Volumetric plastic deformation

Francisco J. R. Prados, Alejandro León Salas and Juan Carlos Torres

Grupo de Investigación en Informática Gráfica
Universidad de Granada, Spain

Abstract

Working with volumetric information is always challenging. Displaying, processing or modifying volumes requires in most cases processing a considerable part of them, if not the whole structure. Such delays might be inconvenient for applications which require fast or smooth interaction, like realistic rendering or virtual sculpting.
This paper focuses on the volume sculpting paradigm. Making use of a haptic device for interaction, the solution presented here provides a smooth and physically correct force feedback. The deformation algorithm is designed to provide a fast visual response. Additionally, this method is able to operate with large volumes, since it was designed to create a progressive, and therefore, scalable simulation.

Categories and Subject Descriptors (according to ACM CCS): I.3.5/6 [Computer Graphics]: Physically based modeling—Interaction techniques

1. Introduction

Deformation has been widely studied in computer graphics. Solids, implicit volumes, FEMs and many other abstractions have been used to create realistic visualizations and fast interactions. However, when representing physical properties of a certain material, volumetric representations seem to be more appropriate [Del98], since they inherently store both external and internal properties.

The main contribution of this paper is precisely a new deformation method which works directly on a 3D image, together with a haptic device for the interaction. Topological changes can be naturally performed due to the volumetric nature of the data. Under a quasistatic assumption, a natural visual output and a physically consistent force are generated. Plasticity is implemented displacing property values. The lazy, progressive deformation approach removes any arbitrary limits on the size of the volumetric representation.

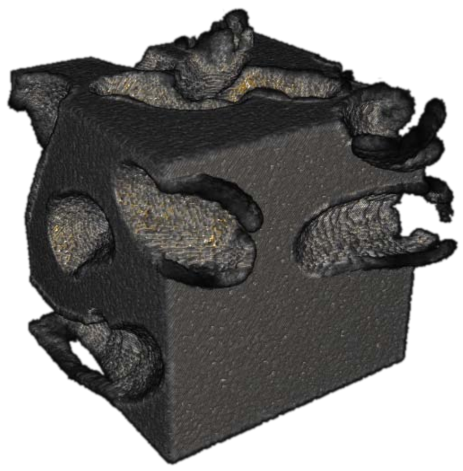

Figure 1: *Volume representing a block of clay which has been plastically deformed.*

2. Related works

There is a vast collection of works which cover, more or less directly, the topic of volumetric deformations. Kim and Park [KP04] developed a sculpting system based on the *proxy* technique. They used implicit volumes to represent the material, and the proxy abstraction to create force feedback. Spring-mass models have also been widely used to calculate deformations on human tissues or elastic materials [Del98], and works such as those of Choi *et al.* [CSH03] and McDonnell [MQW01] have shown satisfactory results when it comes to combining them with haptic interaction. The drawback of the spring-mass methods is that requires heuristic methods to acquire physical reality [NC10]. Dewaele and Cani [DC04] presented a method that made possible the global plastic deformation of a volumetric structure. Displacements fields were computed directly on the grid, using two ore more contact points. The *Chain-Mail Algorithm*, developed by Frisken-Gibson [FG99], makes use of a volumetric structure to compute both elastic deformations and haptic responses. This technique uses chained control points, so when a vertex is translated carries other vertexes in the neighbourhood, therefore simulating plasticity. Many works have also used FEMs to represent deformable solids, such as the one presented by Nesme et al. [NPF05]. Notable results were shown by O'Brien et al. [OBH02], who developed realistic fractures and deformations applying a physically based approach on a FEM. Also noteworthy is the work done by Delingette et al. [DA04]. They presented a solution which makes use of continuum mechanics to perform operations with a virtual liver, built on top of a tetrahedral FEM model.

3. Theoretical background

For the system to produce a physically correct and consistent force feedback, mechanics of deformable solids shall be taken as the basis of the simulation. More precisely, the Incremental Theory of Plasticity [OdSB06] will be for this matter used, for it offers a simple but powerful model which can be easily adapted to our simulation environment.

In the one-dimensional case, this theory defines three differentiated stages in the deformation, which linearly relate the performed deformation ε and the applied stress σ. Briefly, it states that the material accepts a certain amount of elastic deformation, which can be undone if the applied force decreases. The deformation remains recoverable until a critical value of σ is reached. Once crossed, a plastic (and therefore permanent) deformation is produced. This threshold value is known as σ_f, and might increase following a hardening pattern. Anytime the applied force σ decreases, the material returns to an elastic state [OdSB06].

