Finger Sculpting with Digital Clay

Technical Report GIT-GVU-02-22
30 October 2002

Joshua Gargus
schwa@cc.gatech.edu

Byungmoon Kim
bmkim@cc.gatech.edu

Ignacio Llamas
llamasi@cc.gatech.edu

Jarek Rossignac
jarek@cc.gatech.edu

Chris Shaw
cdshaw@cc.gatech.edu

GVU Center and College of Computing
Georgia Institute of Technology

October 30, 2002

Abstract

Digital Clay is a term that signifies a computer-controlled physical surface, capable of taking any of a wide variety of possible shapes in response to changes in a digital 3D model or changes in the pressure exerted upon it by bare hands. The physical properties of such a device impose design and user-interface constraints not encountered in traditional, tracker-based software for the manipulation of virtual models. This paper describes the interaction techniques we have developed to work with this future medium. In particular, we present our solution for tracking the user's fingers using a local deformation of the surface, which we call a *blister*, that senses the tangential and normal displacements of the finger. We also present a solution for creating variable-height bosses and creases with the simple sweep of a finger. Since the Digital Clay hardware is not yet operational, we have have implemented a haptic simulation framework based on a PHANTOM device.

Figure 1: Sculpting with real clay

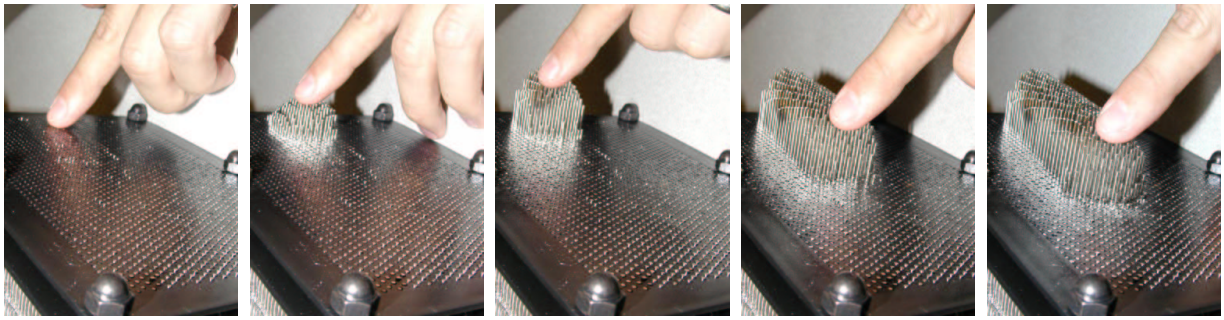


Figure 2: Mock-up using "Pinhead" sculpture toy . The clay raises a *blister* (see Section 4.3) to follow the user's finger as a ridge is raised from the surface (see Section 5).

1 Introduction

Shape is a key element in successful communication, interpretation, and understanding of complex data in virtually every area of engineering, art, science, and medicine. While in recent years the communication of both form and complex data have been greatly enhanced by visualizations based on planar images, computational power and manufacturing technologies have reached the point where it is possible to consider interactive three-dimensional *tactile* input and output devices. The NSF/ITR *Digital Clay* project [1] aims at developing a medium that allows this sort of interaction; it is an *instrumented, actuated, computer-controlled physical volume* bounded by an *actuatable surface* that acts as a haptic interface. We freely abbreviate "Digital Clay" to "clay" throughout the paper.

Digital Clay is conceived as a collaborative tool: as one manipulates the surface, the shape change can be sensed and transmitted to another piece of clay or other computational device. For example, a geographically dispersed team of engineers could interactively explore various shape modifications to a part they are designing.

The goal of the research presented here is to develop user interface techniques that are a good fit for the affordances and constraints of Digital Clay; the challenges inherent in the fabrication of such a device are beyond the scope of this work, and will be discussed elsewhere. Since the Digital Clay hardware is not yet operational, we have implemented a haptic simulation framework based on a PHANTOM device. We are careful to ensure that our clay simulation does not have access to more information about the user than the final hardware will. For example, the clay hardware will only be able to sense the position of the user's finger by the deformations it causes in the clay's surface. We use the PHANTOM only to simulate the effects of the finger on the clay surface; the information available to the clay model is the same, regardless of whether the finger is real or virtual.

Physical interaction between user and clay consists primarily of the forces applied by each to the other. In addition, the user can inspect the shape visually, and by touching it lightly without modifying it. We decided against multimodal strategies such as combined touch and voice input, instead opting to limit ourselves to *"finger sculpting"*: interactions that consist of touching the surface with one or more fingers. The focus of this research is to learn how to interact with a medium whose affordances and limitations are not yet understood. At this early stage, experimenting with multimodal interactions would more likely hinder than facilitate insight into the essential nature of interactions with Digital Clay. The temporary restriction to single-finger interactions rather than whole-hand manipulations results from both the desire to simplify the interaction space that we are exploring, as well as the nature of the PHANTOM device.

