# Virtual Surgery Deformable Modelling Employing GPU Based Computation

**Pengfei Huang, Lixu Gu, Jingsi Zhang, Xiao Yu, Sizhe Lv, Zhennan Yan, Luyang Zhang, Hongshan Zhou**

Image Guided Surgery and Therapy Laboratory, Shanghai Jiao Tong University,
800 Dongchuan Road, Minhang District, Shanghai, 200240, China
*herofay@sjtu.edu.cn*

## Abstract

To achieve the real-time requirement of realistic deformable modelling, it is necessary to use the acceleration techniques such as GPU computing for FEM and employ the feasible hybrid structures in a virtual surgery simulation system. In this paper, we present a linear or nonlinear deformable model of soft tissue. In addition to the efficient meshing and basic finite element method, the high computation rate is achieved through two novel methods. Firstly, the major calculation work in the Conjugate Gradient solver for deformation is moved from the CPU to GPU in order to promote the calculation. Secondly, we apply the hybrid structures of deformable models, by fully calculating the volumetric deformation in the local operation part while only calculating the global deformation by medial representation method. Experiments have been given to show the feasibility and efficiency of the model.

**Key words**: Virtual Surgery, Deformable Model, GPU

## 1. Introduction

Virtual surgery is a promising application to train intern surgeons. It can reduce the costs and risks in surgical training. To pursue the idea of virtual surgery, it is necessary to model the human tissue in virtual reality, simulate the motions and deformations of the tissue and provide interactive interface for the manipulator. The intern surgeon will use common surgery tools to manipulate the artificial soft tissue through the interface of the computer.

Although the simulation of human soft tissue has a long history in biomedical engineering and computer science, physically realistic modelling and computation of soft tissue's deformation has been the bottleneck of many applications, especially virtual surgery system. So far many simulations of deformation have been implemented using simple models: Mass-Spring [1], linear elastic FEM [3], [4], Centerline or Skeleton [11], Medial Representation Model [12] and so on. These methods work well for simulations of very small strains and local deformations, but have poor accuracy for large global deformation modelling.

In previous researches, the efficiency and robustness of the models are the main interests for surgery simulation systems, but the accuracy is of less concern. Nowadays, in some virtual surgeries realistic simulation is more important. For example, in facial plastic surgery simulations, the doctor and the patient are concerned about what it will look like after surgery, while they need a realistic simulation with high accuracy.

Global deformation [2] is commonly happened in medical domain and worth simulating for virtual surgery, such as large twisting or bending of an object, which involves the entire body. We could take the human intestine as a good example, for this case: while observing the patient's intestine, it is inevitable that the patient will move and so will his/her intestine move dynamically. In this situation, which is different from other soft tissue's minor deformations like needle insertion, scalpel cutting or forceps nipping, the intestine will move globally.

Many problems in mechanics and physics lead to differential equations, and most of which are impossible to be solved in analytic way. Finite Element Method ([2]-[7]) is such a numerical method that subdivides the object to a finite set of primitives, such as tetrahedral mesh, with a physical equilibrium equation for each of them. With application of variational principle, it can transform to problems of solving large systems of linear equations. And Conjugate Gradient algorithm [8] is a popular algorithm for solving large sparse symmetrical linear systems.

In this paper, first, we would focus on simulation for deformation of soft tissue in virtual surgery using nonlinear FEM in contrast with linear FEM. Besides, we would consider collision detection [10], as well as the methods for reducing complexity and expense of computation. Then, we mainly introduce the acceleration techniques of GPU Computing and Hybrid Deformable Model Structures. At last, we would conclude our experimental results based on 3D kidney model and blood vessel model.

## 2. Improved Delaunay Meshing

To simulate a continuum solid object using computer,

we must discrete it into a number of elements firstly. In this paper, we generate tetrahedral mesh based on Delaunay criterion with improvement [15].

By far most of the tetrahedral meshing techniques are utilizing the Delaunay criterion. The criterion states that for n-dimensional cases a circum-sphere of each simplex within the mesh contains only the n+1 defining points of the simplex.

