Essential Techniques for Laparoscopic Surgery Simulation

Kun Qian[1], Junxuan Bai [2], Xiaosong Yang [1], Junjun Pan [*2], and Jianjun Zhang[1]

[1]National Center for Computer Animation, Bournemouth University

[2]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University

Abstract

Laparoscopic surgery is a complex minimum invasive operation which requires long learning curve for the new trainees to get adequate experience to become a qualified surgeon. With the development of virtual reality technology, VR based surgery simulation is playing increasingly important role in the surgery training. The simulation of laparoscopic surgery is challenging because it involves large nonlinear soft

[*]e-mail:pan_junjun@hotmail.com

tissue deformation, frequent surgical tool interaction and complex anatomical environment. Current researches mostly focus on very specific topics (such as deformation, collision detection etc.) rather than a consistent and efficient framework. The direct use of the existing methods cannot achieve high visual/haptic quality and a satisfactory refreshing rate at the same time, especially for complex surgery simulation. In this paper, we proposed a set of tailored key technologies for laparoscopic surgery simulation, ranging from the simulation of soft tissues with different properties, the interactions between surgical tools and soft tissues and the rendering of complex anatomical environment. Compared to the current methods, our tailored algorithms aimed at improving the performance from accuracy, stability and efficiency perspectives. We also abstract and design a set of intuitive parameters which can provide developers with high flexibility to develop their own simulators.

Keywords: Laparoscopic surgery simulation, deformation, collision detection, dissection, rendering

1 Introduction

Laparoscopy surgery is a popular minimally invasive operation. It allows the surgeon to access the inside of the human body without having to make large incisions on the skin. However, due to the limitation of manipulation space and viewing angle, there is a higher risk of damaging the internal organs, nerves and major arteries. For new surgeons, performing operations under the supervision of experienced surgeons becomes a practical solution in many scenarios. However, it will inevitably involve a long learning curve for the trainee to gain adequate skills to become a qualified surgeon. With the development of virtual reality technology, training surgeons with a VR based simulator has proven to be effective and can greatly reduce both the risk to patients and the training costs.

Most of the current research on surgical simulation focuses on very specific topics (deformation [1], haptic rendering [2], dissection [3] etc.) rather than the practical framework, especially for larparoscopic surgical simulation. The widely used laparocopic surgery simulators and frameworks in the market such as LapSim and SOFA [4] aim only for training basic skills following strict and relatively simplistic routines. They are not capable of simulating the whole surgery procedure because of the unsolved technique bottlenecks. Getting familiar with the whole surgery procedure is essential not only for developing the surgeons perception in understanding the overall surgical procedure, but also allows them to plan and master the operation tasks in complex operations. There is a high demand from the surgery training market to develop a practical and efficient framework for education oriented surgery

simulation.

In this paper, we are aiming to tackle some of the main challenging issues of laparoscopic surgery simulation, including large deformation of soft tissues, surgical tool interactions, and the rendering of complex materials. We propose an efficient and stable framework integrating a set of specially tailored and designed techniques, ranging from rendering, deformation simulation, collision detection and soft tissue dissection. From a scientific perspective, the main contribution of this paper is the proposal of efficient, stable and fast converging deformation and interaction techniques. They are versatile in simulating complex anatomical soft tissues with different properties. In particular, we are:

- Proposing a set of efficient, fast converging nonlinear large deformation techniques for soft tissue simulation. The biomechanical behaviours can be controlled by the designated parameters.

- Proposing an adaptive spherical collision detection and resolution method, which can improve the convergence rate and alleviate the collision tunnelling artefacts.

- Proposing an interactive haptic dissection approach using implicit shapes.

- Proposing a physically based rendering pipeline to visualize different materials, ranging from soft tissues with muscous layer to metal surgical tools.

