Proceeding of the IEEE
International Conference on Information and Automation
Yinchuan, China, August 2013

A GPU-Accelerated Finite Element Solver for Simulation of Soft-Body Deformation

Jianying Li*, Yu Guo*, Ping Liu, Qiong Wang, Jing Qin
Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
Shenzhen, China 518055
The Chinese University of Hong Kong
Email: {*jy.li, yu.guo, ping.liu, wangqiong, jing.qin*}*@siat.ac.cn*

***Abstract*—A nonlinear physical simulation is presented involving the soft body deformation and interaction contacts. We demonstrate the finite element method relying on Lagrangian discretization to simulate the deformation of the soft body with hyperelastic material properties. To perform a stable simulation, we use the constrained Delaunay Tetrahedralization to resampling and remeshing the object. A new contact strategy is developed and used to replace the collision detection. This method does not need to iteratively achieve the optimal contact response on the constrained boundary. It can dynamically determine whether the contact status of the point should be in a static or a sliding friction mode. The explicit method for the finite element model is employed in order to perform all the steps of the algorithm on the GPUs and achieve a real-time simulation.**

I. Introduction

Nowadays, soft body simulation has become an very important part in the animation filming, game development and virtual reality system creation. To achieve a realistic simulation of the soft body deformation, the Finite Element Method (FEM) is widely used. It is one of the most effective numeric methods to solve the linear and nonlinear multi-dimensional problems, because it allows the modeling of the systems with complex geometries and irregular physical structures in a high precision. To solve the dynamic problems with the FEM, two kind of time integration schemes are frequently used, implicit and explicit. The explicit method is easier to obtain the result than the implicit method.

One disadvantage of the FEM is that it needs long computation time, especially for the nonlinear systems. In addition, its implementation is a challenge, as it requires experience to define the problems such as the mesh generation. An accurate three dimensional model with a fitting mesh approximating to the finite element model is the basis of the simulation. Accurate modeling and computational efficiency is a dilemma for the dynamic simulation. To solve this problem, a rich and diverse studies have been concerned on accelerating the FEM on Graphic Processing Units (GPUs), especially since the Compute Unified Device Architecture (CUDA) was released by NVIDIA in 2007. The Nvidia's CUDA API is widely used in the programmable GPUs, because it allows the programmer directly control the computing and memory resources.

Besides, the texture memory developed in the graphics cards provides a transfer probability from the CUDA API to OpenGL graphics API. Due to this property, the results of the vertex can be effectively displayed on the screen from the texture tunnel. Furthermore, the cores in Kepler architecture provided since 2012 are three times faster than that in the last Fermi GPU, and texture units are twice larger. With the help of the high speed development of the hardware, there is no doubt that it would provide more powerful tools to solve the gigantic calculations of soft body simulation.

The external forces of the deformation are mainly generated from the collision. Compared with the collision detection of the rigid bodies, collision detection of the soft bodies is much more complex. Not only the finite element analysis should infer the deformation of the soft bodies in each time step, but also the constrained optimization problem at the contact requires to be considered.

In this paper, we proposed an efficient numerical algorithm to compute the deformation of the soft body with constrained boundary in real time. This algorithm is based on the finite element method using the explicit scheme and a projection contact model with the Constrained Delaunay Tetrahedralization. We also gave a fast collision detection model fitting for the corresponding computation on GPUs. The whole algorithm is accelerated on GPUs to obtain a real-time deformation simulation of the soft body.

II. Related Works

The FEM is the classical method to solve the elastic mechanics problem. With the development of the hardware, it becomes the mostly popular method to simulate soft body deformation for now. In 2011, Faure [1] introduced a meshless model using the FEM and Simonovski [2] meshed the object with a set of the spatial points. Irving [3] proposed a deformable model based on the FEM to enforce the local incompressibility. Euler method is usually used to simulate the large deformation, such as the water simulation. Levin [4] resolved the frictionless interaction among multiple deformable objects with the Eulerian simulator. Miller [5] used the total Lagrangian FEM to simulate the soft tissue deformation. Sueda [6] combined the Lagrangian and the Eulerian approach, to handle the large-scale simulation of highly constrained strands.

To solve the dynamic problems with the FEM, two time integration methods are frequently used, implicit and explicit method. The advantage of implicit time integration is its good stability, such as the Newmark method [7] and the Wilson

* joint 1st authors

978-1-4799-1334-3/13/$31.00 ©2013 IEEE

θ method [8]. For the linear problems, the time step can be arbitrarily large; for the nonlinear problems, the time step size must be very small due to the convergence difficulties. In the explicit time integration, it will be stable only if the time step size is smaller than the critical time step size. Due to this, the explicit method is very useful for the deformation simulation occurring in short transient.