There are some open source tools that can create mesh for 3D objects maintaining Delaunay criterion, such as the Visualization Toolkit (VTK). But the VTK can't preserve initial boundary, and produces sliver tetrahedrons if the surface is complex.

We have improved the Delaunay algorithm to generate a well-proportioned, boundary preserved mesh for latter deformation. The following is the steps we should take: 1) find the circum-cube of the bound box which contains the source object represented by boundary points and facets, and mesh the cube into 5 tetrahedrons; 2) subdivide the circum-cube into small cubes and store all vertices of the cubes; 3) disarrange all points gotten in step 2 within a small range to decrease the probability of 4 points co-planarity; 4) generate interior points from output of step 3; 5) insert boundary points and interior points into the 5 tetrahedrons gained in step 1 according to the Delaunay criterion; then, check boundary points and topology preservation; last, deal with sliver tetrahedrons. We detect the sliver tetrahedrons by calculating the standard deviation of their edges and comparing with a defined criterion. If the standard deviation is bigger than criterion we would re-mesh the sliver tetrahedron with its neighbours until satisfying.

## 3. Nonlinear Finite Elements Deformation

The theory of elasticity is a fundamental discipline in studying continuum materials. It consists of equilibrium equations, kinematics equations, constitutive equations and boundary conditions. Synthesizing all those equations allows us to establish a relationship between the deformation of object and external force. But in most cases, an analytic expression of this relationship is impossible. Finite element method is one way to solve such problems.

FEM is one of the most popular and stable numerical methods in engineering analysis. In this paper, we only discuss elasticity in the context of FEM.

Using this method, we should follow these steps: discrete the volumetric solid into a number of finite elements, which has been discussed in previous section; select the displacement function, and we use linear interpolation in tetrahedral mesh; construct element stiffness matrices considering physical equilibrium; assemble element matrices into a global stiffness matrix; solve the system of equations; calculate other unknowns such as strains and stresses. In implementation, we only calculate displacements at vertices of each tetrahedron, the values at other points within the elements are interpolated by a weighed sum of all the nodal displacements.

### 3.1 Static Deformation

For static deformation, it is required to solve the following system of equations

$$R(a) = F \qquad (1)$$

where $a = [u_1 \quad v_1 \quad w_1 \quad \cdots \quad u_n \quad v_n \quad w_n]^T$ is the 3n-dimensional nodal displacement vector for 3D objects, and u, v, w are the corresponding displacement variables at given point; R(a), the internal force vector due to deformation; F, the external force vector.

Usually the engineering strain vector is defined as

$$\varepsilon = [\varepsilon_x \quad \varepsilon_y \quad \varepsilon_z \quad \gamma_{xy} \quad \gamma_{yz} \quad \gamma_{zx}]^T. \qquad (2)$$

Many previous works ([3], [4]) using linear strain as following:

$$\varepsilon_x = \frac{\partial u}{\partial x} \qquad (3)$$

$$\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \qquad (4)$$

where x, y, z are the independent variables of the Cartesian frame. Other terms of the strain are defined similarly.

This linear strain vector makes the internal force vector R(a) linear to the nodal displacement vector a. Namely equation (1) can be written as following linear system:

$$Ka = F \qquad (5)$$

where K is the constant stiffness matrix independent with $a$.

However, the approach is only suitable for simulating small local deformations, e.g., poking and small bending. Using the linear strain to simulate large global deformations will cause distortion. It is because that this linear strain models rigid motions as differential motions. If we subject a large rigid body bending to an un-deformed object, the linear strain (3) and (4) will give a non-zero strain, but the object should not have any deformation actually.

In virtual surgery, large global deformations are crucial. To simulate global deformations, we use quadratic strain instead. So the strain vector become following expressions:

$$\varepsilon_x = \frac{\partial u}{\partial x} + \frac{1}{2}\left[ (\frac{\partial u}{\partial x})^2 + (\frac{\partial v}{\partial x})^2 + (\frac{\partial w}{\partial x})^2 \right] \quad (6)$$

$$\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} + \left[ \frac{\partial u}{\partial x}\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\frac{\partial v}{\partial y} + \frac{\partial w}{\partial x}\frac{\partial w}{\partial y} \right] \quad (7)$$

This nonlinear strain vector makes the internal force vector R(a) dependent to a. So equation (1) now becomes a nonlinear system. We can rewrite R(a) as the following expression:

$$R(a) = (K + P(a)) \cdot a \quad (8)$$

where K is the same as the constant matrix derived from linear strain, P(a) is the term depends on the nodal displacement vector a.