2 Related Works

2.1 Deformation Simulation

The simulation of deformable objects has been an active research topic in computer graphics area for decades. A few good surveys [5][6] give a comprehensive overview of this area. For the deformable object simulation, force based dynamics and position based dynamics [7] are two widely used models. Mass spring and finite element are typical force based methods which will suffer from instability especially under explicit integration. Finite element is based on the continuum mechanics which provides an accurate computation model for elastic objects of different material property, but the main disadvantages of finite element method is its high computation cost. Position based dynamics gradually gain its popularity and has been integrated into many commercialized softwares (such as Houdini, Blender etc.) due to its robustness and efficiency. The main feature of PBD is its direct control over vertex positions which eliminates the instability problem in force based method and simplifies the implementation process. Projective dynamics [8] as a more general form of PBD is dependent on the solution of a pre-factorized linear system, which is not efficient when mesh topology changed.

From stability perspective, position based dynamics is the most stable method. Due to the fact that mass spring method and finite element method are force based methods, they suffer from overshoot problem under large time steps or high material stiffness [9]. To solve the stability problem of the force based method, implicit integration method is used which is

computationally expensive because of the computation of the derivate of force with respect to position and velocity. Position based dynamics computes the position changes in each simulation step directly, based on the solution of a quasi-static problem. It provide a stable high level of control even under large time step.

From the efficiency perspective, finite element method is the most inefficient method because it involves expensive global and local matrix operations, such as stiffness assembling, SVD etc. The same problem also exists in mass spring method when assembling the large stiffness matrix and solving large sparse linear system. Position based dynamics is the most efficient method due to the use of the Gauss-Seidel or Jacobian style solver which solve constraint one by one. However, PBD constraint solving style suffers from low convergence rate compared to the global constraint solving method but it can generate visual plausible results under low iteration and deal with topology change more efficiently than global constraint solving method.

From accuracy perspective, finite element method is the most accurate method which can accurately simulate deformable object with different material properties. Mass spring method is based on the hook's law, which can be viewed as a simplified version of finite element method but it cannot capture the volume effect of the solid object. Position based dynamics is comparatively accurate because it is geometrical motivated. However, by integrating continuous material constraint [10], it can also become more physically accurate.

Thus, we choose PBD as the dynamic model for the laparoscopic surgery simulator by balancing the above factors.

2.2 Collision Detection

Collision detection is essential for an interactive real-time application but it could easily become the bottleneck for complex simulation. At coarse level, high efficient spatial data structure, such as bounding volume hierarchies (BVH), spatial hash, distance fields and image based method are proposed to accelerate the broad phase and narrow phase collision detection process. A comprehensive survey has been made in [11][12]. Our method only focuses on the finest level collision detection and resolution so it is compatible with all the above hierarchical spatial data structures and their optimization strategies.

Although many fast polygon intersection test algorithms [13][14] have been proposed, polygon intersection test is still the bottleneck of collision detection especially for complex collision scenarios between deformable objects. To solve this problem, the idea of using computationally efficient and simplified representation for collision primitives becomes popular. Sphere is the most used simplified representation due to its efficiency in overlapping test. For rigid body, generating sphere packing structure based on medial axis [15][16] and inners sphere tree [17] are very popular. However, as the computation of these methods are expensive and mostly occurs in the initial configuration, they are not suitable for dynamic updating of deformable objects. Mendoza [18] proposed a time-critical collision detection algorithm for deformable objects based on a sphere tree constructed using an adaptive medial-axis. Mendoza used detailed mesh for deformation and coarse mesh for sphere tree which is not good for handling detailed collision. Also, this sphere repre-

sentation and similar works [17] does not approximate the original surface well. BD-tree (Bounded Deformation Tree) [19] proposed an output sensitive collision detection for reduced deformable models, but it will suffer from over conservative bound, computational and memory overhead when use high resolution models. Pan [20] proposed a metaball based collision detection method, but it can not provide adaptive approximation when mesh deformed. Algorithms focus on solving sliding collision between tissues in quasi-permanent state have been proposed in [21] [22]. The spherical colliding contact model proposed in [22] is based on the spherical sampling table (SST), which can provide efficient localization of the potentially collided surrounding areas. However, it requires the relative motion between colliding objects is small. In this paper, we proposed a dynamically generated circumsphere structure which is not only computationally efficient for overlapping test but also can approximate the original mesh surface better than existing methods.