Soft bodies generally exhibit nonlinear and elastic properties under the large deformations. Linear elastic model is adequate to the small deformation simulation and is easier to solve. But in real condition, the material is more complicated and the linear model is insufficient. Müller [9] presented a mesh-free animation algorithm that could simulate a wide range of elastic and plastic deformations. Non-linear hyperelastic model and anisotropic behavior is closer to the materials in reality. For the simplification, isotropic or orthotropic behavior was used in the special models. Tanveer [10] presented a mixed *p*-type method to solve incompressible non-linear transient vibration. In addition, a viscoelastic behavior of soft bodies has been implemented in the finite element method recently. Taylor [11] established an efficient procedure to simulate anisotropy and viscoelasticity materials. In different models, different constitutive equations were proposed to indicate the relationship between strain and stress. Gilles [12] presented a frame-based meshless model with arbitrary constitutive laws.

For the collision detection, the recent algorithms presented were mainly based on iterative and local optimization techniques to resolve the soft or solid bodies contact problem. kaufman [13] introduced a non-smooth contact model based the simulation of rigid bodies. Bertails [14] introduced a method to compute self-contacts with the presence of the friction. Daviet [15] introduced a robust iterative solver to compute the friction effect. Je [16] solved the Linear Complementarity Problem (LCP) with the Gauss-Seidel iteration.

To achieve the fast computation and real time interaction, parallel computing using the hardwares like graphics processing units is a new trend. Taylor [17] accelerated the total Lagrangian method with GPUs. Allard [18] applied their collision detection method, contact modeling and constrained solver for the animation simulation on GPUs. NVIDIA [19] simulated a perfect water flow using the Eulerian method with the graphic card. Dick [20] tested a hexahedral-based elasticity simulation using CUDA on Fermi GPU, and achieve 14 times of speeding up compared to a 2 cores CPU. Hahmann [21] presented an explicit formulas to solve the volume preserving free form deformation with GPUs.

III. Methods

Before applying the finite element algorithm to the soft body deformation, a three dimensional Delaunay Mesh Generation method [22] was used to create the tetrahedralized model of the object. With this method, soft body can be properly splitted into tetrahedras with stable properties.

A. Finite Element Simulation

In the proposed approach, the deformation progress is simulated with a nonlinear finite element method. Given the initial coordinates of the nodes $\{\boldsymbol{x_0}\}$, the spatial positions of the nodes need to be updated in each time step. Therefore, the most important thing is to evaluate the global nodal displacements. For the displacements, we use a $3n$-vector $\boldsymbol{u}$ to represent.

From the principle of the FEM method, we consider a solution using the standard equilibrium equations for a nonlinear, dynamic, damped finite element model:

$$\boldsymbol{M}\ddot{\boldsymbol{u}} + \boldsymbol{D}\dot{\boldsymbol{u}} = \boldsymbol{r} - \boldsymbol{f}(\boldsymbol{u}) \tag{1}$$

where $\dot{\boldsymbol{u}}$ is the velocity vector of the nodes, $\ddot{\boldsymbol{u}}$ is the acceleration vector, and dot notations indicate the time (t) derivatives. $\boldsymbol{M}$ is a $3n \times 3n$ mass matrix , $\boldsymbol{D}$ is the damping matrix with the same size, $\boldsymbol{f}(\boldsymbol{u})$ is the internal force vector dependent on the displacement vector of $\boldsymbol{u}$, and $\boldsymbol{r}$ is the vector of the external loads.

The force applied by the internal deformed element to the nodes is given by:

$$\boldsymbol{f} = \boldsymbol{B}^T\boldsymbol{\sigma} = \boldsymbol{B}^T\frac{\partial W}{\partial \boldsymbol{\varepsilon}} \tag{2}$$

where $\boldsymbol{B}$ is the strain-displacement matrix, $\boldsymbol{\sigma}$ is the stress, and $\boldsymbol{\varepsilon}$ is the strain. W is a strain energy density function decided by different constitutive model. A large number of the energy function have been proposed to solve the deformation of the soft body. From the this equation, we could see internal force $\boldsymbol{f}$ also relay on the constitutive model of the soft body.

In previous works [23][24], researchers usually use a linear elastic constitutive model to simplify the internal force computation by making $\boldsymbol{\sigma} = \boldsymbol{S\varepsilon}$, where $\boldsymbol{S}$ is the stress-strain matrix determining the linear relationship between $\boldsymbol{\sigma}$ and ε. Then, the internal force can be recomputed with follows:

$$\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{B}^T\boldsymbol{S\varepsilon} = \boldsymbol{B}^T\boldsymbol{SBu} = \boldsymbol{Ku} \tag{3}$$

where $\boldsymbol{K}$ is called the stiffness matrix, which makes the internal force vector linear with nodal displacement vector.