To solve this nonlinear system of equations, we use Newton-Raphson method as following:

$$a^{(n+1)} = a^{(n)} + \Delta a^{(n)} \quad (9)$$

$$\Delta a^{(n)} = (K + P(a^{(n)}))^{-1} \cdot (F - R(a^{(n)})) \quad (10)$$

This is an iterative numerical solution, and the main cost is calculating the inverse of matrix. Because the stiffness matrix is large, sparse and symmetric, we use Conjugate Gradient [9] method to solve the system of equations.

The simulations of static linear versus nonlinear strain deformation for an elastic beam are shown in fig.1. In the picture, the green mesh represents the original state, while the yellow and red ones as the state caused by body force, for example gravitation. We can see that the yellow mesh is stretched under the body force, which is distortion, while the red one simulates the deformation realistically.

Soft tissue is much more complex, and usually there are many large motions in surgery. So, if we want to get realistic simulations for soft tissue's deformation, the nonlinear strain models should be applied.

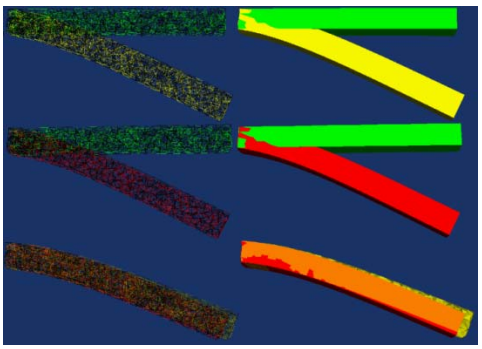

**Fig. 1.** Simulations for static linear strain deformation and nonlinear strain deformation are shown respectively. The yellow mesh is linear strain deformation while the red one is nonlinear strain deformation.

### 3.2 Dynamical Deformation

To model dynamical deformation of elastic objects, we should solve the following system of differential equations:

$$M\ddot{a} + D\dot{a} + R(a) = F \quad (11)$$

where $\ddot{a}$ , $\dot{a}$ are the respective acceleration and velocity vectors; M, the mass matrix, D, the damping matrix. Actually, equation (11) can become equation (1) leave out of account the acceleration and velocity vectors.

We have to solve the system of differential equations (11) approximately by numerically integrating along time dimension. Namely we have to solve the system of equations at a time step according to the previous time steps.

In this paper, we use center difference method, which is simplicity of Newmark recurrence scheme. It's a single-step explicit integration method, namely it only use values at t(n) to solve the system at $t_{(n+1)} = t_{(n)} + \Delta t$ . So we use the following iterative expression:

$$(\frac{1}{\Delta t^2}M + \frac{1}{2\Delta t}D)a_{t+\Delta t} =$$
$$F_t - (R(a_t) - \frac{2}{\Delta t^2}M)a_t - (\frac{1}{\Delta t^2}M - \frac{1}{2\Delta t}D)a_{t-\Delta t} \quad (12)$$

We should solve the system at each time step to get some displacements for dynamical simulation. But fortunately, the mass matrix M and the damping matrix D usually are constant, and we can approximate M by a diagonal matrix [5], then apply Rayleigh damping D=λM. These simplify the computation very much. We only need to recalculate the external force F and internal force R(a) at each step. For soft tissue, the time steps can be large. Besides, we need to initialize $a_0$, $\dot{a}_0$, $\ddot{a}_0$ and calculate the following value:

$$a_{-\Delta t} = a_0 - \Delta t \cdot \dot{a}_0 + \frac{\Delta t^2}{2}\ddot{a}_0 \quad (13)$$

It's necessary to point out that the time step $\Delta t$ must not be larger than a critical time step which has relationship with the smallest element in the mesh [5]. So we want a nice mesh without too small elements or sliver tetrahedrons for soft tissue, that's what we have discussed in section 2.

