EG UK Theory and Practice of Computer Graphics (2005)
L. Lever, M. McDerby (Editors)

Virtual Sculpting Using Implicit Surfaces with Scattered Data Interpolation

K. Zhang , R. A. Noble , R. J. McDermott and A. Wilson

School of Computing, The Robert Gordon University, Aberdeen, United Kingdom

Abstract

This paper presents a virtual sculpting system that shows an approach to the interactive deformation of soft objects. The soft objects are represented by implicit surfaces which are visualized using a constrained particle method. Those particles are arranged in an efficient cubical structure in which its processing speed is faster than some other surface sampling algorithms. The sculpted result can be stored in a compact form by interpolating the constrained particles using scattered data interpolation. The user can control the complexity of the sculpted shape by specifying the sampling density. In addition, the way in which the tool deforms the clay can be controlled explicitly by some control parameters. The surfaces are displayed with a stereoscopic viewer and the virtual tools are manipulated using a spaceball with 6 degrees of freedom.

Key words: *Implicit surface, Virtual sculpting, Scattered data interpolation, Soft object, Stereoscopic.*

Categories and Subject Descriptors (according to ACM CCS): I.3.2 [Computer Graphics]: Graphics Systems; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

1. Introduction

The use of implicit formulations to describe the geometry of objects has become increasingly popular in computer graphics nowadays. Implicit surface has many advantages over methods using parametric surfaces where the points of the object are given explicitly by maps from the parametric space to the object surface. The most significant one is that it is much easier to smoothly deform objects represented by implicitly defined surfaces. Implicit surfaces are always closed and can always be blended smoothly and as such, they present an effective approach to interactive sculpting.

Implicit surfaces are two dimensional, geometric shapes that exist in three-dimensional space [BBB*97]. They are surfaces defined by constant values on a potential field. In implicit surfaces, the points belonging to the object are given indirectly through a classification function which maps the points in object space to a scalar value. The surface is defined by the set of points x that satisfy the equation:

$$f(x) = C \qquad x \in R^3 \qquad (1)$$

where f is a potential function and C a constant value. It is much easier to use implicit surfaces as a mathematical representation to modelling soft deformable objects such as a piece of clay than parametric surfaces. Another useful characteristic of implicit models is that it is always possible to build new surfaces and shape effects by adding or subtracting the functions that define them. However, in order to use it as a model for a sculpting system, we suppose that it is possible to build a complicated shape at interactive speed. There are two major steps to such a sculpting system. First, model the interaction between the tool and the clay and secondly store the results of many such actions in a compact form [NM01].

Gascuel presented an approach to modelling deformable objects in [CG93]. This approach was extended by Noble and McDermott by incorporating a control parameter [NM01]. In their approach, a precise contact surface which maintains C^1 continuity can be produced. This approach can be used to simulate the combined sculpting actions of more than one tool [NZM04].

In this paper, an approach to storing the sculpted surface in a compact form is investigated. This approach interpolates the constrained particles in [NZM04] and generates a single smooth polynomial implicit function that represents the

sculpted surface. Section 2 discusses some previous works of implicit surfaces. In Section 3, the mathematical method concerned with the tool-clay interaction is discussed. Section 4 introduces a more efficient particle management data structure that is used in this system. The scattered data interpolation method this system employs is discussed in Section 5. Some of the implement details and experimental results are given Section 6. In Section 7, some problems associated with this system and some possible further improvements are investigated.

2. Related Works

2.1. Implicit surfaces

Implicit surfaces can be divided into three different classes. One of them is a surface that is generated from a set of point charges or line charges, for example, blobby or meta-ball surfaces [Bli82]. The implicit function of this kind of surface is the summation of the Gaussian functions. Another class of implicit surfaces is the algebraic surface which is described by a polynomial function. A good introduction to this topic can be found in [BBB*97]. Convolution surface [BS91] is the most complicated implicit surface. The surface it generates is the convolution of a filter kernel with a polygonal skeleton. Convolution surfaces can be used to model complex shapes with skeletal structures.