It should be emphasized that the clay metaphor

is imperfect. Digital Clay offers manipulation possibilities beyond those of common modelling clay. For example, Digital Clay may change its volume, provide for multiresolution shape editing [7], or allow cut-and-pasting and procedural generation of surface features [19]. By taking advantage of its ability to change shape, Digital Clay can facilitate completion of a task by actively responding to the user. We believe that the most interesting and useful techniques for interacting with Digital Clay will be of this type.

The research contributions of this paper can be categorized as either interaction elements, or complete interactions composed of these elements. The first category includes our solution for tracking the user's fingers using a local deformation of the surface, which we call a *blister*, that senses the tangential and normal displacements of the finger. Other contributions in this category include detecting the user's intent to begin or end an edit operation. The second category contains our methods for accomplishing specific user tasks, such as creating variable-height bosses on the clay surface. In his thesis [17], Pierce argues that by taking a principled approach to exploring the possibilities of a new medium, we can more quickly build up a lexicon of interactions for that medium. We believe that our techniques represent a first step towards such a lexicon for Digital Clay interactions, analogous to the rich toolkit available for creating GUI applications.

1.1 Kinematic Properties of Anticipated Initial Prototype of Digital Clay

Many of the design problems encountered during this research could not have been resolved without considering the physical properties of Digital Clay. Furthermore, several designs for clay prototypes have been proposed; our research has focused on the one dubbed the *"Bed of Nails"*, since it is most likely to be the first one fabricated. We describe aspects of this design that have relevance to the research presented in this paper.

The Bed of Nails design consists of a rectangular matrix of individually actuated rods. It may be helpful to imagine a Pinhead toy (shown in Figure 2) modified so that each pin is computer-controlled. We model the clay as a heightfield $S(x, y)$, and use the term *grid point* to refer to (x, y) positions in a heightfield's domain.

The motive power for each rod comes from a shared hydraulic reservoir, and is controlled by a hydraulic actuator. Using hydraulics allows higher actuator density (and therefore, surface resolution) and generation of stronger forces than achievable by miniaturized electrical motors. These actuators are able to generate forces approximating a damped spring, described by the equation $F(d) = -kd + rd'$. The coefficients k and r represent the spring and damping characteristics of the surface, and d is the displacement of the actual, measured surface configuration from the internal model of where the surface should be. Furthermore, each rod has an individually specified equilibrium height e from which the value of d is computed. Manipulating the values of k, r, and e allows us to control the shape of the clay and the forces exerted by the clay on the finger. Constraints on the acceptable values for k, r, and e depend on other design decisions which are not discussed here.

1.2 Organization of Remainder of Paper

Section 2 describe the work of other researchers that is similar to, or has provided inspiration for our research. Section 3 describes *layering*, which is useful both for describing the desired behavior of the clay, as well as for implementing this behavior. The next two sections contain our primary research contributions. Section 4 explores how our system handles interaction elements such as deciding whether the user is touching the surface to feel or to modify its shape, and tracking a finger as it pushes into, slides along, or pulls away from the surface. These basic building blocks can be combined to create entire operations for finger sculpting. Section 5 describes several such operations, as well as the task scenarios they are designed to address. We also present solutions to issues unique to each task. Section 6 describes aspects of the architecture of our simulation, in order to facilitate replication of our research. Finally, Section 7 sum-

marizes our contributions, and suggests directions for further research.

2 Previous Work

Many haptic systems have been created to manipulate virtual surfaces and volumes. Physical accuracy is important for some applications, such as tissue simulation for web-based surgical training [5]. Physically-based volumes are also used for sculpting because of their intuitive behavior, but tend to be computationally expensive. Recent research seeks to enable interactive response rates by using multiscale techniques [4, 7, 16], or by developing material models that lend themselves to precomputation [14].

Other sculpting systems (including ours) choose not to use physically-based deformations either because of the computational expense, or because certain characteristics (for example, volume preservation) are undesirable. Free-form deformation [3, 18] is an early and very successful approach of this type. It has since been extended to allow the use of different deformation "tools" [6], and to allow direct manipulation of the surface instead of by adjusting a control lattice [11]. Haptic feedback is added in [10]; as with our system, forces are generated using a spring model.

Several researchers have noted that human touch perception is actually comprised of a number of distinct mechanoreceptive systems [15, 2]. Haptic devices such as the PHANTOM only target the *kinesthetic channel*, which gathers information from sensory receptors in the muscles and tendons. The *cutaneous channel*, which senses pressure information with mechanoreceptors embedded in the skin, is ignored by most haptic devices.

Such observations have motivated researchers to develop hardware that can take advantage of the cutaneous channel. A device that laterally stretches the finger's skin is described in [9]. Shape memory alloy is used as a tactile output device by [20]. Digital Clay can also be included in this category, since it provides an actual surface that can be felt with bare hands.

FEELEX [13] describes a hardware device that bears a striking resemblance to the Bed of Nails design, although the latter will have a significantly higher resolution. However, few details are given about the implementation of specific interactions with their system.

