. The time performance of RBF and fast-tagging algorithms depends on the number of constraints enforced, while the convergent speeds of iterative methods are mainly determined by the sampling rates of the implicit working space.

Despite the direct and powerful modeling advantages of our PDE framework, the major difficulty associated with our PDE techniques is the convergent speed of finite-difference approximation. Thus, faster numerical approximation techniques for solving PDEs need to be considered to improve the time performance of our PDE modeling system.

7. CONCLUSION

We have unified the popular implicit function techniques with the powerful parametric PDE framework to demonstrate more modeling advantages of the PDE-based paradigm. Our prototype system supports interactive shape design of implicit PDE objects through global and local deformation of scattered data points or sketch curves. The implicit PDE model can be defined as the solution of the fourth order elliptic PDE over a scalar intensity field with either scattered-point datasets or a set of sketch curves as generalized boundary and additional constraints. Our implicit PDE approach can also provide an approximation for the missing part in the working space with most of the intensity information already known. Our software environment affords users a set of interactive and direct shape modeling toolkits including: sketch curve sculpting and gradient manipulation, intensity value modification on selected regions, and iso-contour manipulation of specified intensity value inside the volumetric domain. These toolkits provide users an intuitive interface to model implicit PDE objects satisfying a set of design criteria and functional requirements. Our integrated approach and novel PDE techniques further expand the geometric coverage and the topological flexibility of the conventional PDE methodology to implicit functions, and forge ahead towards the realization of the full potential of PDE technology in solid modeling and other visual computing fields.

8. ACKNOWLEDGMENTS

This research was supported in part by the NSF ITR grant IIS-0082035, the NSF grant IIS-0097646, Alfred P. Sloan Fellowship, and Honda Initiation Award.

9. REFERENCES

[1] A. Bærentzen and N. Christensen. Volume sculpting using level-set method. In *Shape Modeling International, Banff, Alberta, Canada*, pages 175–182, 2002.

[2] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *SIGGRAPH 2000, New Orleans, USA*, pages 417–424, 2000.

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

[4] J. Bloomenthal and B. Wyvill. Interactive techniques for implicit modeling. *Computer Graphics*, 24(2):109–116, 1990.

[5] M. I. G. Bloor and M. J. Wilson. Generating blend surfaces using partial differential equations. *Computer Aided Design*, 21(3):165–171, 1989.

[6] M. I. G. Bloor and M. J. Wilson. Using partial differential equations to generate free-form surfaces. *Computer Aided Design*, 22(4):202–212, 1990.

[7] M. I. G. Bloor and M. J. Wilson. Functionality in solids obtained from partial differential equations. *Computing Suppl. 8*, pages 21–42, 1993.

[8] D. Breen and R. Whitaker. A level-set approach for the metamorphosis of solid models. *IEEE Transactions on Visualization and Computer Graphics*, 7(2):173–192, 2001.

[9] J. Carr, R. Beatson, J. Cherrie, T. Mitchell, W. Fright, and B. McCallum. Reconstruction and representation of 3D objects with radial basis functions. In *SIGGRAPH 2001, Los Angeles, USA*, pages 67–76, 2001.

[10] D. Cohen-Or and D. Levin. Three-dimensional distance field metamorphosis. *ACM Transactions on Graphics*, 17(2):116–141, 1998.

[11] B. Cutler, J. Dorsey, L. McMillan, M. Müller, and R. Jagnow. A procedural approach to authoring solid models. In *SIGGRAPH 2002, San Antonio, Texas*, pages 302–311, 2002.

[12] M. Desbrun and M.-P. Cani-Gascuel. Active implicit surfaces for animation. In *Graphics Interface 1998*, pages 143–150, 1998.

[13] M. Desbrun, N. Tsingos, and M.-P. Gascuel. Adaptive sampling of implicit surfaces for interactive modelling and animation. *Computer Graphics Forum*, 15(5):319–325, 1996.

[14] H. Du and H. Qin. Direct manipulation and interactive sculpting of PDE surfaces. *Computer Graphics Forum*, 19(3):C261–C270, 2000.

[15] H. Du and H. Qin. Dynamic PDE surfaces with flexible and general constraints. In *Pacific Graphics 2000, Hong Kong*, pages 213–222, 2000.

[16] H. Du and H. Qin. Integrating physics-based modeling with PDE solids for geometric design. In *Pacific Graphics 2001, Tokyo, Japan*, pages 198–207, 2001.