One of the disadvantages of an implicit surface is that it is not as easy to visualise as a parametric surface. In this paper, a physically-based sampling algorithm introduced by Witkin and Heckbert in [WH94] is used. It is also used to generate the scattered data sets which the interpolation algorithm works on. Some of the key issues will be discussed in Section 4.

2.2. Virtual sculpting

Virtual sculpting is a method of deforming a virtual object in a free-form manner. It has been a subject of interest to many researchers. In recent years, numerous techniques have been developed for constructing a virtual sculpting system. Most of the methods developed concentrate on deforming free-form surfaces by modifying either the control points or the sample points one by one.

An early paper on virtual sculpting was by Bill and Lodha [BL95]. They proposed a sculpting system based on the polygonal model. In this system, the user may alter the topology, but he has to specify the new connectivity at a triangle level. Galyean and Hughes use a better data storage method in which the clay is recorded as a set of values on a rectangular grid [GH91]. Turk and O'Brien presented another interactive deformation method based on Witkin's constrained particle algorithm in [TO02]. In this approach, the clay is displayed and controlled using the constrained particle method. The resulting implicit function is derived by using variational interpolation. This method performs deformation by directly manipulating the clay instead of deforming the clay using other implicit objects.

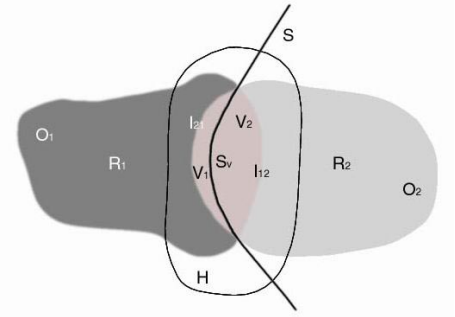

Figure 1: *The areas of tool and clay interaction*

Matsumiya *et al* proposed another immersive sculpting system using implicit surfaces [MTY00]. In this paper, a free-form interactive modeling technique based on the metaphor of clay is presented. It is an immersive modeling system which enables a user to design intuitively 3D solid objects with curved surfaces by using an electronic glove. Wong et al presents another sculpting method based on the use of an electronic glove [WLM00]. The main idea of their system is to make use of the data collected from the electronic glove to create a parametric surface controlled by the hand, which is basically an Open-Uniform bicubic B-Spline tensor product surface. The sculpting action is performed by mapping the object to be sculpted onto the surface of the controlling hand.

Adzhiev *et al* introduced another approach to doing computer-added sculpting concerned with the creation and modification of digital models based on physical abstract sculptures [ACKP05]. A specialised computer language named HyperFun is introduced which facilitates the modelling of complex objects. Gain *et al* presented a spatial deformation method called warping sculpting in [GM05]. In this approach, the object is deformed by warping the surrounding space.

3. Tool and Clay Interaction

The first task for developing a sculpting system is to simulate the deformation of the virtual clay when clay and tool are interacting.

For two objects, the interaction can be illustrated by Figure 1 [CG93]. O_1 and O_2 are two implicit objects where O_1 represents the clay and O_2 represents the tool. The two objects are defined by:

$$f_i(x) = c \qquad x \in R^3$$

S represents the surface on which both O_1 and O_2 have the

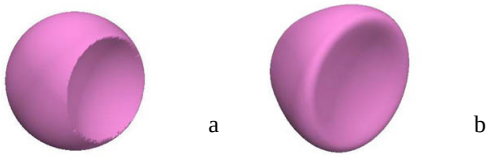

Figure 2: *The contact surface of the clay. (a) Without β function, α = -2; (b) With β function, α = -2.*

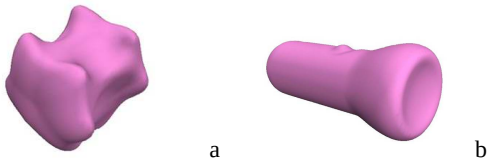

Figure 3: *The sculpted shape of the clay with multiple sculpting actions. (a) Five sculpting on a spherical clay, $\alpha = -1$; (b) A torch sculpted using a rod clay*

same function value. I_{ij} is the part of undeformed surface O_i inside O_j. V_i is the part of O_j between S and I_{ji}. The area of V_1 and V_2 altogether is called contact region. R_i is called intact region and H is called propagation region.