The constitutive law — which establishes a univocal relationship between applied stress σ and caused deformation ε — reads as follows:

- Elastic zone. The solid stays in this zone as long as the stress tensor σ does not exceed σ_f.

$$d\sigma = E\,d\varepsilon \tag{1}$$

- Elasto-plastic load zone. This state applies when $\sigma = \sigma_f$ and there is an increment $d\sigma > 0$ in σ so that $\sigma + d\sigma > \sigma_f$. When hardening is present, σ_f updates to $d\sigma + \sigma_f$.

$$d\sigma = E_{ep}\,d\varepsilon \tag{2}$$

- Elasto-plastic unload zone: Applies when $\sigma = \sigma_f$, and there is an increment $d\sigma < 0$ in σ so that $\sigma + d\sigma < \sigma_f$.

$$d\sigma = E\,d\varepsilon \tag{3}$$

where E and E_{ep} are constants. The generalization of the three-dimensional case is immediate, just using three-dimensional stress and strain tensors and using a matrix of elastic properties for the relation between them in the different zones.

Concerning the transformations occurred in the inner material, the results presented by Prevost [Pre84] shall be used. The work details a simulation of the internal displacements of a deformable solid using a two-dimensional finite-elements method. Those results will be the reference for the later described deformation algorithm.

4. Volumetric deformations

Simulating a *pull* operation performed on the surface of a plastic material requires two differentiated concerns. Firstly, a force signal needs to be created for the haptic device to create a tactile sensation. Secondly, the volume structure requires some processing, for the permanent changes of the plastic deformation to be reflected on the surface.

In this section a brief description of the virtual environment shall be given. The force computation algorithm is afterwards detailed, and finally the deformation technique will be traced.

4.1. Volume domain

The virtual structure which supports the here described technique consists of a 3D image representing the volumetric information. The volume will be defined as a set of homogeneous cubes partitioning the three-dimensional euclidean space, holding each of them a unique property value. Such distribution creates a grid which can be conveniently indexed with three integer numbers i, j, and k. Each cube v_{ijk}, also called a voxel, will be the smallest element used for geometry and force computation. The set of all voxels will be noted as V. The haptic input device will be translated into an implicit function $t(\vec{x}) : \mathbb{R}^3 \rightarrow V$ which returns, for any point of the space $\vec{x}$, a discrete set of voxels which are included in the tool volume.

4.2. Force feedback

The haptic device used for the implementation (a PHANTOM Desktop) does not provide any estimation of the user applied force. However, as every input device, offers a chain of position samples. The Incremental Theory of Plasticity unequivocally relates a certain displacement $d\vec{\varepsilon}$ with the force $d\vec{\sigma}$ required to perform it. The constitutive law for a quasistatic environment reads:

$$d\vec{\sigma} = \mathbb{C}\,d\vec{\varepsilon} \tag{4}$$

$$d\vec{\sigma} = \mathbb{C}_{ep}\,d\vec{\varepsilon} \tag{5}$$

where Eq. 4 refers to the *elastic* and *elastoplastic unload* states and Eq. 5 governs the *elastoplastic load* state, being

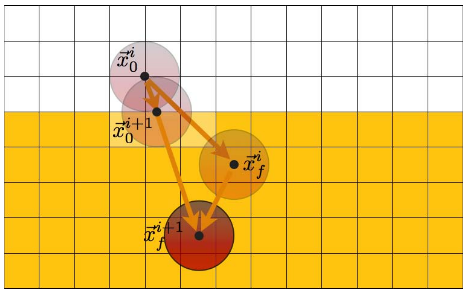

Figure 2: *Computation of the overall displacements during the elastoplastic load. $\vec{x}_0^i$ was the last point where the tool did not intersect the material. $\vec{\sigma}$ increases accordingly with the overall strain until the haptic device reaches $\vec{x}_f^i$, point where the yield condition is met. A further movement causes $\vec{x}_0^i$ to be updated to the new surface at $\vec{x}_0^{i+1}$ (modifications in the surface are marked in pale yellow).*

$\mathbb{C}$ and $\mathbb{C}_{ep}$ the correspondent stiffness tensors. Considering these equations, once the virtual tool has produced a displacement on the volume surface, the force which *should have produced it* can be computed. This calculation produces the normal *cause and effect* flow to be inverted, but that turns out to be a minor flaw due to the fact that the haptic sampling is performed at 1kHz.