3 Layered Clay Representation

In this section, we introduce *layering*, which is pivotal to both conceptualizing and implementing the clay's behavior. We discuss here only the layers of conceptual significance; layers existing only for implementational reasons are discussed in Section 6. An understanding of layering is assumed by the discussions in Sections 4 and 5.

The mathematical representation of the volume is decomposed into n layers $L_{i=0..n-1}(x, y)$, each of which is a heightfield. The height $S(x, y)$ of the clay surface is the sum of the heights of the layers:

$$S(x, y) = \sum_{i=0}^{n-1} L_i(x, y)$$

Another way to look at this is that each layer represents a delta function to be applied to the layer beneath it. These delta values may be either positive or negative.

We also define intermediate heights S_j as the sum

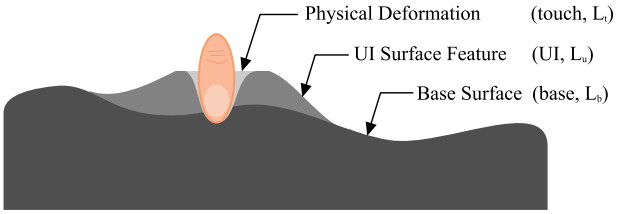


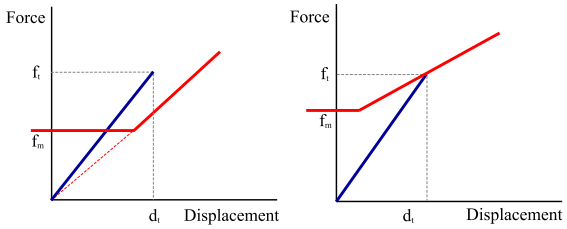
Figure 3: Conceptual and Architectural Layering. Physical deformations are specified as offsets from UI surface features. Similarly, the UI surface features are specified as offsets from the base surface. The names in parentheses are those by which the layers are denoted in the rest of the paper.

of the heights of L_i and all of the layers beneath it:

$$S_j(x, y) = \sum_{i=0}^{j} L_i(x, y)$$

Figure 3 shows the three main conceptual layers in our system, which play a large role in the following sections. The *base layer* L_b reflects the shape of the surface as it was just before the last time it was touched. The *UI layer* L_u is used to locally and temporarily change the shape of the surface to facilitate user interactions with the clay. The *touch layer* L_t represents deformations in the surface caused by user pressure; in physical clay, values for this layer would be computed by measuring the differences between the sensed physical state of the surface and the model of what the surface would be if no external force were being applied (given by L_u in this example).

Extra functionality can easily be added by adding new layers. Section 6 describes one example of how layers have aided in the implementation of our system.



(a) discontinuous transition

(b) continuous transition

Figure 4: Two force profile transitions for switching from *touch mode* to *edit mode*(damping effects are omitted for simplicity). Blue (resp. red) curves describe the force profile before (resp. after) the transition. d_t is the threshold displacement at which the transition occurs, and f_t is the corresponding force. Note that in both diagrams, the edit mode profile has a minimum force of f_m, even for negative displacements; this is discussed in Section 4.3.1.

4 User Interaction

Section 1.1 describes the physical characteristics of Digital Clay. When the clay is not being edited, the spring and damping coefficients are fixed, and it feels like an elastic surface that always returns to the same equilibrium shape. Once an edit operation begins, it actively changes shape in response to the user's input as described in the following subsections.

4.1 How does the clay know where it is being touched?

Digital Clay has no direct way of obtaining the position of the user's finger, and must compute this information based on deviations between the measured and expected surface configurations. Although our simulation has access to the location of the PHANTOM, we refrain from using this information directly in order to remain true to the constraints inherent in Digital Clay. Instead, we deform the touch layer to simulate the reaction to finger pressure.

Since we are concerned only with single-finger interactions, we are able to use a simple algorithm to determine where the user is touching the clay. When the user presses on the surface, the heights of several rods will be displaced from the heights dictated by the internal model of the surface's shape. The position of the user's touch is computed as the centroid of the affected rods. The centroid is calculated by summing the weighted (x, y) coordinates of each rod, where the weight for each rod is proportional to the magnitude of its displacement.

4.2 How does the user indicate an intent to edit?

When touching the surface, the user will in some cases intend to modify it and sometimes only wish to feel its shape. The clay maintains a finite state machine that determines how to react to each of these cases. For example, in *touch mode*, the clay behaves elastically, resisting with a spring force without changing its desired equilibrium shape. In *edit mode*,

the clay's surface is modified in response to the user's touch. There are many types of modifications that can occur; a few of them are described in Section 5.