### 3.3 Efficient Collision Detection by Hashing

Collision detection is very important in any realistic interactive environment, so as in virtual surgery. Generally, collision detection algorithms need to deal with broad-phase which determines all potential collision

pairs and narrow-phase which tests intersection between two complex geometrical models.

However, it is challenging when regarding the deformable models. Here we consider an adaptive multi-resolution spatial partitioning algorithm [10] to detect collision occurring between a simple rigid object and a complex deformable body. It divides the space into a number of grid-cells and uses a hash table to store the position information of the deformable objects and their vertexes. We should find an optimal cell-size for each object and map each cell and vertex to a unique address in hash table. A 2D example is shown in fig. 2. In simulation, the grid-cells each object occupies are computed at each time step and all occupied cells have references to the corresponding objects. Then, for each vertex in the scene, test it for intersection with other objects that occupy the same grid-cell.

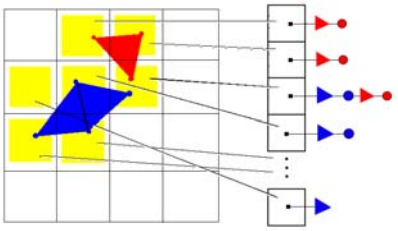

**Fig. 2.** Map cells occupied by triangles and the vertexes to Hash table. Vertexes and triangles in the same Hash table index should be tested for intersection.

## 4. High Precision GPU Numerical Computation

The meshing structure of the deformable model is changing during the simulation. The relevant equation is constructed and solved in every different processing step. Along with the size of the model increasing, the time spent by the collision detection rises as a linear function of the model scale. And the time of constructing the new global stiffness matrix is squared. Solving the linear system is about cubed complexity, which takes almost seventy to eighty percent of the time executed by the operation area. Therefore, solving the linear algebraic equations is the bottleneck of the whole simulation. It directly impacts the feedback speed of the system. In our system, we use the Conjugate Gradient iteration to solve the linear equations. The global stiffness matrix derived from FEM exhibits a feature of large size, sparse distribution of non-zero elements, symmetric and positive definite, this sufficiently meets the requirement of the condition of ordinary Conjugate Gradient algorithm.

The main computation in each round of the Conjugate Gradient algorithm is the matrix and vector multiplication, so the zero elements of the matrix would remain the same in the calculation process, which indicates it more suitable for its implementation onto the GPU. In this section, we introduce our method of mapping the calculation module onto the GPU. The Conjugate Gradient solver includes three main operations: 1) multiplication of the sparse matrix and the vector, 2) addition of two vectors and 3) the sum reduction operation. The kernel step is the multiplication of the sparse matrix and the vector. We move this part of calculation into the fragment processor of the GPU, to utilize GPU fragment processor in its highly efficiently manipulation of the local texture memory on the mathematical calculation. The basic principle is to load matrices and vectors as textures into the GPU, and then rasterize a proper quad of pixels for invoking the fragment program, which process the actual calculation in the fragment processor. The result can be obtained as the color value, transferred directly to the next pass for execution or read back to CPU [6]. Fig. 3 shows the main idea to employ GPU on matrix vector operation, which is the heavies load in deformation computation.