2.3 Soft Tissue Dissection

There are two types of dissection procedures for laparoscopic surgery training. One is called electrosurgical dissection. This procedure usually involves a membranous fat tissue which can be simplified as a 2D mesh. The other type is cutting a substantial soft tissue like a polyp, this type is called scalpel dissection. For electrosurgical dissection, tearing and cutting are commonly employed. Müller [7] first introduced tearing of cloth in PBD and Maciel [23] had used this method in the laparoscopic surgery simulation. Pan [24] used

triangle subdivision to simulate surface cutting procedure, but it is only suitable for coarse mesh. In this paper, we propose a surface cutting method suitable for mesh at any resolution using a unified implicit shape.

Volumetric mesh is commonly used in scalpel dissection. To obtain a smooth incision surface, element subdivision is a conventional way. Mor [3] subdivided the colliding tetrahedron into sub-elements at minimal quantity, and it has been employed by recent work [25] [26] for its practicability. Other dissection methods for volumetric mesh can be found in survey [27]. In this paper, both electrosurgical dissection and scalpel dissection are provided.

2.4 Realistic Rendering

The scene inside the patient's abdomen is complex because of the presence of various types of soft tissues, blood vessels and the dynamic glistening of the membrane under the illumination of the head light. Most previous works on the rendering of surgical simulation use the traditional shading models [28] or video image based method [29]. The traditional shading model needs physically plausible albedo, normal, specular and shininess (gloss) maps. The video image based method combined the image mosaicing and view-dependent texture-mapping techniques used the images obtained from video. However, the flexibility of those techniques are limited by the quality of the image. Due to the development of game industry, physically based rendering gradually replaced the traditional shading model and widely used in photo-realistic rendering [30][31]. Physically based rendering (PBR) refers to the

concept of using realistic shading models along with measured surface values to accurately represent real-world materials. Yet, little literature focuses on PBR in surgical simulation. In this paper, we will introduce a PBR based rendering pipeline for surgical simulation.

3 Deformation Simulation for Laparoscopic Surgery

The direct use of PBD and existing types of constraints is not efficient, flexible enough to simulate laparoscopic surgery because it involves many types of tissue with different properties. The soft tissues involved in a laparoscopic surgery can be categorized into three types: volumetric tissue (such as the human organs), tubiform and glandular soft tissue (blood vessels, nerves and glands), fascia soft tissue (such as mesentery). In the following sections, we will introduce how each type of soft tissues is simulated.

3.1 Volumetric Soft Tissue Simulation

Most of the organs inside the abdomen will keep the shape during the surgery procedure. When a surgical tool contacts them, the deformation will only happen in the local contact area. The deformation algorithm needs to keep the global shape as well as the local deformation. If using volumetric mesh filled with tetrahedral elements, it will cost more computation and memory resources. If using a surface mesh, shape matching can preserve the shape but it cannot handle local deformation well because it provides a global transformation approximation. The core idea of shape matching is to find the optimal least square

best-fit transformation from the rest state ($\mathbf{x}_i^0$) to the current configuration ($\mathbf{x}_i$) which equals to minimize $\sum_{i=1}^{n} w_i(\mathbf{A}(\mathbf{x}_i^0 - \mathbf{x}_{cm}^0) - (\mathbf{x}_i - \mathbf{x}_{cm}))^2$, where n is the number of vertices of the mesh, $\mathbf{x}_{cm}^0$ and $\mathbf{x}_{cm}$ are the centre of mass of rest state and deformed state respectively, w_i are weights of individual vertices. The global transformation matrix $\mathbf{A}$ can be calculated as:

$$\mathbf{A} = \sum_{i}^{n} m_i(\mathbf{x}_i - \mathbf{x}_{cm})(\mathbf{x}_i^0 - \mathbf{x}_{cm}^0)^T \tag{1}$$