How does the user indicate when to make the transition from touching to editing, or vice-versa? We discuss the first case here, and the second in Section 4.4. The transition between touching and editing mode occurs when the user presses deeper into the surface than a threshold displacement d_t. The user is notified to the transition by a change in the clay's force profile. The surface becomes softer, and the user feels as though she has broken through some resistance into editing mode. In Figure 4, we depict the two force profile transitions that we experimented with. We name them the *discontinuous* and *continuous transitions*, according to whether there is a change in applied force at the instant of the transition. During the discontinuous transition, only the slope of the force-displacement curve is changed. As a result of the force discontinuity, the user often pushes unexpectedly far into the surface. This prompted us to develop the continuous transition, where the slope and offset of the force profile are simultaneously adjusted so that the curves of the editing and touching profiles cross at d_t. Users of the system were still distinctly aware of the transition, but did not suddenly push deeper than intended, as they tended to with the discontinuous transition.

4.3 How can the user specify input "above the surface"?

In Section 4.1, our discussion of how to determine the position of the user's finger assumes that the user wants to specify input points beneath the surface of the clay. However, it is easy to devise a scenario where the user wishes to specify an input point above the surface. In one such scenario (explored in greater depth in Section 5) the user desires to raise a ridge whose height follows the trajectory of her finger as it moves above the surface. Since the clay can only locate the user's finger when they are in contact, it must actively seek to maintain contact as the user moves her finger toward a point above the base surface L_b.

We call our solution a *blister*: a temporary bump, raised artificially from the surface to follow the user's finger. The blister is created by modifying the offsets in the UI layer L_u, and is cylindrical in shape.

The following subsections discuss the vertical and horizontal behavior of the blister separately before describing how they are combined in the full behavior.

4.3.1 Vertical blister behavior only

We begin our discussion by keeping the promise made in Figure 4 to explain why the edit force profiles, depicted there in red, extend to negative displacements. The magnitude of a negative displacement is the height above the base layer L_b. However, this does not imply that the finger is not in contact with the clay, since the blister is defined in a layer above L_b, namely the UI layer L_u. The blister adjusts its height to maintain contact with the user in such a way that the force applied to the finger matches the force required by the profile.

This section describes how the height of the blister varies in the simplified case where the finger may only move vertically. Our goal is to derive an expression for the blister height that is general enough to accommodate the edit force profiles for both the continuous and discontinuous cases. We proceed by first finding an expression that generates heights matching a very simple force profile. This expression is twice generalized to encompass more complicated profiles, resulting in an expression capable of describing the cases of interest. Figure 5 depicts these force profiles, each of which will be discussed in the following paragraphs. The figure also shows the blister heights corresponding to each profile.

We start with the simplest force profile, which has the equation $f = kd$ (drawn in red). Since the clay cannot stick to the finger, it cannot apply a negative force; we therefore restrict the domain of this profile to $d \geq 0$. Since the curve passes through the origin, the force profile can be satisfied by adjusting only the spring constant k. No blister is necessary.

Next, we modify the force profile to $f = kd + f_b$ (drawn in green). This is nearly the same as the continuous profile, except that we extrapolate along the same slope (dashed line) instead of clamping to

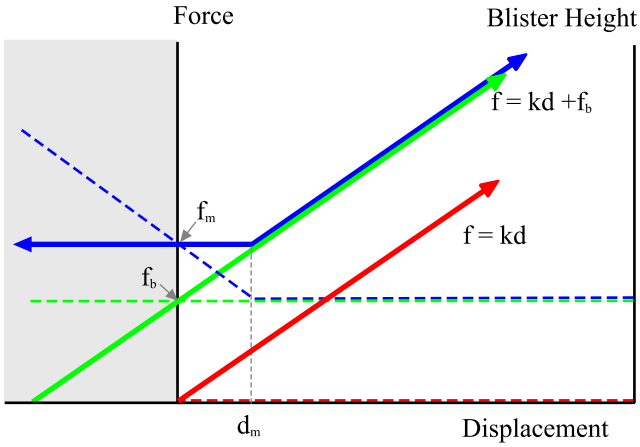


Figure 5: Three force profiles (solid lines), and the corresponding blister heights (dashed lines). f_b raises the blue and green force profiles by a constant amount, and f_m is the minimum force allowed by the blue profile; Section 4.3.1 uses these values to derive the blister height for a given displacement. The shaded area represents negative displacements: positions above the original surface height. The scales of the "Force" and "Blister Height" axes are such that the height of a blister is equivalent to the extra force applied for a given displacement from the base surface.