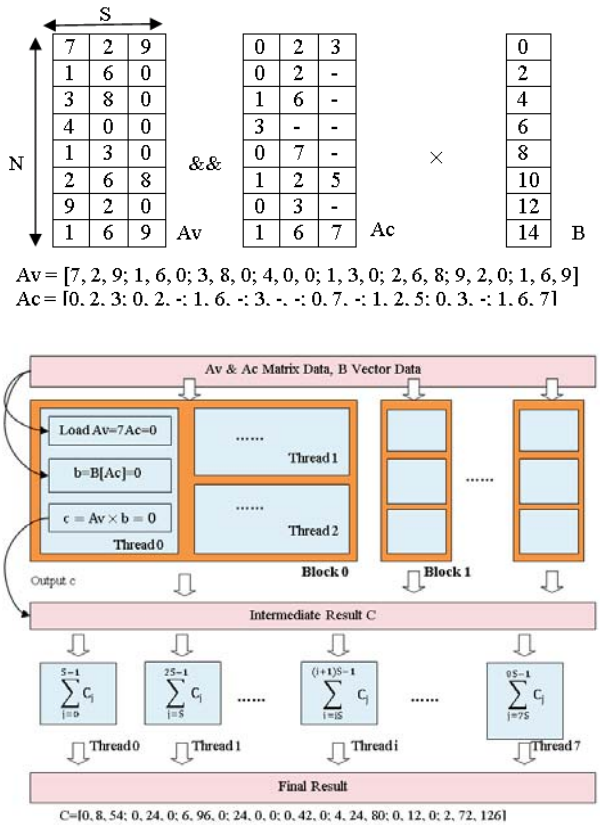


**Fig. 3.** GPU Parallel Computing Architecture

However, the traditional GPU computing only has Single Float data type, while fast Conjugate Gradient solver requires high precision data type for each computational iteration. Therefore, we employ Double Float Precision to achieve it [14].

## 5. Hybrid Structure of Medial Line and Local Deformation

### 5.1 Medial Centerline and Medial Representation

Medial Representation algorithm is a model based on

Centerline. It records information of the centerline atoms only. When there is a deformation on the centerline, we can redraw the object's surface through the centerline information. So Medial Representation is quite appropriate to model soft tissue like blood vessel.

In our model, the centerline is modeled into a Mass-Spring system. That is, each two neighbor atoms on the centerline are linked by a spring, as fig.4 shows. When a force is added on the centerline, it'll deform like any mass-spring system. Then we accordingly alternate the surface mesh points to reconstruct the entire mesh rendering.

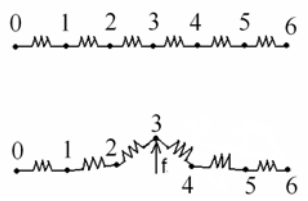


**Fig.4.** Mass-Spring centerline

Due to the efficiency priority to the accuracy in simulation, the original Medial Representation method is simplified as hubs and spokes structure to sacrifice accuracy for speed.

Traditionally, on the basis of Blum's medial axes and from Medial Representation, Prizer proposed the M-Rep method, which is excellent to represent the internal structure and uses medial atoms and a particular tuple $\{x, r, F(\bar{b}, \bar{n}), \theta\}$ to indicate the boundary of the deformable object (as shown in Fig.5).

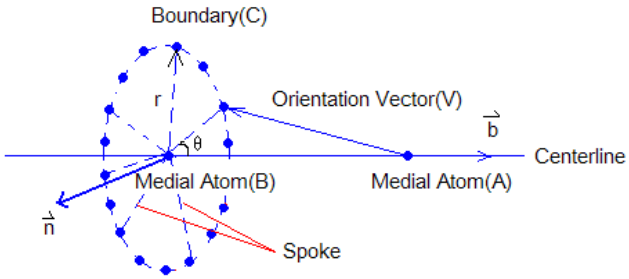


**Fig. 5.** Traditional Medial Representation Structure

In traditional M-Rep method, to draw the whole surface, calculate every boundary node by the following formula:

$$C = x + R_{V,AB}(\theta)(\overrightarrow{AB} \bullet \overrightarrow{r_{BC}})/|AB| \quad (14)$$

Where C and x are the coordinates of boundary C and medial atom B respectively; R denotes the operator to rotate its operand by the argument angle in the plane spanned by V and AB; |AB| means the length of the vector AB. Other spokes connecting with B can be calculated by rotating BC around BA and scale the r length. Iterating the process can obtain all boundaries. So we can get the whole surface.