3.1. Modelling of the clay

Suppose there are only two objects. As shown on Figure 1, O_1 represents the virtual clay and O_2 represents the virtual tool. When O_1 and O_2 make contact, both objects are deformed and produce a common contact surface S_v.

For O_1, the surface deforms from I_{12} to S_v. For O_2, the surface deforms from I_{21} to S_v (Figure 1).

As discussed in [NZM04], the common contact surface functions of two objects can be formulated as:

$$h_1(x) = f_1(x) + \beta_2(x) \cdot g_2(x) \tag{2}$$

$$g_2(x) = \begin{cases} \alpha_2(f_2(x) - c) & x \in V_1 \\ 0 & Otherwise \end{cases} \tag{3}$$

and the deformed shape of the clay after multiple sculpting actions can be represented as:

$$h_1(x) = f_1(x) + \sum_{i=2}^{n} \beta_i \cdot g_i(x) \tag{4}$$

$$g_i(x) = \begin{cases} \alpha_i(f_i(x) - c) & x \in V_1 \\ 0 & Otherwise \end{cases} \tag{5}$$

In both representations,

$$\beta_i(x) = \begin{cases} 1 & x \in V_1 \\ f_i(x)/c & Otherwise \end{cases}, \quad \alpha \in [-\infty, +\infty]$$

In the above equations, $h_1(x)$ and $f_1(x)$ represent the resulting clay function and original clay function respectively. $f_i(x)$ represents the function of virtual tools when $i \geq 2$. The definition of $g_i(x)$ guarantees that the surface is C^0 continuous at the join of the propagation region and the contact region. As α varies from $-\infty$ to 0, a concave depression is produced and as α varies from 0 to $+\infty$ a convex bulge is produced. Figure 2a shows the deformed shape without using $\beta_i(x)$ function. The $\beta_i(x)$ function makes the resulting function C^1 continuous. Keeping $\beta_i(x) = 1$ inside the tool guarantees that the contact surface is maintained. The deformed function h_1 with $\beta_i(x)$ function integreted is illustrated by Figure 2b.

Equation 4 can be used to simulate the combined sculpting actions of more than one tool. The total deformation is the sum of the deformations of the individual sculpting actions. Therefore, h_1 is still C^1 continuous in Equation 4. Figure 3 shows some sculpted shapes of multiple sculpting using the previous sculpting prototype system. More detailed discussion about modelling the multiple tools actions can be found in [NZM04].

4. Constrained Particles

A visualization method called constrained particles is used in this system to sample the implicit surface. This method is first introduced by Witkin and Hecbert in [WH94]. The basic idea of their method is to solve a constraint problem by defining an objective function and deriving some constraints from the implicit function. This algorithm starts with just a few particles which spread across the surface. At the same time, they will keep fissioning until all the particles get to equilibrium. When all the particles are in stable states, they are evenly distributed on the surface. This method can be used to display a surface with good sampling density if the proper parameters are given. It provides an algorithm that visualizes the implicit surface at interactive speed. Witkin's method was originally designed for manipulating two kinds of particles: control points and floaters. However, it is only used in this system to visualize implicit surface. Therefore, all the particles are floaters and the shape parameter is treated as a constant. The implementation details of Witkin's method in this sculpting system can be found in [NZM04]. Moreover, the constrained particles are also used as the scattered data point for the scattered data interpolation algorithm.

4.1. The PDS structure

In Witkin's algorithm, each particle is given independent control. The only way they "communicate" with others is

to use another control called repulsion radius. For an individual particle, all the particles within the repulsion radius have non-negligible effect to it. A Particle Division System (PDS) is an enhancement of constrained particles. This idea was first introduced by Heckbert in [Hec97].

It comprises a more efficient data structure which arranges all the particles in order and a series of operations that may selectively pick particles they need. Therefore, some expensive calculations apply only to particles that the query operation picks up. That means we can save computation time by simply ignoring those particles which are distant to the particle being considered. Since the repulsion radius σ is the standard deviation of the Gaussian energy function, we say that if the distance between two particles is more than 3σ, the energy they give each other is negligible.