The problem of this linear strain approximation is that it cannot correctly model the rotation of the finite element during the deformation. In the contrary, it was verified in [25] that the quadratic strain could handle arbitrary large rigid body motions and the internal force $\boldsymbol{f}(\boldsymbol{u})$ does not need to be a linear term of the nodal displacement. Different kind of strain energy function W can be used to define the internal force. Therefore, in the current work, we choose to use a hyperelastic model, named Neo-Hookean constitutive model [17] to solve this problem.

For the external force $\boldsymbol{r}$, we consider the gravity, contact force and other forces exerted by the user. The mass matrix $\boldsymbol{M}$ is an approximation of the inertia property of the continuum, including the total mass and moment of inertia. Additionally, Rayleigh damping is employed in the equilibrium equations, then, we can obtain $\boldsymbol{D} = \alpha\boldsymbol{M} + \beta\boldsymbol{K}$.

B. Explicit method

Generally, soft material has small stiffness in all directions. This makes the explicit time integration scheme appropriate to our issues, because we can use a relatively large time step. Therefore, in this work, we use the central difference method to handle the time (t) derivatives of the displacement [5]:

$${}^t\dot{\boldsymbol{u}} = \frac{1}{2\Delta t}({}^{t+\Delta t}\boldsymbol{u} - {}^{t-\Delta t}\boldsymbol{u}) \tag{4}$$

$$ {}^{t}\ddot{\boldsymbol{u}} = \frac{1}{\Delta t^2}({}^{t+\Delta t}\boldsymbol{u} - 2\,{}^{t}\boldsymbol{u} + {}^{t-\Delta t}\boldsymbol{u}) \tag{5} $$

where Δt is the time step. Displacement ${}^{t}\boldsymbol{u}$ is for current time step, ${}^{t-\Delta t}\boldsymbol{u}$ and ${}^{t+\Delta t}\boldsymbol{u}$ is is for the previous and the next time step. With the equation $(1)(4)(5)$, we can deduce the following equation:

$$ (\frac{\boldsymbol{M}}{\Delta t^2}+\frac{\boldsymbol{D}}{2\Delta t})^{t+\Delta t}\boldsymbol{u} = ({}^{t}\boldsymbol{r}-{}^{t}\boldsymbol{f})+\frac{2\boldsymbol{M}}{\Delta t^2}{}^{t}\boldsymbol{u}-(\frac{\boldsymbol{M}}{\Delta t^2}-\frac{\boldsymbol{D}}{2\Delta t})^{t-\Delta t}\boldsymbol{u} \tag{6} $$

To solve this equation, the invertion of a large sparse matrix is necessary in each time step. We make an hypothesis that all mass of the tetrahedron is lumped to its four nodes, in which we could get a diagonalized $\boldsymbol{M}$. Let α be a constant and $\beta = 0$, then $\boldsymbol{D}$ equals to $\alpha\boldsymbol{M}$ and is also diagonal. The only thing we need is to precompute the $\boldsymbol{M}^{-1}$. The internal force ${}^{t}\boldsymbol{f}$ is handled as follows:

$$ {}^{t}\boldsymbol{f} = \sum_{e}\int {}^{t}\boldsymbol{B}^{T}\,{}^{t}\boldsymbol{\sigma}\,dV \tag{7} $$

that we first calculate the internal force for each element, and then sum them together to get the final internal force for each node.

With the previous procedure, the nonlinear finite element equation is simplified into a set of independent *algebraic* equations as follows:

$$ {}^{t+\Delta t}u_i = \gamma_1 M_{ii}^{-1}({}^{t}r_i - {}^{t}f_i) + \gamma_2\,{}^{t}u_i + \gamma_3\,{}^{t-\Delta t}u_i \tag{8} $$

where $\gamma_1 = 2\Delta t^2/(\alpha\Delta t + 2)$, $\gamma_2 = 4/(\alpha\Delta t + 2)$, $\gamma_3 = 1 - \gamma_2$. u_i is the i-th $(1 \leqslant i \leqslant 3n)$ component of $\boldsymbol{u}$, r_i and f_i are the external and internal force element corresponding to u_i. M_{ii} is the diagonal entry of $\boldsymbol{M}$, where $\boldsymbol{M} = diag(M_{11}, M_{22}, \cdots, M_{nn})$. Solving this equation systems doesn't require complex matrix calculation and is easier to be resolved with a parallel computing.

