Virtual Clay for Direct Hand Manipulation

Guillaume Dewaele and Marie-Paule Cani

Abstract

In order to make virtual modeling as easy as real clay manipulation, we describe a realtime virtual clay model, specially designed for direct hand manipulation. We build on a previous layered model for clay, extending it to handle local properties such as colour or fluidity, to deal with an arbitrary number of tools, and to capture twist effects due to rotating tools. The resulting clay model is the first step towards a more long term goal, namely direct interaction through video tracking of the user's hands.

1. Introduction

Many artists find clay modelling to be a simpler, more intuitive way to design complex shapes than numerical shape modelling. We believe that a virtual clay model could be even more advantageous. While keeping an intuitive interaction, it would enable copy/paste and undo operations, avoid unwanted drying of the clay, enable changing physical properties, and allow the tools to scale as much as needed for easily modelling both large scale and very fine features.

We previously presented a volumetric model for virtual clay [DC03]. This was the first realtime model to enable global and local deformations mimicking the central features of real clay, namely plasticity, volume conservation and surface tension. However, this model suffered from important limitations: the clay had to be homogeneous, only two tools could simultaneously interact with the clay, and only the tools translations were taken into account during interaction.

This paper presents an extension of the model that enables local properties to the clay such as colour and fluidity, allows an arbitrary number of tools, and captures twist effects due to rotation of the tools. These contributions make the clay ready for direct hand manipulation, as will be discussed below.

1.1. Previous work

Volumetric clay models: Both surfaces and volumetric representations have been used for interactive sculpting (see [DC03] for a detailed state of the art). This section only reviews previous volumetric models, which are closer in spirit to real-clay and easily capture topological changes.

Most volumetric models define the object as an iso-surface of a scalar field, either stored in a grid or controlled through primitives such as B-spline volumes [IESQK99, PF01, RE00]. These models allow volumetric operations such as adding material or carving it. Local deformations were made possible [FCG00], but with no control of volume variations. Cellular automata were also used to compute local deformation of virtual clay [ATTY99, DAB03] or sand [ON03]. These models allow free-form modelling with volume conservation and topological changes. However, the user can't perform large-scale deformations such as bending the limbs of a model.

Physically-based simulation is another option to model clay, viewed as a material that lies between plastic solids and viscous fluids. However, plastic-solids require the addition of specific mechanisms for handling topological changes [MQW01]. Non-realtime, particle-based models were used to represent viscous material [Ton91] somewhat similar to clay. Realtime performances were achieved using Eulerian fluid simulations [Sta99], but this approach requires the fluid to occupy the entire space and thus fails to describe clay, which is bounded by a time-varying surface.

A layered model for virtual clay: The model presented in this paper is an extension of that introduced in [DC03]. This was the first model to simulate in realtime the main features of real clay, namely plasticity, volume conservation and surface tension. The clay surface is defined as the iso-surface at 0.5 of a scalar field

modelling the clay density. Field values are stored in a 3D grid and clamped between 0 (an empty cell) and 1 (a cell full of clay). In addition to classical carving or addition of material, the model captures local and global deformations which are expressed through clay displacement from a grid cell to another. To our knowledge, there is no realtime physically-based model for virtual clay . Three layers are used to achieve deformations, applied in turn at every time step:

- The first layer simulates large-scale, plastic deformations. It allows the user to bend parts of the sculpted model using two rigid tools, possibly one of which freezes a region of the clay to keep it still. The translation δ to apply to clay material lying in a given cell is computed as a linear combination:

$$\delta = (1 - f)(k_1\delta_1 + k_2\delta_2) \tag{1}$$

of the two tools translations vectors δ_1 and δ_2. The fluidity f scales the displacement field to take into account that only a part of the motion of the tool is transmitted to clay when its fluidity is high. The weights are defined as:

$$k_1 = \frac{1 - \frac{d_1 - d_2}{d_{12}}}{2} \quad \text{and} \quad k_2 = \frac{1 - \frac{d_2 - d_1}{d_{12}}}{2} \tag{2}$$