In the sculpting program, 3D space is divided into $2^k \times 2^k \times 2^k$ cubes, each of which consists of a linked list of the particles within it. The k should be no more than 6. During the running process of this program, queries of looking up adjacent particles can be answered quickly with this grid data structure by finding all cells that intersect or lie interior to the sphere of radius r centred at p.

The best performance can be achieved by selecting the proper cube size so that the list traversal cost and the calculation cost are balanced. If the cell size is very small, the list tranversal cost will be very high since many cells must be visited. If the cell size is too big, the advantage of the PDS will lost.

Table 1 shows that the PDS version is almost two times faster than non-PDS version.

Radius (Units)	Number of particles	PDS (Seconds)	Non-PDS (Seconds)
2	54	2	2
3	110	4	7
4	224	6	13
5	307	12	27

Table 1: *The Speed of PDS and non-PDS version*

5. Scattered Data Interpolation

5.1. The problem

The ideas introduced in Section 3.1 and 3.2 simulate the tool-clay interaction by manipulating their potential fields. However, the clay acts like an elastic balloon. It deforms when the tool penetrates the clay and regains its original shape after the tool leaves. Therefore, in order to deform the soft object, a new implicit function for the deformed soft-object needs to be constructed.

In the previous approach [NZM04], the implicit function of the deformed shape is calculated by using both the implicit function of the clay and some invisible tools at the position where sculpting actions are made. Therefore all of the invisible tools have to be kept in order to calculate the sculpted shape. It has an obvious problem that the number of terms in Equation 4 becomes larger if more sculpting actions are made.

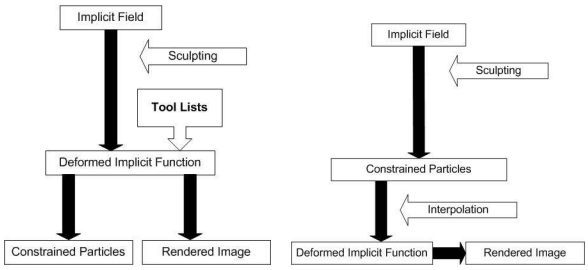

Figure 4: *Two methods of constructing the deformed implicit function (Left) The original method; (Right) Surface Reconstruction*

If N represents the number of terms in the field function of the sculpted shape, then:

$$N = N_{f_1(x)} + N_s \tag{6}$$

where $N_{f_1(x)}$ is number of terms of the original clay function and N_s is the number of sculpting actions. As N_s increases, the whole process will be slowed down and more memory will be required.

To prevent this, a compact and concise representation of the deformed shape is needed. The complexity of this representation should be independent to the number of sculpting actions.

5.2. Surface reconstruction

One possible solution to the problem is to use surface reconstruction technique. The idea is that the deformed implicit function is always constructed from the positional data set that is used to visualize the implicit field, instead of the original implicit functions. The resulting representation is an interpolated algebraic function generated using scattered data interpolation method. The difference between this method and the original method is illustrated in Figure 4. The major differences of these two approaches are:

1. Tool lists are not needed for the surface reconstruction method.

2. The deformed implicit function generated from the surface reconstruction method is independent of the implicit function of the original clay and tools.

3. A new interpolation algorithm is needed for the new approach. This algorithm is used to construct a smooth, continuous and closed implicit function from the positional data set.

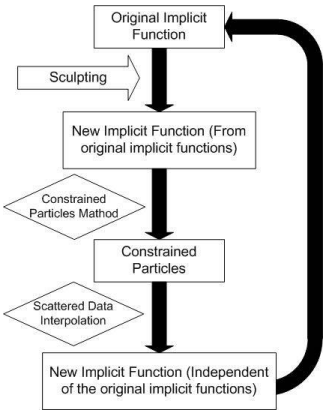

Figure 5: *The surface reconstruction algorithm*

4. The original deformation algorithms can still be used in the new approach.

5.3. Scattered data interpolation

In this sculpting system, a variational scattered data interpolation method [TO02] is used to construct the resulting surfaces. These surfaces are created by interpolating the particles through which the surface should pass, and also the points that are interior or exterior to the surface.