Here, we modified the implement method of the M-Rep algorithm a little to simplify the model. That is, we don't use the concept of Orientation Vector, but the method as follows. For atom i (i=2,3…n-1) on the centerline, the first boundary node B of this atom can be anyone(usually we choose the one that on the plane xOy of Cartesian coordinates) that satisfies the formula:

$$\begin{cases} \overrightarrow{C_{i-1}C_{i+1}} \bullet \overrightarrow{C_iB} = 0 \\ |C_iB| = r \end{cases} \quad (15)$$

Where Ci means the ith atom on the centerline. That is, CiB must be vertical to vector Ci-1Ci+1, and have the length of r.

Then calculate every boundary node, use the following formula:

$$B' = x + R_{C_{i-1}C_{i+1}}(\theta)B/|r| \quad (16)$$

R denotes the operator to rotate its operand by the argument angle $\theta$ with the axis represented by the vector Ci-1Ci+1, B' and x are the coordinates of boundary and medial atom Ci respectively. The method is showed in Fig.6.

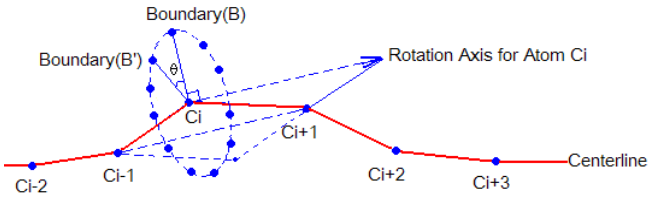


**Fig. 6.** Surface reconstruction method of simplified M-Rep (the red line is the centerline)

### 5.2 Local Region Deformation

Most surgery operations are performed in a small area on a soft-tissue, so we restrict the deformation in a local level if the external force is relatively small in order to reduce the calculation time. In the preprocessing stage of the application, the Dijkstra algorithm is employed to calculate the shortest distance between mass points and a distance table, thus recording the distance between all points is generated. The deformation procedure is the following: if a small force is applied on the white spots (Fig. 7) and the propagation of the force is limited in the second layer. The optimization will be skipped when the external force is larger than the threshold value.

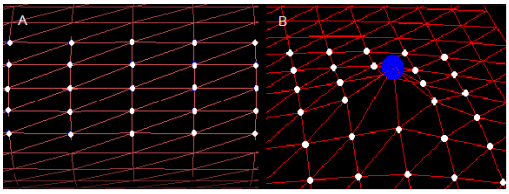


**Fig. 7.** (A) The local deformation region (the big white spot) on an object. (B) The distances between the center white spot and other mass points.

## 6 Experiments

We generate a human kidney mesh using the meshing technique discussed above and simulate its deformation under a boundary force. Fig. 8 shows the results. And observations are showed in Table 1.

We can see in fig. 8 and table 1 that nonlinear strain deformation is closer to the realistic situation, while the linear strain deformation is more distorted.

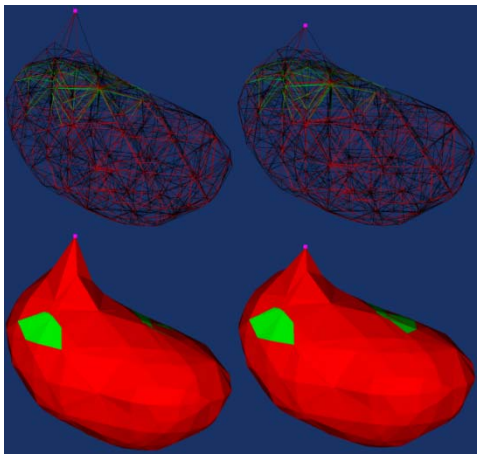

**Fig. 8.** Left is linear strain deformation of human kidney, while right is nonlinear strain deformation. Green meshes represent the original state, while the red ones represent deformation state that is caused by an equal external force at the purple point.

| Item | Mesh state | Quantity |
|---|---|---|
| Volume | Original state | 332889 ($mm^3$) |
| | Linear strain deformation | 364543 ($mm^3$) |
| | Nonlinear strain deformation | 352602 ($mm^3$) |
| Surface Area | Original state | 208017 ($mm^2$) |
| | Linear strain deformation | 211821 ($mm^2$) |
| | Nonlinear strain deformation | 210744 ($mm^2$) |

**Table. 1.** Deformation Statistics (on CPU)

However, solving the system of equations takes most percent of total time, especially for nonlinear strain deformation. In our system, we use Conjugate Gradient algorithm to solve the system, which is suitable to implement onto the GPU [6], as demonstrated in fig. 9 and table 2.

