Computers in Biology and Medicine 83 (2017) 84–93

Contents lists available at ScienceDirect

Computers in Biology and Medicine

journal homepage: www.elsevier.com/locate/compbiomed

A new deformation simulation algorithm for elastic-plastic objects based on splat primitives

Yanni Zou[a], Peter X. Liu[a,b,*]

[a] The School of Information Engineering, Nanchang University, Nanchang, Jiangxi 330031, China
[b] The Department of System and Computer Engineering, Carleton University, Ottawa, Canada K1S 5B6

ARTICLE INFO

Keywords:
Elastic-plastic model
Fracturing
Splat
K-means
Degrees of freedom

ABSTRACT

To achieve high computational efficiency and realistic visual effects, a new simulation algorithm for soft tissue deformation, which is based on a shape-matching scheme using splat primitives, is presented for interactive real-time applications, such as surgery simulation and video games. The most important novelty of the proposed approach lies in the fact that surface splats instead of points are employed in the computation of the deformation and fracturing of an elastic-plastic object. By controlling the sampling density and automatically adjusting the size of the circular splats, the surface of the simulated object can be seamlessly covered with a much small number of splats than points. Splats are then divided into clusters using the K-Means clustering algorithm. As a result, the elastic-plastic deformation of these clusters can be simulated using a shape-matching strategy, allowing more degrees of freedom (DOFs) in the simulation. Experimental results demonstrate that the proposed algorithm enormously reduces memory space and greatly improves computational efficiency (approximately twice in simulating plastic deformations compared with classical shape-matching methods), making it more suitable for interactive and real-time applications.

1. Introduction

Over the past two decades, the physically based modeling of deformations in virtual simulation has attracted wide attention in computational physics and computer graphics. The primary objective is to provide real-time and realistic visualization of deformations of common materials such as cloth, animal skins, or human soft tissues [1–6]. The most commonly used physically based models include the mass-spring system (MSS), the finite element method (FEM) and meshless methods. The MSS is an easy-to-understand technique in modeling deformable objects, because the deformation of an object is described as the displacement of mass points, evoked by internal and external forces. A large variety of MSS models have been developed because of their simplicity of implementation and relatively low computational complexity [7,8]. The MSS models, however, have inherent limitations, because of the low-level of realism of their deformation simulation because spring and damper parameters are not closely related to the constitutive laws of elastic materials. To achieve better visual effects, FEMs and meshless models are preferred because they accurately describe the behaviors of a wide range of materials compared with MSS models. For example, an example-based approach to simulating elastic materials by calculating the objects trajectory with FEM was proposed in [9], and a physical model derived from continuum mechanics was reported in [10]. In the above two models, the elastic and plastic deformation of volumetric objects are simulated by solving the equations of motions using both explicit and implicit integration schemes.

Although physically based methods provide relatively accurate and realistic simulations of the deformation of an object, they suffer from the expensive computational cost needed to describe the complex material properties of objects. Many methods have been developed to improve computational efficiency, and they can be roughly divided into two categories. The key idea of the first category is to simplify the data model. James et al. [11], for instance, designed an ArtDefo system with the boundary element method, which simulates the elastic deformation of an object with only surface points. The method was also extended by some other researchers [12,13]. Müller et al. [14] proposed techniques to use the finite element method based on cubical elements, which were represented with only surface meshes for simulating physical effects such as motion, deformation and fracture of an object. The calculation speed of these approaches is much faster than the methods involving volumetric data, but the accuracy and authenticity of the simulation were compromised to a large degree. The other category uses a data-driven approach. For instance, Bickel et al. [15] introduced a method

* Corresponding author at: The School of Information Engineering, Nanchang University, Nanchang, Jiangxi 330031, China.
E-mail addresses: zouyanni@163.com (Y. Zou), xpliu@sce.carleton.ca (P.X. Liu).

http://dx.doi.org/10.1016/j.compbiomed.2017.02.007
Received 27 April 2016; Received in revised form 23 January 2017; Accepted 17 February 2017
0010-4825/

for simulating the non-linear heterogeneous deformation of soft tissue by acquiring a set of example deformations of a real tissue and representing all of them as spatially varying stress-strain relationships in a finite element model. Seiler et al. [16] also applied this method to their simulation. The major limitation of the data-driven methods comes from the fact that valid and useful data are difficult to obtain. In fact, these algorithms are seldom used for real-time applications (such as surgery simulation and animation game) because of their respective limitations mentioned above.

In order to satisfy the real-time and stability requirements of such a system, Müller et al. [17] proposed a meshless shape-matching (MSM) scheme, which does not fit into any of the preceding categories. The object was approximated by a simple particle system without any link between points. In every time step of the simulation, each particle moves independently under external forces towards its target position. The MSM method has good stability, strong controllability, and high efficiency, and a number of improved MSM algorithms have been proposed in recent years. For instance, Rivers et al. [18] proposed the so-called FastLSM algorithm, which used a region-based convolution of rigid shape-matching transformations on cubic lattices, and many clusters were overlapped for smooth deformation. The computational cost of this algorithm, however, is increased exponentially for updating the data in clusters when large topology changes occur. Steinemann et al. [19] extended the FastLSM algorithm by using an octree to achieve dynamic adaptive selection of the levels of details. The octree adopted in their work, however, is also time-consuming when the topology of an object is changed. Rungjiratananon et al. [20] modified the FastLSM algorithm to simulate complex hairstyles. Yuki et al. [21] presented an elastic deformation method based on examples, which uses a shape-matching framework. Ijiri et al. [22] generated active animations of unarticulated objects based on the idea of shape matching by expanding and contracting local regions. Gerszewski et al. [23] proposed an approach to animate elastoplastic materials based on point primitives by computing the deformation gradient of each particle.