a minimum force. Since the surface can only apply spring forces, we must add a blister to the surface to apply the extra force $f_b = f_t - kd_t$. Dividing by the slope k gives us the height of the blister:

$$h_1 = f_t/k - d_t$$

Finally, we consider the continuous profile (drawn in blue), which is the same as the last, except that the minimum force is clamped to f_m. The displacement d_m, where the minimum force is first seen, is found by dividing the force by the slope. To see the effect that f_m has on the blister height, consider some $d < d_m$, and look at the difference between the minimum force and the extrapolated (dashed line) force:

$$\Delta f = f_m - (kd + f_b)$$

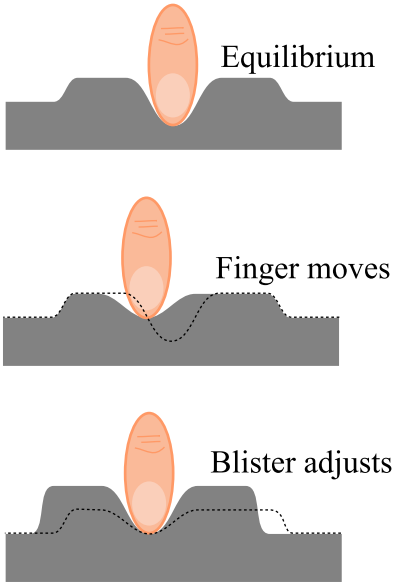


Figure 6: Tracking of Finger by Blister. The user's finger starts in the center of the blister (top), and is moved to the left and up (middle). The clay detects that the blister is no longer centered on the finger, and adjusts the position and height of the blister accordingly(bottom).

This extra force must be generated by raising the blister by the appropriate amount; dividing by the slope k and rearranging terms gives $\Delta h = \frac{f_m - f_b}{k} - d$. The diagram shows that the blister is not affected for $d > d_m$, so we can write the effect of the minimum force constant as

$$h_2 = \max(\frac{f_m - f_b}{k} - d, 0)$$

The total height of the blister is then given by

$$h = h_1 + h_2$$

It is evident from the figure that the discontinuous case is the same as the continuous case with the added constraint $f_b = 0$. Therefore, the expression for h is capable of describing both cases.

4.3.2 Combined horizontal and vertical blister behavior

In order to track the finger horizontally, the blister compares its center with the position of the finger as described in Section 4.1. If there is a discrepancy, then the blister is moved so that it is centered on the finger position.

After the blister is centered on the finger position, the height of the blister may need to be adjusted so that the force applied by the blister to the finger matches the chosen force profile. Note that even completely horizontal finger movement can result in a change of the height of the blister, since horizontal movement will change the displacement of the finger with respect to a non-horizontal clay surface. If the blister height needs to be adjusted, the new height is computed as though the finger motion is purely vertical. Figure 6 shows how the blister tracks the finger when it is moved diagonally.

4.4 How does the user end an edit operation?

If the clay is continually maintaining contact with the user with a raised blister, how does the user notify the clay that the edit operation should end? The answer is to set a maximum rate of change for the blister height. If the user moves her finger up rapidly enough, the blister is unable to follow. As soon as the clay no longer senses the user's touch (see Section 4.1), the edit operation is ended.

Since the user cannot instantaneously break contact with the clay, the last part of the input trajectory will consist of positions that the user does not intend to be part of the edit operation. Our solution is to discard the last part of the input. We have found that discarding 0.2 seconds works quite well; the user need only pause for an instant before raising her finger in order to ensure that no desired input is discarded.

5 Applications

We now suggest several applications of the techniques described in the previous section. The first two have been implemented in our system, and the last two show the broad applicability of our techniques.

Before proceeding, we introduce notation for assigning values in layers. We write $L_i(x, y) \leftarrow h$ to associate a new value with a grid point in a particular layer. We also define assignment for intermediate heights:

$$S_i(x, y) \leftarrow h \quad \equiv \quad L_i(x, y) \leftarrow h - S_{i-1}(x, y)$$

We refer to these as the *default assignment behaviors* for layers.

5.1 Raising Bumps and Digging Holes

The user wishes to dig a hole or raise a bump with a circular cross-section of radius r. After entering the editing mode by pressing into the surface, the position of her fingertip is tracked as described in the previous section. Once a satisfactory position $p_0 = (x_0, y_0, z_0)$ has been chosen, the edit operation is completed by quickly raising the finger from the surface.

The new shape of the base surface is determined as follows. Let $d = z_0 - L_b(x_0, y_0)$ be the height difference between the finger and its projection on the base layer. Also, let p be an arbitrary grid point no farther than r from (x_0, y_0). The base layer heights are modified by a Gaussian function whose steepness is determined by σ, so that $S_b(p) \leftarrow S_b(p) + e^{-\sigma \|p_0 - p\|^2}$. We call this offset function a *Gaussian bump*. Note that raising and digging are accomplished identically; if d is positive, a bump is raised, otherwise a hole is created.

5.2 Embossing

Instead of creating a single bump or hole, the goal is to raise a ridge or dig a ditch. Once can switch between digging a ditch and raising a ridge simply by changing whether the input position is below or above the surface. As with bumps and holes, ridges

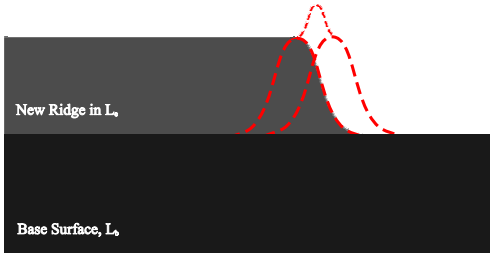


Figure 7: Result of immediately incorporating offsets into base surface. The user has specified a level trajectory for the apex of the ridge, but the resulting surface is above this trajectory. Note that the severity of the undesired effect is exaggerated by the large distance between the two adjacent bumps.

and ditches are created similarly; this observation allows us to simplify our presentation by allowing a mention of one to refer to both whenever the intended meaning is clear.