Collisions are detected checking for each sampled position $\vec{x}$ that all voxels in the set $t(\vec{x})$ have zero property value. Once a collision has been detected, the differential displacements $d\vec{\varepsilon}$ generated by each unit area of the tool surface will be computed. Differential displacements must be then composed to get the overall stress $\vec{\sigma}$.

For the sake of simplicity, it shall be considered that the material will remain in the *elastic load* state until $|\vec{\sigma}| < \sigma_f$, being σ_f a scalar value inherent to the simulated material. This criterion is formally known as the *yield condition*. When the tool reaches the position $\vec{x}_f$ within the material so that the yield condition is met ($|\vec{\sigma}| = \sigma_f$), the material enters the elasto-plastic load zone. Provided that $\vec{x}_0$ stores the last collision-free position of the tool, any registered displacement that makes $|\vec{\sigma}| > \sigma_f$ will trigger the following changes:

1. $\vec{x}_f$ updates to the latest $\vec{x}$.
2. $\vec{x}_0$ needs to be recomputed so it lays on the *new* surface.
3. The surface of the volume must be modified, so it captures the permanent deformation.

Force is computed during the elasto-plastic load using Eq. 5. As it will be later explained, changes in the surface are performed by the deformation algorithm. The updating process can be seen in Fig. 2. Should a displacement cause the stress tensor $\vec{\sigma}$ to decrease at any time of this process, the system would inmediatly move into the *elasto-plastic unload* zone. No further permanent effect in the volume would be then registered.

4.3. Deformation localization

Ideally, once the force applied to the surface is known, the elastic-plastic equations could be applied to the whole volume to solve the deformation, as seen in Prevost's study [Pre84]. However, the whole volume should be processed each time a new differential change in the tensor $\vec{\sigma}$ appeared, leading to unacceptable response times for an interactive system.

Therefore, for the material to reflect the permanent changes occurred when the system enters the elasto-plastic load zone, the proposed deformation algorithm shall progressively operate with the volume from the initial state of the deformation (localized changes in the surface) up to its final state (deformation propagated throughout the material).

Whenever a change in the surface is produced, the position of the tool in the *new* surface ($\vec{x}_{i+1}$ in Fig. 2) is passed to the deformation algorithm, which shall be noted in the following as $\vec{c}_i$. Since the force estimation and deformation algorithm run at different speeds, and also due to the fact that deformation is much slower that estimation, subsequent changes have to be *queued*, for the deformation algorithm to process them when possible. Considering all this, deformation will work as follows:

1. While there are positions $\vec{c}_i$ in the queue:
   a. Move the property value of the voxels in $t(\vec{c}_i)$ in the direction of the estimated displacement at that point. Add the removed property value to the next voxel in that direction. After deletion, $t(\vec{c}_i)$ must be empty, creating a *hole* in the material.
   b. Put every modified voxel in the *active voxels* list.
2. While there are no new positions $\vec{c}_i$ in the queue and there are voxels in the *active voxels list*:
   a. Take the active voxel with the highest property value.
   b. Compute the mean density in the neighbourhood of the voxel.
   c. Change the voxel property value back to the *normal* value, being the normal value an arbitrary value intrinsic to the material.
   d. Send the remaining material to neighbours which have a property value lower than the mean density, updating them to the mean density.
   e. Put every modified voxel in the *active voxels* list.
3. Go to step 1.

In short, the deformation algorithm first *compresses* the material where the tool pushed enough to deform the surface, and, when possible, starts smoothing this overload throughout the volume until everything returns to a relaxed state.

Doing such an incremental processing provides two main advantages. First, it gives an interactive response, as valleys are instantly created in the surface — thanks to the fact that deletion is done with the highest priority. Second, it makes