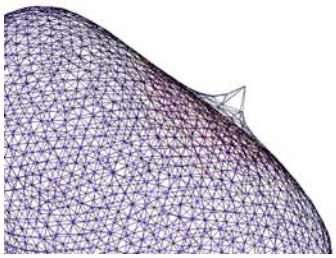

**Fig. 9.** FEM Deformation with Local Region Constrain (on GPU)

| N | 15 | 20 | 25 | 30 |
|---|---|---|---|---|
| Vertex | 1018 | 1922 | 3281 | 5071 |
| Tetra | 3272 | 6538 | 11626 | 18519 |
| Local Vertex | 105 | 187 | 335 | 525 |
| Loading Time (ms) | 63 | 62 | 78 | 109 |
| Matrix | 30542 | 57662 | 98432 | 152132 |
| Error (mm) | 0.1 | 0.1 | 0.1 | 0.1 |
| Iteration | 34 | 44 | 52 | 62 |
| Solve Time (ms) | 219 | 422 | 750 | 1297 |
| Average (ms) | 6.44 | 9.59 | 14.42 | 20.92 |
| Original Volume | 111282 | 118615 | 122672 | 124949 |
| Alternated Volume | 111667 | 118983 | 123166 | 124611 |
| Original Area | 11945.5 | 12449.2 | 12739.8 | 12896.5 |
| Alternated Area | 11974.7 | 12488 | 12799.6 | 12914.6 |

**Table 2.** Deformation Solving Time and Comparison (on GPU)

Last but not least, we apply the hybrid structures of deformable models, by fully calculating the volumetric deformation in the local operation part while only calculating the global deformation by medial representation method. Experiments below show the feasible result of the model, as shown in fig.10 to fig. 12.

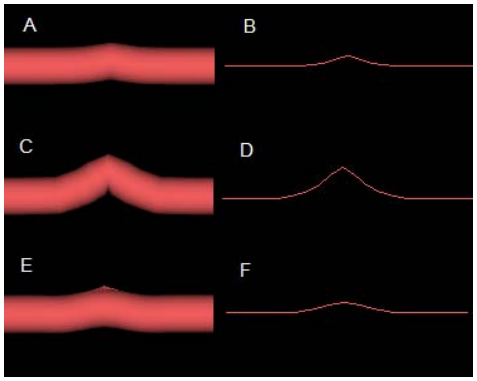


**Fig. 10.** Medial Representation Global Deformation with Local Region Deformation (A,B shows Global Deformation, C,D shows large Global deformation, E,F shows Global Deformation with Local deformation)