Our first attempt at an embossing operation modified the base surface only after the entire trajectory was known. This proved unsatisfactory, as users complained loudly about the lack of intermediate feedback. We addressed these concerns by incrementally raising the ridge as the trajectory is specified. The following discussion is concerned only with the revised behavior.

While defining a trajectory of sampled points, the user will occasionally move quickly enough that there is a significant gap between points. To avoid the jaggy ridge that would result, we add additional points along the line segment between widely spaced points, until there are no points that are more than a single grid unit apart. This works fairly well because when the user moves quickly, the motion is usually quite linear. Had this assumption proved to be incorrect, it would be straightforward to sample new points from a spline fit to the sensed positions.

As points are added to the trajectory, we compute their effect on the ridge by applying a Gaussian bump based on the displacement of the point from L_b. However, we do not take the naive approach of merging the effect of each point into the base surface before the next point is handled. Figure 7 shows that doing so results in a ridge that is higher than the trajectory provided by the user. Our solution involves adding a new layer, the *embossing layer* L_e, which stores surface modifications that have not yet been incorporated into the base surface. The embossing layer fits between L_b and L_u (i.e. $0 = b < e < u < t$), and differs from the previously introduced layers in two ways. The first is that it overrides the default assignment operator, and the second is that it merges surface modifications into the base layer once they exceed some threshold age.

5.2.1 Overriding the Default Assignment Operator

We modify the default assignment behavior (defined at the beginning of this section) so that the new value h is only assigned if its magnitude is greater than that of the existing value. This is formalized in the following equation, where $\leftarrow_o$ and $\leftarrow_d$ respectively denote the overridden and default assignment operators:

$$L_e(x,y) \leftarrow_o h \quad \equiv \begin{cases} |h| > |L_e(x,y)| \Rightarrow L_e(x,y) \leftarrow_d h \\ |h| \leq |L_e(x,y)| \Rightarrow \text{unchanged} \end{cases} \tag{1}$$

Regardless of whether the assignment makes a change, the age associated with $L_e(x,y)$ is reset to zero.

5.2.2 Aging of Surface Modifications

Before discussing the details of how surface modifications are aged, we first justify why we do it at all. Figure 8 depicts the desired behavior when a ridge, after being reduced in height, intersects itself. As the lower part of the ridge encounters the higher part, a furrow is cut through the higher part. In other words, we want the older part of the ridge to be treated as if it were part of the base surface, even though we have discussed why this is unsatisfactory for the leading edge of the ridge. Our solution has two components. First, we increment the age of all edited grid points, and merge the offsets into the base layer when the age exceeds a threshold t_a. Next, we reset the age of any grid point to which we attempt to assign a new

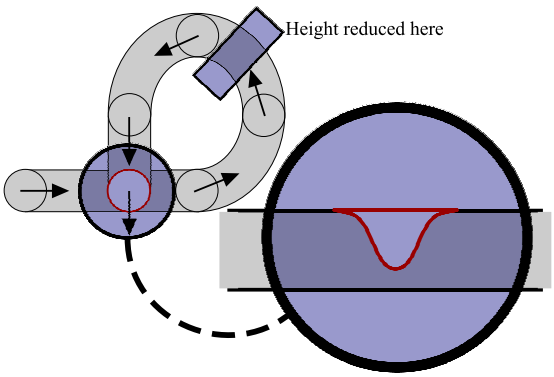


Figure 8: Desired behavior of self-intersecting ridge when the height of the ridge is lower at the time of intersection than at the start of the manipulation. On the left is a top-down view of such a ridge. On the right is a cross-sectional view of the intersection. The height of the ridge is reduced in the area bounded by the blue rectangle.

value h, even if the assignment does not affect the heightfield because magnitude of h is less than the existing value of $L_e(x, y)$.

5.2.3 Fixing Mistakes

The user will sometimes raise a ridge higher than desired, and be unable to lower it because the desired height has a smaller magnitude than the existing height. We solve this problem by not resetting grid point ages when the input position is stationary. We allow a small amount of leeway, since the user will not be perfectly steady. Let p_s represent the *stationary point*: the point that other points are compared to to determine whether the user is stationary. p_s is initially set to the first input point. We use a threshold t_s to define the region in which the finger is considered to be stationary. If $\|p_s - p\| < t_s$, then no ages are reset with the edit region. Otherwise, the ages are reset, and the value of p_s is set to p. If the input point remains stationary for longer than t_a, then this aging behavior will have the result of incorporating the entire ridge into the base layer. p_s is now beneath the base surface.