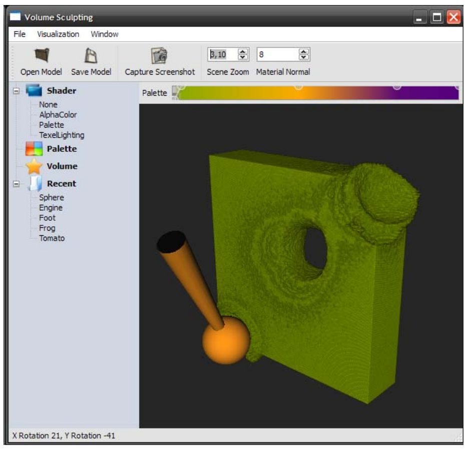

Figure 3: *Test sculpting application.*

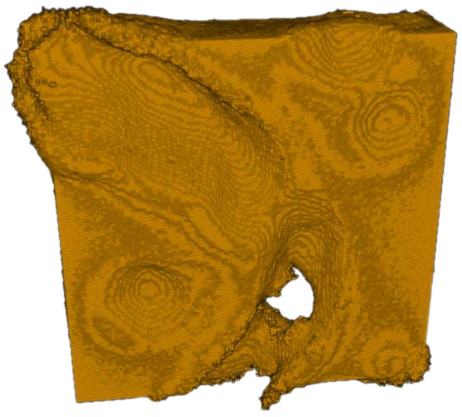

Figure 4: *Strongly deformed clay.*

the algorithm to be independent from the volume size: no matter how big the volumetric data is, interactive results can be achieved.

5. Results and conclusions

In Fig. 1 and Fig. an example of the results achieved with the described method is shown. Fig. 3 shows the sculpting application which implements the work described here. The interaction depicted there shows an example of a topological change in the solid, and also the modifications produced in the nearby of the zone where the tool deformed the surface.

Table 1: *Framerates for the force computation thread*

Volume \Tool	16	32	64
128^3	4654Hz	566Hz	74Hz
256^3	4612Hz	567Hz	71Hz
512^3	4567Hz	560Hz	67Hz

Performance rates for the force computation thread are given in Table 1. For a 512-voxel sided volume, deletion is performed at about 2kHz using a 16 voxel diameter spherical tool, 250Hz for a 32 voxel sized tool and at 24Hz for the 64 diameter tool. The relaxation of the volume is done at a mean speed of 217.000 voxels/second. Speeds were obtained using a Intel Core 2 Quad CPU at 2.93GHz with 3.50GB of RAM.

Results show that both force computation and deformation rates are only tool-size dependent. Force is computed locally just using the tool implicit volume, and deformation is progressively computed, no matter the size or nature of the interaction. Together with the natural feedback supported by the theoretical background, the contribution of this paper turns out to be a robust, plausible method for plastic deformation.

References

[CSH03] Choi K.-S., Sun H., Heng P.-A.: Interactive deformation of soft tissues with haptic feedback for medical learning. *Information Technology in Biomedicine, IEEE Transactions on 7*, 4 (Dec. 2003), 358–363.

[DA04] Delingette H., Ayache N.: Soft tissue modeling for surgery simulation. In *Computational Models for the Human Body*, Ayache N., (Ed.), Handbook of Numerical Analysis (Ed : Ph. Ciarlet). Elsevier, 2004, pp. 453–550.

[DC04] Dewaele G., Cani M.-P.: Virtual clay for direct hand manipulation. In *Eurographics (short papers)* (2004).

[Del98] Delingette H.: Towards realistic soft tissue modeling in medical simulation. *Proceedings of the IEEE : Special Issue on Surgery Simulation* (April 1998), 512–523.

[FG99] Frisken-Gibson S. F.: Using linked volumes to model object collisions, deformation, cutting, carving, and joining. *IEEE Transactions on Visualization and Computer Graphics 5*, 4 (1999), 333–348.

[KP04] Kim L., Park S. H.: A haptic sculpting technique based on volumetric representation. *Articulated Motion and Deformable Objects* (2004), 14–25.

[MQW01] McDonnell K. T., Qin H., Wlodarczyk R. A.: Virtual clay: a real-time sculpting system with haptic toolkits. In *I3D '01: Proceedings of the 2001 symposium on Interactive 3D graphics* (New York, NY, USA, 2001), ACM Press, pp. 179–190.

[NC10] Natsupakpong S., Cenk M.: Determination of elasticity parameters in lumped element (mass-spring) models of deformable objects. *Graphical Models 72*, 6 (2010), 61 – 73.

[NPF05] Nesme M., Payan Y., Faure F.: Efficient, physically plausible finite elements. In *Eurographics (short papers)* (august 2005), Dingliana J., Ganovelli F., (Eds.).

[OBH02] O'Brien J. F., Bargteil A. W., Hodgins J. K.: Graphical modeling and animation of ductile fracture. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 291–294.

[OdSB06] Olivella X. O., de Saracíbar Bosch C. A.: *Mecánica de medios continuos para ingenieros*. Universitat Politècnica de Catalunya, 2006.

[Pre84] Prevost J. H.: Localization of deformations in elastic-plastic solids. *International Journal for Numerical and Analytical Methods in Geomechanics 8*, 2 (1984), 187–196.