The key idea of the above MSM-based algorithms is to compute the new location of each moved point, rather than to re-mesh or to recalculate the changed topology. Although more points generally give better deformation simulation results, this also consumes more memory and requires more time for calculation. As a fact, it is still very difficult to achieve real-time performance when large-scale models are used with these methods.

In order to overcome the problem faced by the current MSM-based methods, this paper presents a novel and improved scheme for shape-matching based on splat primitives. A splat is a rendering primitive in the form of a disk, that is aligned perpendicular to the surface normal. A point-sampled surface model can be rendered by creating many splats. The surface splatting approaches are already reported in [24,25] for rendering purposes. However, their surface splat is obtained by using a screen space formulation of the Elliptical Weighted Average filter, which is different with the proposed method. Their ultimate goal is only high rendering quality, while our aim is less computation units and appropriate visual effect, which is achieved by selecting a much small number of splats than points to seamlessly covered the surface of the simulated object. The main contributions of our scheme compared with similar work in the literature (such as [26,27]) can be summarized as follows:

- Data points are represented using circular splat primitives. As a direct benefit, the implementation requires much less memory space compared with conventional ones.
- Both the calculation and rendering of deformation are based on splat primitives instead of point primitives, so the number of primitives that need to be computed is significantly reduced, significantly improving computational efficiency.
- The proposed deformation model can be extended to the cases of linear, quadratic, plastic and fracturing deformation. As a result, the algorithm can simulate more degrees of freedom (DOFs) of deformation by clustering splats into multiple regions.

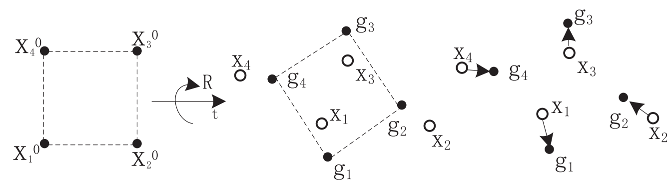

Fig. 1. Meshless shape matching.

This paper is organized as follows. Section 2 describes the key idea of the meshless shape-matching algorithm. In Section 3, we propose the algorithm based on splats, which can simulate elastic-plastic deformation and fracturing. Section 4 shows that the proposed method provides superior quality regarding computational efficiency compared with classical shape-matching based on point primitives. The paper concludes with suggestions for future work in Section 5.

2. Point-based deformation through shape-matching

In this section, we first introduce the shape-matching method [17], as well as its variants, which forms the basis of the proposed method. As shown in Fig. 1, the key idea is to approximate an object with a set of particles with masses m_i and initial positions x_i^0, where i is the serial number of a particle. No connectivity information or mesh is needed. In each time step, the particle i is moved independently based on its external forces, and then pulled to its target position g_i. The target position is calculated by assuming that the object is a rigid body and matching its current positions with its initial positions. Specifically, given a set of simulation particles, with x_i^0 and x_i as their initial and deformed positions, a rotation matrix R and the translation vectors are computed to minimize the sum of the squared distance of each particle i between the initial and deformed configurations. The translation vectors are given as centers of mass, $x_{cm}^0 = \frac{\sum_i m_i x_i^0}{\sum_i m_i}$ and $x_{cm} = \frac{\sum_i m_i x_i}{\sum_i m_i}$. The linear transformation

$$A = (\sum_i m_i p_i q_i^T)(\sum_i m_i q_i q_i^T)^{-1} = A_{pq} A_{qq} \tag{1}$$

is computed first, with $q_i = x_i^0 - x_{cm}^0$ and $p_i = x_i - x_{cm}$, and A_{qq} is a symmetric matrix containing only scaling. The rotation matrix R is extracted from A using the polar decomposition A=RS. Next, the target position of each particle i is calculated as

$$g_i = R(x_i^0 - x_{cm}^0) + x_{cm} \tag{2}$$

Finally, the position x_i and velocity v_i are updated as

$$v_i(t+h) = v_i(t) + \alpha \frac{g_i(t) - x_i(t)}{h} + h f_{ext}(t)/m_i \tag{3}$$

$$x_i(t+h) = x_i(t) + h v_i(t+h) \tag{4}$$

where t is the present time, h is the time step, f_{ext} is the external force and $\alpha (0 < \alpha \le 1)$ is the stiffness parameter.

According to the equations cited above, only small deviations from the rigid shape can be simulated. To obtain an extension of the range of motion and to simulate a soft body, Eq. (2) is extended to

$$g_i = (\beta A + (1-\beta)R)(x_i^0 - x_{cm}^0) + x_{cm} \tag{5}$$

where A is a linear transformation matrix, that expresses the best linear transformation of the initial shape to match the actual shape in the least squares sense. $\beta (0 \le \beta \le 1)$ is a control parameter. If β is large, then the object becomes soft; otherwise, the object becomes hard.

However, a linear transformation can only represent stretch and shear, which does not satisfy the application requirements. To simulate