where d_1 and d_2 are the respective pseudo-distance from the current cell to the tools, and d_{12} the pseudo-distance between them. The pseudo-distance models the propagation of the quantity of movement inside a semi-fluid material. It can be seen as the length of the path, inside the object, along which the motion is transmitted. The longer this path is or the smaller the clay density is along it, the smaller the generated motion is. The pseudo-distance is computed through a propagation scheme that starts from a tool and propagates in the clay until it reaches its border or cells covered by other tools. For each non-empty cell c_i:

$$d_i = min_{neighbours}(d_j) + \frac{1}{\rho_i} \tag{3}$$

where ρ_i is the density of clay in c_i.

- The second layer prevents volume variation by iteratively poring clay in excess (i.e. clay in cells which density value exceeds 1 or which are occupied by a tool) into neighbouring cells. This results into intuitive folds and local bulges when the user deforms of presses the clay.

- The third layer mimics surface tension by moving clay in cells where the density value is below 0.5 towards the surface of the sculpted model. As a result, clay does not spread into non-visible low-density regions, and the object remains compact anytime.

1.2. Discussion and overview

Although this model mimics clay behaviour in realtime, it suffers from three major limitations: first, the clay parameters, such as fluidity, are uniform across the volume. The clay thus always behaves as a homogeneous material, although in the real world, a designer often moistens specific regions in order to get locally smaller scale deformations. Specifying different parameter values in each cell is not sufficient to solve the problem: the values also have to be transported with the clay and to be taken into account when computing deformations. Secondly, although a designer generally needs all his fingers for deforming real clay, the large-scale deformation layer is only able to handle interaction with two tools, due to the very specific formula (2) used for computing the influence of the tools. This makes the model non-usable in the context of direct-hand manipulation, which is our long term goal. Lastly, only the tools translations were considered when computing large scale deformations. Tool rotation is highly desirable, for instance to locally twist the clay.

The remainder of this paper presents an extension of the model that overcomes these limitations. Section 2 shows how local properties (fluidity, colour...) can be defined and transported when clay deforms. The influence of a locally-defined fluidity on large-scale deformations is addressed. Section 3 generalizes the model to handle the simultaneous interaction of an arbitrary number of tools. Section 4 extends it to take into account tools rotations. Section 5 discusses the use of this model in the context of direct hand manipulation, which is the long-term goal we are trying to reach.

2. Local properties of clay

Our first goal is to be able to define local parameters for the clay and adequately transport them during deformations. This enables us to model non-homogeneous clay exhibiting local variations of fluidity, but may also serve for attaching any other local property to the clay. We illustrate this by introducing a colour parameter. Depending on the needs, one could also think of attaching temperature or surface roughness values to the clay, or of using local parameters to tune the effects of the volume preservation and surface tension layers.

Adding a new local property is made straightforwards by our volumetric representation: we just store the extra parameter values in each non-empty grid cell. Classical 3D painting tools are used to increase or decrease the values in the clay volume. The main concern is how to adequately attach the local properties to the clay when it moves and deforms.

2.1. Updating local properties

A local property should be linked to the clay material in a cell rather than to its specific position in space. We thus have to update cell parameter values each time some clay moves due to the action of one of the layers. Although clay motion is modelled by increasing the density value in a destination cell while decreasing it in a source cell, the values of local properties only have to be updated in the destination cell, since the material's nature in the source cell remains the same.

Let ρ_i represent the amount of clay being transported to the destination cell and ν_i be the vector of associated local properties. Let ρ_j and ν_j respectively be the quantity of clay and its parameters already stored in the destination cell. Then the natural choice for computing the new values of local properties associated to the quantity of clay $\rho_i + \rho_j$ in the destination cell is the weighted average:

$$\nu_{destination} = \frac{\rho_j \nu_j + \rho_i \nu_i}{\rho_j + \rho_i} \qquad (4)$$

For instance, in the case of fluidity, ν represents the proportion of water in the clay. The new proportion is indeed the weighted average of the previous values. Another example is to attach colour parameters to the clay: a vector of three values respectively representing the amount of red, green and blue is then stored in each cell. Deformations result into a quite natural colour blending effects, as depicted in Figure 1.