Does this give the desired behavior? Consider a bump raised to a height h_1 that is higher than the desired height h_2. For simplicity, the height of the base surface is initially zero. The desired height of the bump at a point p in the neighborhood of $p_2 = (x_0, y_0, h_2)$ is given by $h_2(e^{-\sigma\|p_0-p\|^2})$. If we first raise the bump too high to h_1 and then correct it as described, the result will be

$$h_1(e^{-\sigma\|p_0-p\|^2}) + (h_2 - h_1)(e^{-\sigma\|p_0-p\|^2})$$

which simplifies to the desired result:

$$h_2(e^{-\sigma\|p_0-p\|^2})$$

.

After experimenting with a variety of values for t_a, we found 0.5 seconds to be satisfactory. This provides the stated benefit of aging without making the user wait annoyingly long to fix a mistake.

5.3 Finger Painting

The 3D haptic painting functionality of [8] allows the user to paint directly on a virtual clay-like surface. Digital Clay could act as an input mechanism for *finger painting*, where the color of the clay is changed by drawing on it with the finger. As in the FEELEX system [13], a projector directed at the clay could show the results of the painting. Alternately, advances in LEP (light emitting polymer) and OLED (organic light emitting device) allow us to imagine a future version of Digital Clay that can change the color of its surface; such a device would be perfect for digital finger painting.

5.4 Extrusion of Region

Teddy [12] interprets the user's freeform 2D strokes to define plausible 3D models. Of particular interest to us are the multi-step operations such as extrusion. The user first draws on the surface the outline of the base of the region that is to be extruded. A second stroke defines how region is raised from the surface.

Multi-stroke operations can also be implemented for Digital Clay. We again use extrusion as an example. The area surrounded by a closed ridge could

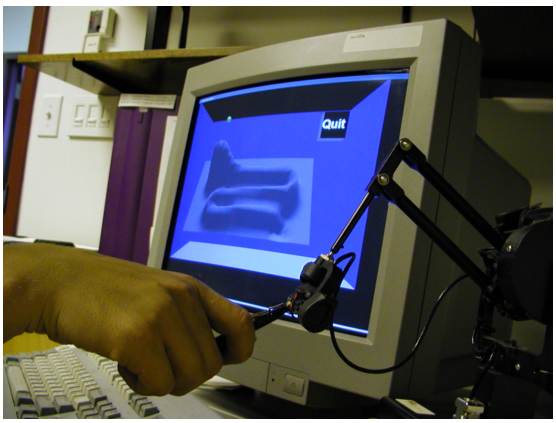


Figure 9: The simulation in use by a user, who has just finished raising a serpentine boss from the surface.

be interpreted as the base for an extrusion operation. The shape of the extruded volume could depend on the point at which the extrusion is initiated, as well as the subsequent trajectory of the finger. Such multi-stroke operations will allow greater control over the shape of Digital Clay.

6 Implementation

6.1 Control Flow: Haptic and Control Loops

GHOST, the standard framework for developing PHANTOM applications, structures applications by splitting execution into two loops. The *haptic loop* is in charge of generating feedback forces. The *control loop* is responsible for updating the graphic display, and for computing the clay's response to user input.

In order to maintain the illusion that a surface exists, the haptic loop must have an update rate of at least 1000Hz. This strict performance requirement means that the computations done by this loop must be kept simple. For example, operations that modify entire regions of the surface run too slowly for the haptic loop. Such computationally intensive operations are instead processed in the control loop, which has a frequency of approximately 20Hz. Unfortunately, changing the surface's height at this relatively low rate results in perceptible jerkiness. The next section describes a new layer that addresses this problem.

6.2 Layers Revisited

The layers described thus far have both a conceptual and an implementational role. Each plays a part in describing how the clay behaves, and each is included in our implementation in a straightforward manner.

Other layers do not have a place in the conceptual understanding of Digital Clay, and exist solely for implementation reasons. For example, we have implemented a type of layer that linearly interpolates between two reference surface shapes. This layer addresses the problem of jerky height transitions by storing the previous and current surface heights, and blending them. The interpolation parameter ranges from zero at the instant that the current height is set, to one just before the next height is set. The perception is now that the height is changing smoothly.

7 Conclusion

This paper has introduced the concept of *Digital Clay*, and described the salient characteristics of the planned physical prototype. We have described our PHANTOM-based haptic simulation of a finger interacting with Digital Clay, emphasizing that we have refrained from using any information that the physical prototype will not have access to; in doing so, we are confident that the techniques we have developed can be smoothly transferred to the physical prototype.

We have coined the term *finger sculpting* to describe interactions that involve manipulation of the surface with one or more fingers. Due to the hardware at our disposal, we have developed techniques that work with a single finger.

Our research represents the first steps towards an interaction lexicon for Digital Clay. We have described interaction elements that allow the clay to sense the position of the user's touch, and to deter-

mine when the user intends to start and end editing. We also presented the *blister*, which allows the clay to maintain contact with the finger, even when the finger is raised above the original level of the surface.