Explicit time integration scheme such as the central difference method can only maintain a conditionally stability. In linear elastic analyses, time step Δt should be limited within a certain area:

$$ \Delta t = \lambda\Delta t_{cr} = \lambda\frac{L_e}{c}(0 < \lambda \leqslant 1) \tag{9} $$

where L_e is the smallest characteristic length of one tetrahedron element and c is the dilatational wave speed. c is given by

$$ c = \sqrt{\frac{E(1-\nu)}{\rho(1+\nu)(1-2\nu)}} \tag{10} $$

where E is Young's modulus, ν is Poisson's ratio and ρ is density.

In the explicit time integration, there are two different kinds of integration styles. One is the Update Lagrangian formulation, in which stresses and strains rely on the condition of previous step; the other one is the Total Lagrangian formulation, in which stresses and strains are measured with respect to the original configuration. In this paper, we employ the latter one. This choice allows for the pre-computing of most spatial derivatives before the commencement of the time-stepping procedure.

C. Contact Constraints

Soft body deforms to different shapes under the effect of the external forces $\boldsymbol{r}$. $\boldsymbol{r}$ includes the constant gravity force and the contact force. In the deformation process, the stiffness of soft body limits the deformation of itself. On the other hand, many other objects also constrain the soft body and give contacts to change the original status. To find a locally optimal solution to locate the closest point, two projection approaches were developed to solve the problems under different situations. Locally optimal solution is obtained whenever the position of the node, $\boldsymbol{x}$ is in-contact.

Soft body contact with the LCS (Local contact surface) is simplified as the interaction between several points and the surface. Then, we consider the interaction between a single point and the LCS. At first, we rotate the whole model in order to make the normal of contact surface parallel to the y-direction. In Fig. 1 and 2, we illustrate two different kinds of contact models in two dimension.

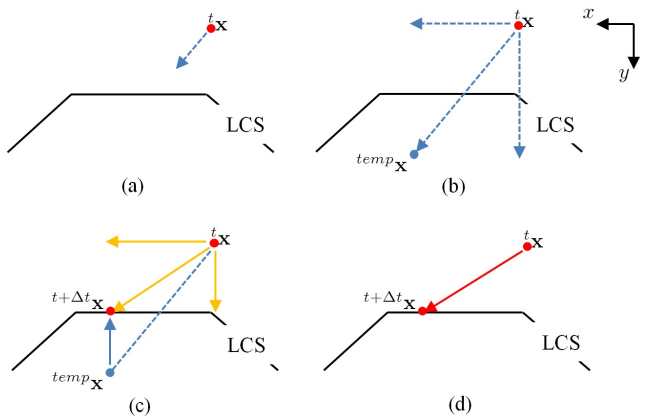

Fig. 1. The in-projection model of in-contact, ${}^{t+\Delta t}\boldsymbol{x}$ is the final in-contact position of the node.

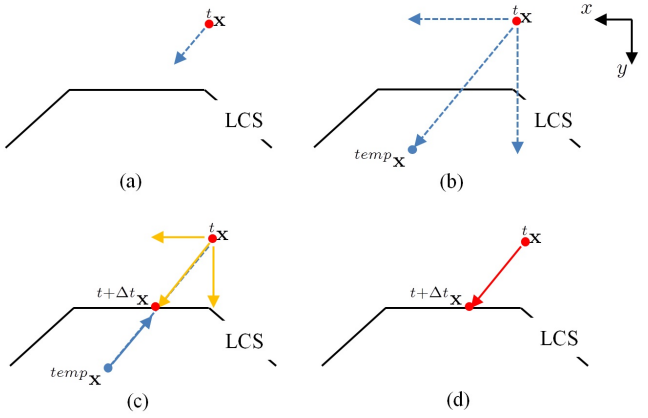

Fig. 2. The out-projection model of in-contact, ${}^{t+\Delta t}\boldsymbol{x}$ is the final in-contact position of the node.

In Fig.1 and 2, a node with the original position of ${}^{t}\boldsymbol{x}$ should be located at the position of ${}^{temp}\boldsymbol{x}$ during the moving, if there is no deformation or contact occurring. However, in reality, this node should be blocked by the contact surface, therefore we need to find a projection point on LCS for this

node. In the in-projection model, displacement of node ${}^t\boldsymbol{x}$ in x-direction is not changed but shortened in y-direction. It means that there is no friction in the x-direction between the contact surface and the node, and node is sliding on LCS. But in the out-projection model, displacements in both directions are shortened. This model simulates the situation that the node contacts with a rough surface and the node stops moving under effect of the static friction. For both of these two situations, node will not sticking or sliding for a long distance, because kinetic friction will finally take effect.