The rotation matrix $\mathbf{R}$ can be extracted from $\mathbf{A}$ using polar decomposition $\mathbf{A} = \mathbf{RS}$ [32]. The goal position $\mathbf{g}_i$ can be calculated using $\mathbf{R}$ for each vertex i.

$$\mathbf{g}_i = \mathbf{R}(\mathbf{x}_i^0 - \mathbf{x}_{cm}^0) + \mathbf{x}_{cm} \tag{2}$$

Then each vertex is dragged towards its goal position to recover to the original shape. To achieve the local and non-linear deformation, cluster could be used to discretize the shape matching area. However, too many clusters will greatly influence the overall performance. On the other hand, less clusters cannot handle detailed local deformation well (such as collision with small object) because the optimal transformation matrix $\mathbf{A}$ is a global rigid transformation matrix. When collision happened, obvious oscillation or collision tunneling will happen. The oscillation is caused by the repetitive violation of the collision constraint (unilateral constraint, which is solved only when it is violated) by shape matching. The collision tunneling is caused by the tendency of recovering to goal position. The goal position may be far from current position so that the recovery step may make the vertex run through thin object, resulting in collision tunneling.

In this paper, we solve this problem by dynamically limiting the tendency of going back to the goal position. For a vertex $\mathbf{x}_i$ and its predicted position $\mathbf{p}_i$, we interpolate its goal position $\mathbf{g}_i$ and $\mathbf{p}_i$ by a dynamic recovering factor $\eta_i \in [0, 1]$, $\mathbf{g}_i = \mathbf{g}_i + \eta_i(\mathbf{p}_i - \mathbf{g}_i)$. When $\eta_i = 0$, $\mathbf{x}_i$ will directly move to the goal position. When collision happened we set $\eta_i = 1$, then $\mathbf{g}_i = \mathbf{p}_i$ which means $\mathbf{p}_i$ will not move immediately. After that it will be dragged to original shape gradually by recovering η_i at the user defined rate R_r, $\eta_i = \eta_i - R_r$ (See Figure 1). During the process, both η_i and $(\mathbf{p}_i - \mathbf{g}_i)$ will decrease with the evolving of time which means the speed of recovering to original shape is decreasing with time (see Algorithm 1). This can well reflect the property of highly damped soft tissue and its shape recovery procedure. The value of η_i can be determined by the further evaluation process performed by surgeons.

3.2 Fascia Simulation

The property of most fascia is not totally inextensible. The extensibility of fascia varied according to the way it connected to surrounding tissues. Low convergence rate of PBD will result in the stretchy looking of the fascia, making it like a piece of cloth. Long range attachment (LRA) [33] constraint is used to deal with the inextensibility of cloth in a fast converged manner. The core idea of LRA constraint is confining each unconstrained vertex within the initial distance of the attached vertices. The LRA method can be divided into two steps: Firstly, calculate an initial distance $d_i^{LRA'}$ between unconstrained vertex ($\mathbf{x}_i$)

and attached vertex ($\mathbf{x}_i^a$). Secondly. if the unconstrained vertex is out of range of the initial distance, project it back to the range of the sphere centered at the attached vertex with radius of $d_i^{LRA'}$.

For each unconstrained vertex, it may be constrained by different attached vertices. The influence of each attached vertex to the unconstrained vertex can be averaged using Jacobian style. However, the fascia does not like inextensible cloth. If directly using LRA in fascia simulation, obvious artefact and oscillation will appear when surgical tool drag the fascia. This is because the LRA constraint is too hard which conflicts with the drag constraint. In fact, the portion of fascia which is far from attached vertex is more stretching than the part near attached vertex, see Figure 3. In order to take the advantage of the fast convergence rate of LRA, we make relaxation to the LRA constraint, making each vertex's tendency of stretching evolves with the distance to attach point ($\mathbf{x}_i^a$) (See Figure 2):

$$d_i^{LRA} = d_i^{LRA'} + \beta max((\|\mathbf{x}_i^a - \mathbf{x}_i\| - d_i^{LRA'}), 0) \tag{3}$$

where β is a user-defined parameter controlling the level of relaxation, which influences the stiffness of the fascia. d_i^{LRA} is the new LRA constraint distance between $\mathbf{x}_i$ and $\mathbf{x}_i^a$. Such relaxation turns the hard LRA constraint into a soft constraint which can benefit the stability and convergence rate.