Mathematically, the scattered data interpolation can be described as following: Given a set of points p_i, and its correspond function value v_i ,find a function f so that:

$$f(p_i) = v_i \qquad i = 1,2,\cdots,n \tag{7}$$

Figure 5 illustrates how the surface reconstruction method works in this sculpting system. With all the constrained particles, the variational scattered data interpolation method is employed to find a proper function whose trajectory passes through every particle smoothly. For any two particles, there are infinite ways to join them but the smoothest one could be selected if the smoothness is properly defined. In this paper, smoothness in 2D is defined by an energy function:

$$E(f) = \int_{\Omega} f_{xx}^2(x) + 2f_{xy}^2(x) + f_{yy}^2(x)dx \tag{8}$$

This energy function is introduced in surface reconstruction by Turk in [TO02]. In this equation, Ω is the region where the interpolation applies; $f_{xx}^2(x)$ is the second partial derivative in the x direction and the other two terms are similar partial derivatives. It is clear that any unsmooth segment of interpolated surface will result in a large energy value. Therefore, the interpolation solution is the function $I(x)$ that pass through all the constraint particles and that has the smallest energy value.

In the interpolation methods used in this system, the candidate functions for representing an implicit surface take the form:

$$I(x) = \sum_{i=1}^{n} \lambda_i \phi_i(\|x - x_i\|) + P(x) \tag{9}$$

where $x \in R^3$ represents a general point; x_i is the constrained particles which the resulting surface should pass; λ_i is the weight for constraint point i and $P(x)$ is a low degree linear polynomial with respect to variables x,y,z. Alternatively, $P(x)$ could be a linear polynomial represented as a constrained ellipsoid. ϕ_i is a radial basis function which can automatically solve differential equations, such as the energy function Equation 8. The radial basis functions have been widely used for function approximation [LWP*04] [TO02]. The function used in this paper is a commonly used radial basis function for three dimensional interpolation, $\phi(x) = |x|^3$.

To solve the set of λ_i and the coefficient of linear polynomial function $P(x)$ that will satisfy the interpolation constraints $f(p_i) = v_i$, the right side of Equation 9 can be substituted for $f(p_i)$, which gives:

$$I(x_i) = \sum_{i=1}^{k} \lambda_i \phi(\|p_i - p_j\|) + P(p_i) \tag{10}$$

Equation 10 is a linear system:

$$AX = V \tag{11}$$

where

$$A = \begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} & \cdots & \phi_{1k} & 1 & p_1^x & p_1^y & p_1^z \\ \phi_{21} & \phi_{22} & \phi_{23} & \cdots & \phi_{2k} & 1 & p_2^x & p_2^y & p_2^z \\ \phi_{31} & \phi_{32} & \phi_{33} & \cdots & \phi_{3k} & 1 & p_3^x & p_3^y & p_3^z \\ \cdots & \cdots & \cdots & \cdots & \cdots & 1 & \cdots & \cdots & \cdots \\ \phi_{k1} & \phi_{k2} & \phi_{k3} & \cdots & \phi_{kk} & 1 & p_k^x & p_k^y & p_k^z \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ p_1^x & p_2^x & p_3^x & \cdots & p_k^x & 0 & 0 & 0 & 0 \\ p_1^y & p_2^y & p_3^y & \cdots & p_k^y & 0 & 0 & 0 & 0 \\ p_1^z & p_2^z & p_3^z & \cdots & p_k^z & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$X = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \cdots \\ \lambda_k \\ a \\ b \\ c \\ d \end{bmatrix} \quad \textit{and} \quad V = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \cdots \\ v_k \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Using Equation 11, the weights of the radial basis functions and the coefficient of the linear part (a,b,c,d) can be solved and the resulting function is generated. However, the system gives poor interpolating result if only the constrained particles are used, i.e.:

$$v_i = C \qquad i = 1,2,\cdots,n \tag{12}$$

where C is a constant value. That is because the interpolation algorithm does not have any information about the interior/exterior of the surface. In the sculpting system, some other constraints called internal/external constrained particles are introduced. These particles have potential values that differ from the potential at the surface. This helps determine the interior/exterior of the surface. These particles are found by applying a small displacement from the constrained particles along the surface normals. For each constrained particle, there is one interior and one exterior particle corresponding to it. Therefore, the number of terms of the interpolating function with n constrained particles is $3n+4$.