Taking three dimensional situation into account, for the position of each contact node ${}^t\boldsymbol{x}$, its displacement ${}^{t+\Delta t}\boldsymbol{u}$ in (8) will have an update:

$$ {}^{t+\Delta t}\boldsymbol{u} = {}^t\boldsymbol{u} + ({}^{t+\Delta t}\boldsymbol{x} - {}^t\boldsymbol{x}) = {}^t\boldsymbol{u} + [u_x, u_y, u_z]^T \qquad (11) $$

u_x stands for friction ranging from static friction to zero, u_y is a constant value, and u_z is same to u_x. We define ${}^{temp}\boldsymbol{x} - {}^t\boldsymbol{x} = [\hat{u}_x, \hat{u}_y, \hat{u}_z]^T$, and the updated displacements are represented as below:

$$ u_y = distance\ from\ {}^t\boldsymbol{x}\ to\ LCS $$

$$ (u_y/\hat{u}_y)\hat{u}_x = u_{xout} \leqslant u_x \leqslant u_{xin} = \hat{u}_x \qquad (12) $$

$$ (u_y/\hat{u}_y)\hat{u}_z = u_{zout} \leqslant u_z \leqslant u_{zin} = \hat{u}_z $$

D. Implementation in GPU

Algorithm 1: Deformation Simulation Step

```
Begin
Tetrahedralize the surface model to an element model
Load the surface and element models
Transmit nodes, surface and elements to device
Precompute the constant variables [CPU&GPU]
do{
    Compute mass matrix M and external forces r [GPU]
    Compute internal forces f [GPU]
    Compute displacement vector u [GPU]
    Update u with contact model [GPU]
    For new nodes, u'= old one + u [GPU]
    Bind nodes to texture memory, display [GPU]
}while (TRUE)
End
```

Using CUDA API and OpenGL API, the algorithm was implemented with GPUs. This tabular shows the complete progress involving the whole system of deformation simulation. In most of the previous research, the mass computing is generally done in the pre-computation procedure, because it changed a little in a small deformation. But in our method, with the shape changes of the tetrahedra elements during the deformation, the mass was allocated to different nodes in a dynamic mode. So, it's much better to take the mass computation into the GPU cycles although it will bring some complexities.

IV. Results

Stability of the GPU computing in the explicit dynamic solving system is the most essential problem to be considered. Cube, cylinder or ellipse suitable for the hexahedral mesh are usually be used, but to handle the complex geometric object, tetrahedral model is much more helpful. The accuracy of the deformation simulation is also what we need to focus on. It relates to the hardware ability, the number of element used, and the robust of the algorithm.

We firstly used a 10mm diameter torus model with 2145 vertex meshed by 7456 four-node tetrahedron with Tetgen (Fig. 3). We placed a solid floor 2mm lower than the torus model and only applied the gravity force to the model to observe contact performance between the soft torus and the solid ground. We used C++ with CUDA4.2 and OpenGL4.2 to develop the contact deformation program, and our experimental platform mainly includes a 2.4GHz Inter CPU, a 4GB of memory and a NVIDIA Geforce GTX480 with 1.5GB graphics memory, and etc.

TABLE I. Details of mesh in three kinds of torus models: outer radius is $10mm$; outer radius is $100mm$; outer radius is $10mm$ using subdivision

	r_{10}	r_{100}	r_{sub}
Input points	1729	1729	6916
Input facets	3458	3458	13832
Input segment	5187	5187	20748
Mesh points	2145	2136	8395
Mesh tetrahedra	7456	7396	27918
Mesh faces	16641	16521	62752
Mesh edges	11330	11261	43229
Boundary faces	3458	3458	13832
Boundary edges	5175	5185	20744

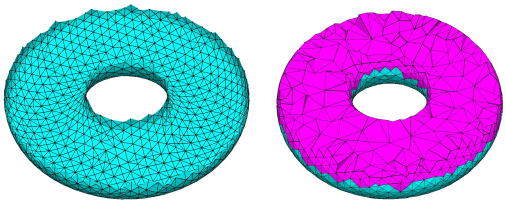

Fig. 3. Left: The PLS torus (1729 nodes, 3458 faces), the outer radius $r = 10mm$, the inner radius $r = 5mm$. Right: The output Delaunay mesh (2145 nodes, 7456 tetrahedra) at $\rho_0 = 2$.