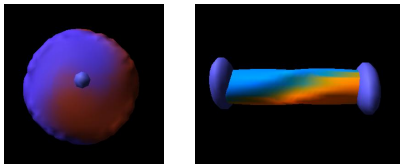

Figure 1: *Twisting coloured pieces of clay.*

2.2. Deforming non-homogeneous clay

In the previous model, fluidity f used in equation citeeq:translation was considered uniform throughout the clay. Clay with locally varying fluidity is modelled by setting an extra parameter in each cell in order to store the proportion of water contained. In areas where the density of clay is low or where fluidity is high, the movement should propagate less from a cell to another. To achieve this, we still compute the pseudo-distance from a tool using the previous scheme, but we replace equation 3 by:

$$d_i = min_{neighbours}(d_j) + \frac{1}{(1 - f_i)\rho_i} \qquad (5)$$

so that the "distance" increases more quickly where density ρ_i is low and fluidity f is high.

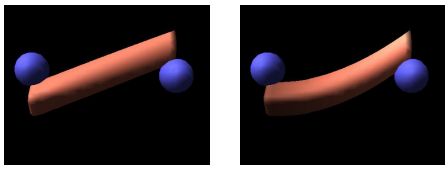

Figure 2: *Bending homogeneous clay (left) vs clay with locally-defined fluidity, depicted using colour indexes (right).*

3. Simultaneously applying multiple tools

Interacting with the clay using an arbitrary number of tools is not a problem for the second and third layers of the model, since they simply move clay if the amount of it in a cell is above of below given thresholds. However, the specific formula used for computing the displacement field in the first layer (see equation 2) was restricting the previous model to the interaction with no more than two tools.

First, we can come up to an intuitive interpretation of the previously used formula: the parameters k_i kind of partition the volume into voronoi regions attached to the tools. But these regions have a "soft border" between the tools where their respective actions are blended.

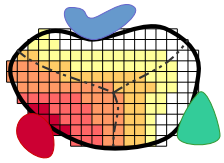

Figure 3: *Influence region for the red tool.*

We aim at defining similar regions of influence in the case of several tools. In order to achieve this, for each tool involved, we compute the weight coefficients k_i and displacement δ as:

$$k_i = \frac{1 - \frac{d_i - min_j(d_j)}{min_j(d_{ij})}}{2} \quad \textbf{and} \quad \delta = \frac{\sum_i k_i \delta_i}{\sum_i k_i} \qquad (6)$$

where j refers to all the other involved tools. As in the case of two tools, this results into Voronoi-like regions of influence, with a continuously varying effect of a tool's motion between them (see figure 3).

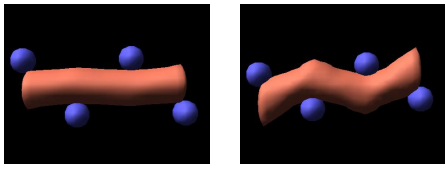

Figure 4: *Simultaneous action of four tools*

4. Rotating tools

Up to this point, we only considered that tools motion would be translations when they are in contact with the clay. However, the user may also rotate tools, expecting to produce rotations or twists in the clay.

The motion of a solid rotating object cannot be described by a simple displacement vector. Instead, a point A rigidly linked to the tool moves according to displacement field:

$$\delta_A = \delta_O + OA \times \omega \tag{7}$$

where δ_O is the translation of point O (the center of the tool) and ω, the screw of the tool. To take into account the rotation of the tool, we simply replace the previous δ_i in equation (6) by the δ_A for cell i in equation (7). This way, more general deformations can be generated. For instance, a bar of clay can be twisted by simply turning a tool at one end of the bar.