3.3 Tubiform and Glandular Soft Tissue Simulation

Compared to the organs and fascia, the nonlinearity property of tubiform and glandular soft tissue are more obvious because the internal of these tissue are not completely solid, which is usually composed of loose packed fat, protein and fibres. Thus, tetrahedral discretization is not suitable for such structure although it is widely used for the simulation of continuum object. Also, frequent expensive inversion handling and memory inefficiency makes tetrahedral discretization not suitable for such tissue simulation. Simply use or combine existing constraint, such as bending, stretching, pressure, volume preservation etc., is hard to effciently achieve such complex nonlinear effect (see Figure 4). In this paper, due to the tubiform and thin shape of this kind of tissue, we inspired by the truss structure[34] from structural engineering (see Figure 4), which is a triangle based support structure. We fill the tubiform and glandular soft tissue with elastic truss with different stiffness as a simplified approximation to the internal of these non-solid soft tissues, achieving high efficiency and high flexibility.

Such designing cannot only preserve the shape of the soft tissue to certain extent but also provide high movement flexibility by strut with different stiffness. In the pre-processing stage, we generate the truss structure as described in Algorithm 2: Firstly, localize potential strut pair. For vertex $\mathbf{x}_i$ with normal $\mathbf{n}_i$, we find the nearest face intersected by $-\mathbf{n}_i$. Using the vector determined by the face center and $\mathbf{x}_i$ to generate a cone with the user-define angle θ. Marking the vertices inside the cone as the potential strut pair (Line 2-9 in algorithm 2,

also see Figure 5). Secondly, from the strut candidates, if cosine of the angle between the potential strut vertex's normal and $\mathbf{n}_i$ satisfies the user-defined bending threshold ($\theta_{threshold}$), then connect them as a strut. The value of $\theta_{threshold}$ determines the bending feature of the object. During this process, the maximum and minimum strut length was used for stiffness coefficient calculation (Line 11-18 in algorithm 2). Thirdly, define the shortest strut as the most stiff one. Based on a user-defined minimum stiffness factor k_{min}, we make an interpolation between the shortest and longest struts to determine the stiffness factor for each strut (Line 19-22 in algorithm 2). For the detail of the algorithm, please refer to Algorithm 2.

4 Collision Detection and Resolution

In this part, we present an adaptive implicit circumsphere collision detection and resolution method based on the local geometry features and material properties, which is a much faster alternative for polygon collision tests. Under the PBD framework, we demonstrate that our method can effectively accelerate the convergence rate without causing noticeable visual artefacts. What's more, although our method cannot eliminate collision tunnelling completely, our method can actually provide better prevention to collision tunnelling without applying any continuous collision detection because the structure of circumsphere actually provide a well approximated safety thickness for the thin surface, which is different from the bad approximated safety thickness provided by traditional methods (such as collision

margin, sweep volume etc.).

4.1 Overview

In the broad phase, we use the spatial hash structure to localize the potentially colliding pairs due to its convenience to handle self-collision. Our method concentrates on the subsequent narrow phase where we separate it into two sub-phases, bounding sphere phase and circumsphere phase. For the bounding sphere phase, potentially colliding pairs are checked by a simple bounding sphere test of the triangle primitives. When the likelihood of collision is established, it moves on to the circumsphere phase, in which we test the overlap of the dynamically generated circumsphere for the underlying primitives and use the circumsphere for the succeeding collision resolution.