These interaction elements were combined to form two complete clay manipulation operations. We describe how we have implemented digging holes and raising bumps, as well as a variable-height embossing operation.

8 Acknowledgements

This work was supported by the National Science Foundation under grant NSF-ITR/PE+SY Award#:0121663, and by a Seed Grant from the GVU Center at the Georgia Institute of Technology. The authors thank Norberto Ezquerra for providing access to a Phantom.

References

[1] Allen, M., Book, W., Ebertuphoff, I., Rosen, D., and Rossignac, J. Digital Clay for Shape Input and Display: NSF-ITR/PE+SY Award#:0121663, September 2001.

[2] Ambrosi, G., Bicchi, A., Rossi, D. D., and Scilingo, P. The role of contact area spread rate in haptic discrimination of softness. In *Proceedings of IEEE International Conference on Robotics and Automation* (May 1999), IEEE, pp. 305–310.

[3] Barr, A. H. Global and local deformations of solid primitives. In *Proceedings of ACM SIGGRAPH* (July 1984), ACM SIGGRAPH, pp. 21–30.

[4] Capell, S., Green, S., Curless, B., Duchamp, T., and Popovic, Z. A multiresolution framework for dynamic deformations. In *Proceedings of ACM SIGGRAPH* (2002), ACM SIGGRAPH.

[5] Choi, K. S., Sun, H., Heng, P. A., and Cheng, J. C. Y. A scalable force propagation approach for web-based deformable simulation of soft tissues. In *Proceeding of the seventh international conference on 3D Web technology* (2002), ACM Press, pp. 185–193.

[6] Coquillart, S. Extended free-form deformation: A sculpting tool for 3d geometric modeling. In *Proceedings of ACM SIGGRAPH* (August 1990), ACM SIGGRAPH, pp. 187–196.

[7] Debunne, G., Desbrun, M., Cani, M.-P., and Barr, A. H. Dynamic real-time deformations using space time adaptive sampling. In *Proceedings of ACM SIGGRAPH* (2001), ACM SIGGRAPH, pp. 31–36.

[8] Ehmann, S. A., Gregory, A. D., and Lin, M. C. A touch-enabled system for multiresolution modeling and 3D painting. *The Journal of Visualization and Computer Animation 12*, 3 (2001), 145–157.

[9] Hayward, V., and Cruz-Hernandez, J. M. Tactile display device using distributed lateral skin stretch. In *Proceedings of the Haptic Interfaces for Virtual Environment and Teleoperator Systems Symposium (ASME IMECE2000)* (2000), pp. 1309–1314.

[10] Higashi, M., Aoki, N., and Kaneko, T. Application of haptic navigation to modify free-form surfaces through specified points and curves. In *Proceedings of the seventh ACM symposium on Solid modeling and applications* (June 2002).

[11] Hsu, W. M., Hughes, J. F., and Kaufman, H. Direct manipulation of free-form deformations. *Computer Graphics 26*, 2 (1992), 177–184.

[12] Igarishi, T., Matsuoka, S., and Tanaka, H. Teddy: A sketching interface for 3d freeform design. In *Proceedings of ACM SIGGRAPH* (1999), ACM SIGGRAPH, pp. 409–416.

[13] Iwata, H., Yano, H., Nakaizumi, F., and Kawamura, R. Project feelex: Adding haptic surface to graphics. In *Proceedings of ACM*

SIGGRAPH (August 2001), ACM SIGGRAPH, pp. 469–476.

[14] James, D. L., and Pai, D. K. A unified treatment of elastostatic contact simulation for real time haptics. *Haptics-e, The Electronic Journal of Haptics Research 2*, 1 (September 2001).

[15] Johnson, K. O. The roles and functions of cutaneous mechanoreceptors. *Current Opinion in Neurobiology 11* (2001), 455–461.

[16] McDonnell, K. T., and Qin, H. FEM-based subdivision solids for dynamic and haptic interaction. In *Proceedings of 6th Symposium on Solid Modeling and Application*, pp. 312–313.

[17] Pierce, J. S. *Expanding the Interaction Lexicon for 3D Graphics*. PhD thesis, Carnegie Mellon University, School of Computer Science, Pittsburgh, Pennysylvania 15213, 2001.

[18] Sederberg, T. W., and Parry, S. R. Free-form deformation of solid geometric models. In *Proceedings of ACM SIGGRAPH* (August 1986), ACM SIGGRAPH, pp. 151–160.

[19] Velho, L., Perlin, K., Ying, L., and Biermann, H. Procedural shape synthesis on subdivision surfaces. In *SIBGRAPI 2001 - XIV Brazilian Symposium on Computer Graphics and Image Processing* (October 2001).

[20] Wellman, P. S., Peine, W. J., Favalora, G. E., and Howe, R. D. Mechanical design and control of a high-bandwidth shape memory alloy tactile display. In *International Symposium on Experimental Robotics* (June 1997).