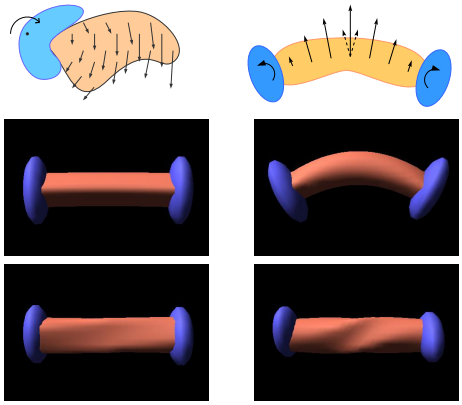

Figure 5: *Up: displacement field induced by rotating tools. Bottom: bending and twisting a bar of clay*

5. Towards hand manipulation of clay

This paper presented a realtime clay model enabling local user-controlled variations of physical parameters, deformation through the simultaneous action of an arbitrary number of tools, and the use of tools rotations during interactions.

This model is a first step towards a very challenging long-term goal: enabling an artist to use his hands for modelling virtual clay as he would with a real material. Of course, an appropriate interface is needed to capture hands and fingers motion. The most obvious solution would be to use an haptic glove, possibly with an exoskeleton for force feedback. We're currently studying a less invasive approach, using a vision-based interface. The motion of the user's hands, filmed by a few video cameras, will be used as an input to control virtual hands serving as multiple tools. However our current solution to hand tracking [DDH04] is not realtime yet. We are also considering the user getting force-feedback by manipulating a real object (e.g. a ball full of sand), the latter serving as an avatar for the virtual clay.

References

[ATTY99] Arata H., Takai Y., Takai N. K., Yamamoto T.: Free-form shape modeling by 3d cellular automata. In *Shape Modeling International* (1999), pp. 242–247.

[DAB03] Druon S., A.Crosnier, Brigandat L.: Efficient cellular automata for 2d / 3d free-form modeling. In *WSCG* (Feb. 2003).

[DC03] Dewaele G., Cani M.-P.: Interactive global and local deformations for virtual clay. In *Pacific Graphics 2003* (2003).

[DDH04] Dewaele G., Devernay F., Horaud R.: Hand motion from 3d point trajectories and a smooth surface model. In *ECCV* (2004).

[FCG00] Ferley E., Cani M.-P., Gascuel J.-D.: Practical volumetric sculpting. *the Visual Computer 16*, 8 (Dec. 2000), 469–480.

[IESQK99] IX F. D., El-Sana J., Qin H., Kaufman A.: Haptic sculpting of dynamic surfaces. *1999 ACM Symposium on Interactive 3D Graphics* (April 1999), 103–110.

[MQW01] McDonnell K. T., Qin H., Wlodarczyk R. A.: Virtual clay: A real-time sculpting system with haptic toolkits. *2001 ACM Symposium on Interactive 3D Graphics* (2001).

[ON03] Onoue K., Nishita T.: Virtual sandbox. In *Pacific Graphics 2003* (2003), pp. 252–259.

[PF01] Perry R. N., Frisken S. F.: Kizamu: A system for sculpting digital characters. *Proceedings of SIGGRAPH 2001* (2001), 47–56.

[RE00] Raviv A., Elber G.: Three-dimensional freeform sculpting via zero sets of scalar trivariate functions. *Computer-Aided Design 32*, 8-9 (2000), 513–526.

[Ros03] Rossignac J.: Finger sculpting with digital clay: 3d shape input and output through a computer-controlled real surface. In *Shape Modeling International* (2003), pp. 229–231.

[Sta99] Stam J.: Stable fluids. In *Proceedings of SIGGRAPH 99* (Aug. 1999), Computer Graphics Proceedings, pp. 121–128.

[Ton91] Tonnesen D.: Modeling liquids and solids using thermal particles. In *Graphics Interface'91* (Calgary, AL, June 1991), pp. 255–262.