A. Simulating the Accuracy

The material properties of the soft torus model is a key factor in the contact model in the measurement of the accuracy and the facticity of the proposed method. The parameters used in the experiment are approximate to the human organ. The density of the tours model is $0.001g/mm^3$, damping coefficient α is 10. Young's modulus of $3000MPa$ and Poisson's ratio of 0.49 for uncompressed material was used in the Neo-Hookean constitutive model. We employed a conservative time step about $0.000025s$ and the total time period is around $0.2s$. For the deformation of the torus, we also implement it with the ABAQUS/Explicit, in order to compare with the result using the proposed GPU-based dynamic explicit finite element method.

In Fig. 4, the above four figures show the displacements in x, y, z component and the whole displacement for a random node during the deformation process. The blue lines stand for the result obtained from ABAQUS and the red lines are the GPU-based results with the proposed method. It can clearly find that the distances between these two values are very close.

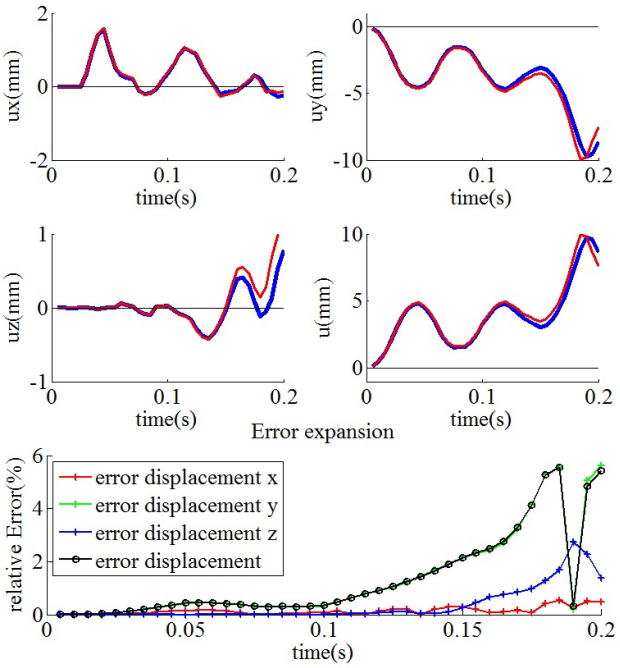

Fig. 4. The error expanded without friction

The figure on the bottom of Fig. 4 shows the demonstration of the errors between ABAQUS and our method. The average error of the 2145 nodes in the model is controlled within 0.1mm in the dynamic finite element method for a 20mm width object. The difference between the GPU-based method and the ABAQUS/Explict is lower than 0.5% in the first 0.1s. Our contact detection model is different from the ABAQUS, and the error after 0.1s begins to increase, but still controlled within a small range.

B. Simulating the Deformation

Fig. 5 shows the deformation of the torus during its falling down process. We divide the process into four periods.

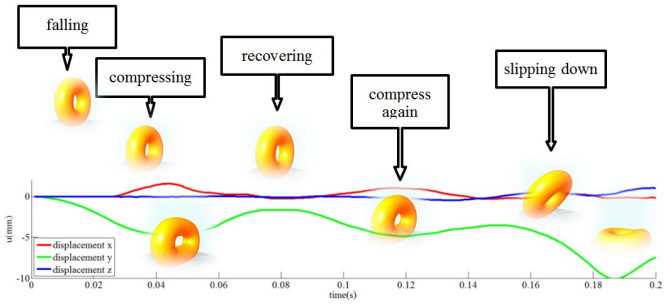

Fig. 5. The simulation of soft torus deformation

a) $0s \sim 0.03s$, *falling period:* In the first $0.03s$ period, the torus contacts with nothing but only accelerates to drop due to the gravity. It can be found that displacements in x and z-direction are barely zero, and its displacement only raises in the y-direction due to the falling process.

b) $0.03s \sim 0.07s$, *first compressing:* From $0.03s$, the torus touches the floor and the contact begins. Under the constraint of the floor, the torus is compressed in the y-direction. It is clear that the red line begins to increase, which represents that an expanding happens in the x-direction. At $0.45s$, the deformation is the largest that displacements in both x and y directions reach the highest. After this second, the torus begins to recover back to the normal size.

c) $0.07s \sim 0.15s$: *continued compressing:* At the time of $0.07s$, the torus almost turns back to normal size because the three lines reach to their smallest values. Then, the torus begins the second compressing cycle. The damping force makes displacements of the second deformation smaller than the first time. If the mesh of the model is absolutely symmetrical and the damping matrix is neglected, the compressing loop will never stop.

d) $0.15s \sim 0.2s$: *sliding down:* After several times of the compressing cycle (the number of times is decided by the friction), the torus model begins to slide down in an unbalance structure. During a series of oscillations, the torus reaches to a balance state at the last.