[17] D. S. Ebert, F. K. Musgrave, P. Prusinkiewicz, J. Stam, and J. Tessendorf. *Simulating Nature: From Theory to Practice*. SIGGRAPH 2000 Course Notes 25, 2000.

[18] E. Ferley, M. Cani, and J. Gascuel. Practical volumetric sculpting. *The Visual Computer*, 16(8):469–480, 2000.

[19] N. Foster and D. Metaxas. Realistic animation of liquids. In *Proceedings of GI 1996*, pages 204–212, 1996.

[20] N. Foster and D. Metaxas. Modeling the motion of hot, turbulent gas. In *SIGGRAPH 1997, Los Angeles, CA USA*, pages 181–188, 1997.

[21] S. Frisken, R. Perry, A. Rockwood, and T. Jones. Adaptive sampled distance fields: A general representation of shape for computer graphics. In *SIGGRAPH 2000, New Orleans, USA*, pages 249–254, 2000.

[22] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *SIGGRAPH 1992*, pages 71–78, 1992.

[23] J. Hua and H. Qin. Haptic sculpting of volumetric implicit functions. In *Pacific Graphics 2001, Tokyo, Japan*, pages 254–264, 2001.

[24] J. Hua and H. Qin. Dynamic implicit solids with constraints for haptic sculpting. In *Shape Modeling International 2002, Banff, Alberta, Canada*, pages 119–128, 2002.

[25] W. Lorensen and H. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *SIGGRAPH 1987*, pages 163–169, 1987.

[26] B. Morse, T. Yoo, P. Rheingans, D. Chen, and K. Subramanian. Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. In *Shape Modeling International 2001, Genova, Italy*, pages 89–98, 2001.

[27] S. Muraki. Volumetric shape description of range data using blobby model. *Computer Graphics*, 25(4):227–235, 1991.

[28] K. Museth, D. Breen, R. Whitaker, and A. Barr. Level set surface editing operators. In *SIGGRAPH 2002, San Antonio, Texas, USA*, pages 330–338, 2002.

[29] R. Perry and S. Frisken. Kizamu: A system for sculpting digital characters. In *SIGGRAPH 2001, Los Angeles, USA*, pages 47–56, 2001.

[30] W. H. Press, S. A. Teulolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 1993.

[31] A. Raviv and G. Elber. Three dimensional freeform sculpting via zero sets of scalar trivariate functions. In *Proc. of 5th ACM Symposium on Solid Modeling and Applications, Ann Arbor, Michigan, United States*, pages 246–257, 1999.

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

[33] R. Schneider and L. Kobbelt. Generating fair meshes with G^1 boundary conditions. In *Geometric Modeling and Processing Conference Proceedings, 2000*, pages 251–261, 2000.

[34] S. Sclaroff and A. Pentland. Generalized implicit functions for computer graphics. *Computer Graphics*, 25(4):247–25, 1991.

[35] J. Stam. Stable fluids. In *SIGGRAPH 1999, Los Angeles, CA, USA*, pages 121–127, 1999.

[36] G. Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, 1986.

[37] G. Turk, H. Q. Dinh, J. O'Brien, and G. Yngve. Implicit surfaces that interpolate. In *Shape Modeling International 2001, Genova, Italy*, pages 62–71, 2001.

[38] G. Turk and J. O'Brien. Shape transformation using variational implicit functions. In *SIGGRAPH 1999, Los Angeles, CA, USA*, pages 335–342, 1999.

[39] G. Turk and J. O'Brien. Modeling with implicit surfaces that interpolate. *ACM Transaction on Graphics*, 21(4):855–873, 2002.

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

[41] R. Whitaker and D. Breen. Level-set models for the deformation of solid objects. In *Conference of Implicit Surface 1998, Seattle, USA*, pages 19–36, 1998.

[42] A. Witkin and P. Heckbert. Using particles to sample and control implicit surfaces. In *SIGGRAPH 1994*, pages 269–277, 1994.

[43] J. J. Zhang and L. You. Surface representation using second, fourth and mixed order partial differential equations. In *Shape Modeling International 2001, Genova, Italy*, pages 250–256, 2001.

[44] H. K. Zhao, S. Osher, and R. Fedkiw. Fast surface reconstruction using level set method. In *IEEE Workshop on Variational and Level Set Methods (VLSM 01), Vancouver, Canada*, pages 194–202, 2001.

[45] H. K. Zhao, S. Osher, B. Merriman, and M. Kang. Implicit and non-parametric shape reconstruction from unorganized points using variational level set method. *Computer Vision and Image Understanding*, 80(3):295–319, 2000.