5.4. Improvement with surface reconstruction

The surface reconstruction routine is employed to make the resulting function compact and independent of the number of sculpting actions.

With the surface reconstruction method, the surface deformation routine (Equation 2, 3, 4, 5) can be described as following:

For the first sculpting action:

$$h_1^1(x) = f_1(x) + \beta_2 \cdot g_2(x)$$
$$g_2(x) = \alpha_2(f_2(x) - c)$$
$$I^1(x) \Leftarrow h_1^1(x)$$

For the rest of the sculpting actions ($i \geq 2$):

$$h_1^i(x) = I^{i-1}(x) + \beta_{i+1} \cdot g_{i+1}(x)$$
$$g_{i+1}(x) = \alpha_{i+1}(f_{i+1}(x) - c)$$
$$I^i(x) \Leftarrow h_1^i(x)$$

where $h_1^i(x)$ is the resulting clay function after i sculpting actions and $I^i(x)$ is the interpolated clay function after i sculpting actions. The arrow means the left hand side function is calculated by interpolating the constrained particles that sample the right hand side function.

Since $I^i(x)$ takes the form of Equation 9, its complexity only depends on the number of data points being used in the interpolation algorithm rather than the number of sculpting actions. Therefore, the performance of the sculpting system will not degrade as more sculpting actions are made.

5.5. Controlling the number of particles

In the sculpting system, users are allowed to specify the number of particles they need to visualize and interpolate the virtual clay. This is required by the interpolation algorithm so that the interpolated function (Equation 9) has a fixed number of terms. In Witkin's paper [WH94], arbitrarily specifying the number of particles is not possible. Therefore, it is modified by introducing a feedback control parameter S and changing the fission/death conditions.

The key is that Witkin's algorithm manages particles by fissioning and killing them. Both birth and death depend upon the particles' radius of influence σ^i in the form:

Fission if $\sigma^i > \hat{\sigma}$

Kill if $\sigma^i < \theta \cdot \hat{\sigma}$

where $\hat{\sigma}$ is the constant desired repulsion radius and θ is a constant. The particles will be in equilibrium if the distances between all the particles are equal to the desired repulsion radius so that the energy between them is balanced. The value $\hat{\sigma}$ effectively controls the packing density and hence the number of particles.

In this system, by introducing a variable S, the particle density can be changed by replacing the desired repulsion radius by $S\hat{\sigma}$. The S is a feedback function depends on the difference between the actual number of particles, n and the desired number of particles N. The number of particles can be driven to the desired value by updating S each iteration by:

$$S = S + S^2 \cdot (n - N) \cdot t \tag{13}$$

Where t is a user specified variable which controls the speed of approaching to N.

In the sculpting system, the complexity of the resulting shape is limited by specifying a given number of particles. It is important to use a sufficient number of particles. Otherwise surface resolution will be limited and surface details will be lost. Figure 6 shows the reconstructed object from a same shape using different numbers of particles. The resolution problem is one of the problems need to be addressed in future research.

6. Sculpting System and Results

This program is an improved version of the prototype system introduced in [NZM04]. It performs sculpting in two stages. First, Equation 2 and 4 are used to deform the clay as the sculpting take place. The constrained particle algorithm is then used to visualize the surface. The constrained particles sample the potential field and, at the same time, collect information needed for interpolation. Secondly, each time the user performs a sculpting action, the interpolation algorithm is performed to find the new clay function.

After the user finishes sculpting, the sculpted surface may be rendered using the polygonization algorithm. In addition, the interpolated sculpted function can be exported to some format which can be used in rendering packages. Users can go back to the sculpting stage and modify the clay at any time.