C. simulating different friction modes

The meshed torus model is not exactly symmetrical. After two contacts on floor, the center of gravity is deviated from the middle line. There produces a large velocity in the z-axis direction to pull the torus down. A contrast movement happens when the soft torus begins to fall down. In different models, it brings different results.

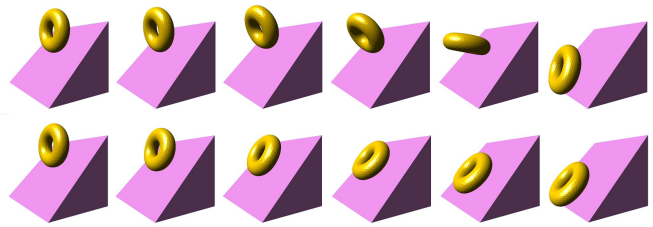

Fig. 6. Above: Fall over with static friction; Below: Sliding down with non-friction.

Fig. 6 shows what happens when a torus drops onto a slope surface. From the sequence of figures on the first line, the torus contacts with the slope and jumps up suddenly, then flips and falls over from the surface. This phenomenon can traced back to the model with static friction demonstrated in Fig. 2. On the bottom figures, the torus slides down along the slope surface with no friction, illustrating the model in Figure 1. This example shows that our constraint contact model has the ability to simulate the interaction force between deformable object and solid surface under different friction modes.

D. Computational Efficiency

We implemented this improved algorithm on GPUs, and compared it with the CPU algorithm. Table II shows the average time of 8000 iterations with different tetrahedron numbers. The computation time is nearly linear to the tetrahedron number when using the CPU algorithm. In contrast, the computation time of GPU algorithm is reduced around

100 times for the 7465 tetrahedrons model compared to the CPU algorithm. Also, the computational efficiency of the GPU algorithm improves greatly with the increasing of the number of tetrahedrons.

TABLE II. GPU COMPUTATION TIME

Tetrahedron No.	GPU Time (ms)					CPU Time(ms)
	$\mathbf{M}$	$\mathbf{f}$	$\mathbf{u}$	$\mathbf{u}'$	Total	
7456	50.0	2280.0	54.8	500.0	2884.8	298288
12584	102.8	4024.0	101.2	980.0	5208.0	666624
27918	190.0	6604.0	160.4	1788.0	8742.4	1186344

V. CONCLUSION

We presented a simulation method of the soft body deformation with GPU implementation. Based on the Delaunay Tetrahedralizition meshing algorithm, the finite element method with a contact model was improved and applied to present the friction efforts. Experiments using a torus model were conducted to demonstrate that the proposed algorithm, involving the isotropic and hyperelastic constitutive model, works well for the large deformation simulation.

One of the issues that we would like to explore in future is to add a locally tessellation into the model. During the soft body interactions, the part deforming hardly needs more meshes and details to be taken into account. Another possible direction is to further improve the contact model. In this paper, the compression happened between two surfaces is only simulated by the displacement boundary condition, which may occur a few of unrealistic deformations.

ACKNOWLEDGMENT

The work described in this paper was supported by several grants, including a grant from the National Natural Science Foundation of China (Project No. 61233012), a grant from Ministry of Science and Technology of the People's Republic of China under the Singapore-China 9th Joint Research Programme (Project No.: 2013DFG12900) and a Shen Zhen Basic Research Grant (Project No.: JCYJ20130402113127511) .

REFERENCES

[1] F. Faure, B. Gilles, G. Bousquet, and D. K. Pai, "Sparse meshless models of complex deformable solids," *ACM Transactions on Graphics*, vol. 30, no. 4, p. 1, Jul. 2011.

[2] I. Simonovski and L. Cizelj, "Automatic parallel generation of finite element meshes for complex spatial structures," *Computational Materials Science*, vol. 50, no. 5, pp. 1606–1618, Mar. 2011.

[3] G. Irving, C. Schroeder, and R. Fedkiw, "Volume conserving finite element simulations of deformable models," *ACM Transactions on Graphics*, vol. 26, no. 3, p. 13, Jul. 2007.

[4] D. I. W. Levin, J. Litven, G. L. Jones, S. Sueda, and D. K. Pai, "Eulerian solid simulation with contact," *ACM Transactions on Graphics*, vol. 30, no. 4, p. 1, Jul. 2011.

[5] K. Miller, G. Joldes, D. Lance, and A. Wittek, "Total Lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation," *Communications in Numerical Methods in Engineering*, vol. 23, no. 2, pp. 121–134, Aug. 2006.