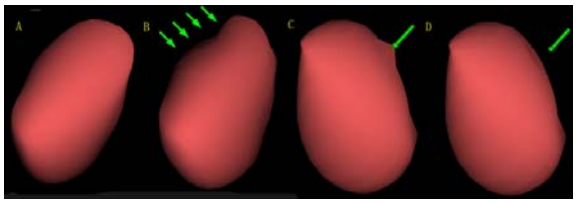

**Fig. 11.** Hybrid Model Deformation with Collision Detection

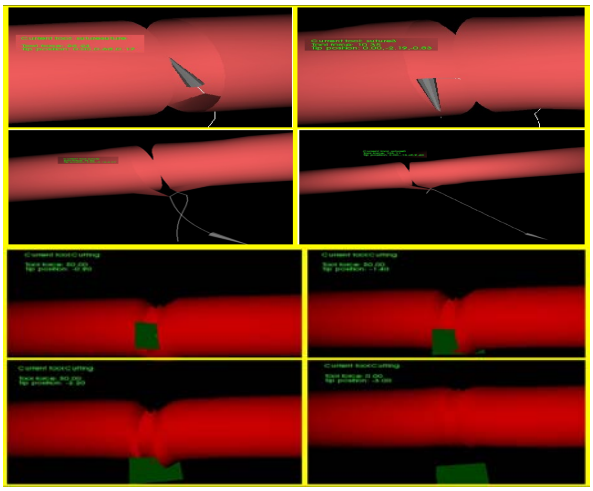

**Fig. 12.** Cutting and Suturing Applications Based on Deformable Models

## 7 Conclusion

In this paper we focus on the simulation for realistic deformation of soft tissue using nonlinear strain FEM, because it's important to simulate large global deformation in virtual surgery. And through experiments, we take the advantage of using nonlinear strain deformation to simulate soft tissue. To achieve real-time performance, we apply Revised Delaunay Meshing, GPU Computing and Hybrid Deformable Structures.

Aiming at a more accurate and robust level of virtual surgery, some other technologies will be also implemented, such as condensation [6] for decreasing the size of system in FEM. The force feedback devices integration and deformation validation problems will also be considered and solved in the future.

## Acknowledgement

The research was partially supported by the Natural Science Foundation of China, Grant No. 70581171, and the Shanghai Municipal Research Fund, Grant No. 045118045. The authors are grateful to Prof. Pizer for sharing his Medial Representation expertise, and Prof. Peters and Prof. Fenster for the valuable feedbacks. We are also grateful to Shanghai Renji Hospital and Shanghai 9th People's Hospital for providing the medical data.

## References

1. Duysak, A. and Zhang, J. J. 2004. Fast Simulation of Deformable Objects. In Proceedings of the information Visualisation, Eighth international Conference on (Iv'04) - Volume 00 (July 14 - 16, 2004). IV. IEEE Computer Society, Washington, DC, pp.422-427.
2. Y. Zhuang. Real-time Simulation of Physically Realistic Global Deformation. Ph. D. thesis of Univ. of California, CA, 2000. The Eurographics Association 2001.
3. Morten Bro-nielsen. Finite Element Modeling in Surgery Simulation, Proceedings of IEEE, March 1998, 86(3): pp. 490-503.
4. Igor Nikitin, Lialia Nikitina, Pavel Frolov, Gernot Goebbels, Martin Göbel, Real-time simulation of elastic objects in Virtual Environments using finite element method and precomputed Green's functions, Eighth Eurographics Workshop on Virtual Environments, pp. 47-52, 2002.
5. Xucheng Wang, "Finite Element Method," Qing Hua University Publishing Company, Beijing, 2003.
6. Wu, W. and Heng, P. A. 2004. A hybrid condensed finite element model with GPU acceleration for interactive 3D soft tissue cutting: Research Articles. Comput. Animat. Virtual Worlds 15, 3-4 (Jul. 2004), pp.219-227.
7. Kardestuncer H, et al. "Finite Element Handbook," McGraw-Hill: New York, c1987.
8. Shewchuk, J. R. 1994 An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. Technical Report. UMI Order Number: CS-94-125., Carnegie Mellon University.
9. D. J. Hebert. Symbolic local refinement of tetrahedral grids. Journal of Symbolic Computation, 11, 1994.
10. Eitz, M.: Realtime Soft Tissue Simulation employing Constraint Based Particle Systems and Hierarchical Spatial Hashing. Master Thesis of Shanghai Jiao Tong University(2006).
11. Huang, P., Gu, L., and Zhang, S.: Real-Time Simulation for Global Deformation of Soft-Tissue Using Deformable Centerline and Medial Representation. ISBMS (2006)67-74.
12. Shaoting Z, Lixu G, Weiming L, Jingsi Z, Feng Q: Real-time virtual surgery simulation employing MM-Model and Adaptive spatial hashing, ICAT(2006).
13 NVIDIA CUDA Programming Guide (v.0.8). NVIDIA corporation: 2007.4
14 Jens Krüger, Rüdiger Westermann. Linear Algebra Operators for GPU Implementation of Numerical Algorithms. SIGGRAPH. 2003.
15 Xiao Yu et al. Novel Tetrahedral Mesh Generation Method based on Delaunay Criteria and Space Disassembling. Submitted to ICAT 2007.