Figure 7 shows the interface of the sculpting system when the user is sculpting on the clay. The program runs on a Pentium 4 / 2.66GHz PC with a stereoscopic attachment and a 3D spaceball. It is able to perform sculpting actions at interactive speed with fully 3D control using the spaceball which

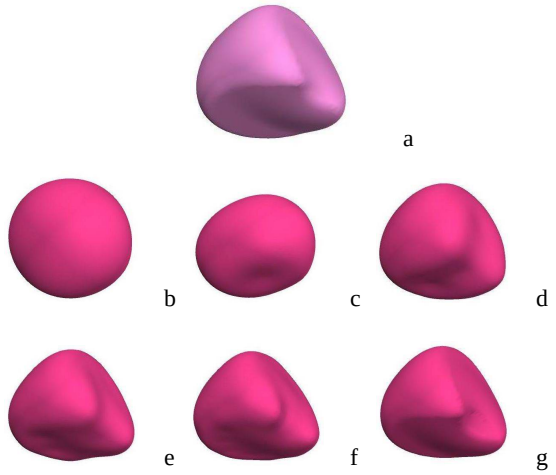

Figure 6: *Surface reconstructed from different number of constrained particles. (a) Original shape; (b) 5 particles; (c) 10 particles; (d) 20 particles; (e) 40 particles; (f) 60 particles; (g) 80 particles.*

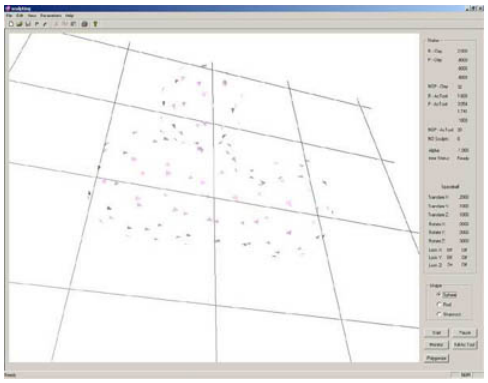

Figure 7: *User interface of the sculpting system*

has 6 degrees of freedom. A pair of shutter glasses is used to help the users recognize the shape in 3D when they are sculpting.

Figure 8 shows a monster head that was created using this system using spherical clay using 120 particles. Figure 9 shows the particle sampled monster head at an earlier stage of sculpting. These two images show that deformed surface can be clearly approximated by the interpolated algebraic function. An numerical evaluation to measure how closely the interpolated surfaces matches the original surface is currently underway.

Figure 8: *A monster head sculpted from spherical clay. Number of particles = 120.*

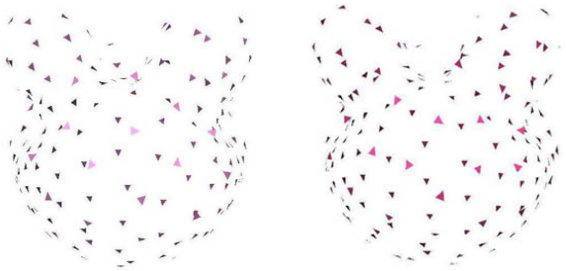

Figure 9: *Particle sampled monster head. Number of particles = 120. (Left) Sculpted shape with phantom tools without using interpolation. (Right) Sculpted shape after interpolation.*

7. Future Work

In this system, a user sculpts on a particle-sampled surface rather than a solid surface. It is possible for a user to perform a sculpting action in the gap between particles when they are sparsely distributed. As a result, the particles can not "feel" the full local deformation of the potential function and the interpolated function will not be calculated accurately. Figure 10 shows a monster head sculpted from spherical clay with inadequate number of particles used for interpolation. Basically, it is a resolution issue. Although the user sculpts the actual surface in terms of the underlying func-

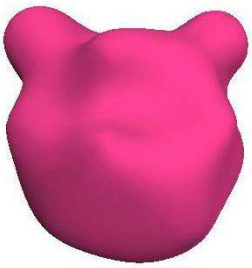

Figure 10: *A monster head sculpted from spherical clay with inadequate number of particles. Number of particles = 30.*

tion, the display and the interpolation are based on the particles. The sampled surface, given by the particles, therefore suffers from aliasing, or a limited resolution and the interpolated function is not entirely accurate.

To maintain the maximum possible accuracy, it is important that the constrained particles are moved to positions which minimize the visualization and interpolation error.