[6] S. Sueda, G. L. Jones, D. I. W. Levin, and D. K. Pai, "Large-scale dynamic simulation of highly constrained strands," *ACM Transactions on Graphics*, vol. 30, no. 4, p. 1, Jul. 2011.

[7] K. Bathe, *Finite element procedures*, 1996.

[8] J. Kuan-Jung and E. L. Wilson, "An adaptive finite element technique for structural dynamic analysis," *Computers & Structures*, vol. 30, no. 6, pp. 1319–1339, Jan. 1988.

[9] M. Müller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa, "Point based animation of elastic, plastic and melting objects," in *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '04*. New York, New York, USA: ACM Press, 2004, p. 141.

[10] M. Tanveer and J. W. Zu, "Non-linear vibration of hyperelastic axisymmetric solids by a mixed p-type method," *International Journal of Non-Linear Mechanics*, vol. 47, no. 4, pp. 30–41, May 2012.

[11] Z. a. Taylor, O. Comas, M. Cheng, J. Passenger, D. J. Hawkes, D. Atkinson, and S. Ourselin, "On modelling of anisotropic viscoelasticity for soft tissue simulation: numerical solution and GPU execution." *Medical image analysis*, vol. 13, no. 2, pp. 234–244, Apr. 2009.

[12] B. Gilles, G. Bousquet, F. Faure, and D. K. Pai, "Frame-based elastic models," *ACM Transactions on Graphics*, vol. 30, no. 2, pp. 1–12, Apr. 2011.

[13] D. M. Kaufman, T. Edmunds, and D. K. Pai, "Fast frictional dynamics for rigid bodies," *ACM Transactions on Graphics*, vol. 24, no. 3, p. 946, Jul. 2005.

[14] F. Bertails-Descoubes, F. Cadoux, G. Daviet, and V. Acary, "A non-smooth Newton solver for capturing exact Coulomb friction in fiber assemblies," *ACM Transactions on Graphics*, vol. 30, no. 1, pp. 1–14, Jan. 2011.

[15] G. Daviet, F. Bertails-Descoubes, and L. Boissieux, "A hybrid iterative solver for robustly capturing coulomb friction in hair dynamics," *ACM Transactions on Graphics*, vol. 30, no. 6, p. 1, Dec. 2011.

[16] C. Je, M. Tang, Y. Lee, M. Lee, and Y. J. Kim, "PolyDepth: Real-Time Penetration Depth Computation Using Iterative Contact-Space Projection," *ACM Transactions on Graphics*, vol. 31, no. 1, pp. 1–14, Jan. 2012.

[17] Z. a. Taylor, M. Cheng, and S. Ourselin, "High-speed nonlinear finite element analysis for surgical simulation using graphics processing units." *IEEE transactions on medical imaging*, vol. 27, no. 5, pp. 650–63, May 2008.

[18] J. Allard, F. Faure, H. Courtecuisse, F. Falipou, C. Duriez, and P. G. Kry, "Volume contact constraints at arbitrary resolution," *ACM Transactions on Graphics*, vol. 29, no. 4, p. 1, Jul. 2010.

[19] N. Chentanez and M. Müller, "Real-time Eulerian water simulation using a restricted tall cell grid," *ACM Transactions on Graphics*, vol. 30, no. 4, p. 1, Jul. 2011.

[20] C. Dick, J. Georgii, and R. Westermann, "A real-time multigrid finite hexahedra method for elasticity simulation using CUDA," *Simulation Modelling Practice and Theory*, vol. 19, no. 2, pp. 801–816, Feb. 2011.

[21] S. Hahmann, G.-P. Bonneau, S. Barbier, G. Elber, and H. Hagen, "Volume-preserving FFD for programmable graphics hardware," *The Visual Computer*, vol. 28, no. 3, pp. 231–245, Jul. 2011.

[22] H. Si and A. TetGen, "A quality tetrahedral mesh generator and three-dimensional delaunay triangulator," *Weierstrass Institute for Applied Analysis and Stochastic, Berlin, Germany*, 2006.

[23] J. Tan, G. Turk, and C. K. Liu, "Soft body locomotion," *ACM Transactions on Graphics*, vol. 31, no. 4, pp. 1–11, Jul. 2012.

[24] M. Nesme, Y. Payan, F. Faure *et al.*, "Efficient, physically plausible finite elements," in *Eurographics*, 2005.

[25] Y. Zhuang and J. Canny, "Haptic interaction with global deformations," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 3. IEEE, 2000, pp. 2428–2433.