Since high detail is associated with high curvature, we propose to investigate controlling the particles to bias them towards the higher curvature regions and away from "flatter" low curvature regions.

The previous section showed how to control the total number of particles on the surface and we will investigate modifying the control to achieve the required bias.

8. Conclusion

This paper presents an approach to doing interactive sculpting using implicit surfaces. One major advantage of this system is that the resulting shape is stored in a single compact implicit function whose complexity is independent of the number of sculpting actions and is controllable by the user. The constrained particles are not only used to visualize the implicit object at interactive speed, but also used as the input data set for the surface reconstruction algorithm. The particles are arranged and managed in a more efficient data structure which doubles the processing speed. In addition, both 3D control devices and the stereographic viewers are used to give the user a real 3D experience. Further work needs to be done on investigating the resolution problem and complicated tool actions.

References

[ACKP05] ADZHIEV V., COMNINOS P., KAZAKOV M., PASKO A.: Functionally based augmented sculpting. *Computer Animation and Virtual Worlds 16*, 1 (2005), 25–39.

[BBB*97] BLOOMENTHAL J., BAJAJ C., BLINN J., CANI-GASCUEL M.-P., ROCKWOOD A., WYVILL B., WYVILL G.: *Introduction to Implicit Surfaces*. Morgan Kaufmann, San Francisco, California, USA, 1997.

[BL95] BILL J. R., LODHA S. K.: Sculting polygonal models using virtual tools. *Graphics Interface '95* (1995), 272–278.

[Bli82] BLINN J. F.: A generalization of algebraic surface drawing. *ACM Transactions on Graphics 1*, 3 (1982), 235–256.

[BS91] BLOOMENTHAL J., SHOEMAKE K.: Convolution surfaces. *Computer Graphics 25* (1991), 251–256.

[CG93] CANI-GASCUEL M.-P.: An implicit formulation for precise contact modeling between flexible solids. In *Proceedings of SIGGRAPH 93* (1993), vol. 27, pp. 313–320.

[GH91] GALYEAN T. A., HUGHES J. F.: Sculpting: An interactive volumetric modeling technique. *Computer Graphics 25*, 4 (1991), 267–274.

[GM05] GAIN J., MARAIS P.: Warp sculpting. *IEEE Transactions on Visualization and Computer Graphics 11*, 2 (2005), 217–227.

[Hec97] HECKBERT P.: *Fast Surface Particle Repulsion*. Tech. rep., CMU Computer Science Tech, 1997 1997.

[LWP*04] LI Q., WILLS D., PHILLIPS R., VIANT W. J., GRIFFITHS J. G., WARD J.: Implicit fitting using radial basis functions with ellipsoid constraint. *Computer Graphic Forum 23*, 1 (2004), 55–69.

[MTY00] MATSUMIYA M., TAKEMURA H., YOKOYA N.: An immersive modeling system for 3d free-form design using implicit surfaces. In *Proceedings of Virtual Reality Software and Technology 2000* (Seoul, Korea, 2000), pp. 67–74.

[NM01] NOBLE R., MCDERMOTT R.: Virtual sculpting using implicit surfaces. In *Proceedings of International Conference on Visualization, Imaging and Image Processing* (Spain, 2001), pp. 107–112.

[NZM04] NOBLE R., ZHANG K., MCDERMOTT R.: An investigation of multiple tools actions for virtual sculpting using implicit surfaces. In *Proceedings of EuroGraphicsUK, Theory and Practice of Computer Graphics 04* (Bournemouth, 2004), IEEE, pp. 24–31.

[TO02] TURK G., O'BRIEN J. F.: Modelling with implicit surfaces that interpolate. *ACM transactions on Graphics 21*, 4 (2002), 866–873.

[WH94] WITKIN A. P., HECKBERT P. S.: Using particles to sample and control implicit surfaces. *Computer Graphics 28*, Annual Conference Series (1994), 269–277.

[WLM00] WONG J. P., LAU R. W., MA L.: Virtual 3d sculpting. *The Journal of Visualization and Computer Animation 11*, 3 (2000), 155–166.