4.2 Adaptive Circumsphere Generation

The circumsphere is generated based on a shift of the circumcenter center of the triangle along the negative direction of the surface normal. How far to shift the circumscenter will determine how well the circumsphere approximates the surface. If we use uniform scale to shift the circumpshere, it will generate bad initial approximation because the size of each triangle primitive may be different, as can be seen in Figure 6. To create a better initial approximation, we generate the initial circumsphere according to the primitive size:

$$\mathbf{x} = \mathbf{x}_c - \mathbf{N}_c \varphi R_b cot\theta \tag{4}$$

where $\mathbf{x}_c$ is the circumcenter of the corresponding triangle primitive, $\mathbf{N}_c$ is the normalized surface normal at $\mathbf{x}_c$ and φ is a global scale factor for translating $\mathbf{x}$ along $-\mathbf{N}_c$, R_b is the bounding sphere's radius, θ is a fixed angle called safe angle.

4.3 Local Feature Based Circumsphere

However, such circumsphere approximation will become more and more inaccurate when the mesh is deformed. To improve the approximation accuracy, we take the surface curvature as a factor to influence the position of the circumsphere. To cater for the need of real-time response, we adopt a computationally efficient dual mesh [35] based structure to approximate the curvature of the original triangle mesh surface. We connect the geometric centre of the neighbour faces to construct a dual mesh structure and use the curvature of the dual mesh's vertices to approximate the curvature of its corresponding original surface. However, the dual mesh structure is composed of polygons rather than triangles. It is difficult to approximate the curvature at dual mesh's vertex because the curvature calculation method such as Gauss-Bonnet requires the computation of surrounding polygon area. To make the area calculation easier, we make the following modification to the dual mesh structure. Inside each polygon of dual mesh, there must be one vertex of the original mesh. We connect this original mesh's vertex with its surrounding dual mesh's vertices, decomposing the polygon dual mesh into triangles, as can be seen in Figure 7.

Then we can use the Gauss-Bonnet scheme to approximate the local curvature of the

dual mesh vertex $\mathbf{x}_i^{dual}$[36]. To integrate the influence of curvature into the position of circumsphere, we need to apply a factor (we use $H_3(K)$) associating with curvature to influence the shifting distance ($-\mathbf{N}_c\varphi R_b cot\theta$) in Equation 4. The range of curvature can be $(-\infty, +\infty)$. When the mesh surface is a plane (zero curvature), we want curvature factor has no influence on the circumsphere adjustment (circumsphere shift), which can also interpreted as we use the initial circumsphere for plane area. For the very large curvature area, we want the circumsphere to shrink to bounding spheres which means the curvature factor should cancel the influence of circumsphere shift. Although the bounding sphere provides bad approximation to the surface, it actually provides a safety range and better tolerance to collision tunneling for the large curvature area. Also, it can reduce the unnecessary collision detection using this shrinked circumsphere.

To achieve this, we use cubic Hermite interpolation $H_3(K)$ as the curvature factor to interpolate the curvature into the range of [0,1] (see Equation 5 and Figure 8). When $K = 0$, we use the initial circumsphere to approximate the mesh surface ($H_3(K)$=1). When $K \rightarrow \infty$, we let the circumsphere shrink to the bounding sphere of the primitive ($H_3(K) = 0$). The cubic Hermite can be written as:

$$H_3(x) = y_0 h_0(x) + y_1 h_1(x) + y_0' H_0(x) + y_1' H_1(x) \tag{5}$$

, where the constraint point of the interpolation curve are $(x_0, y_0) = (0, 1), (x_1, y_1) = (\infty, 0)$ and the corresponding tangent are $y_0'(x_0) = 0, y_1'(x_1) = 0$. $h_0(x), h_1(x), H_0(x), H_1(x)$ are the Hermite basis functions. This interpolation mapping function can be calculated at the

initial configuration and used directly afterwards. After applying the curvature factor, the equation for circumsphere's center becomes:

$$\mathbf{C}_{cir} = \mathbf{x}_c - \mathbf{N}_c \varphi R_b cot\theta H_3(K) \tag{6}$$

4.4 Material Property Based Circumsphere

For a deformable object, the surface geometry feature changes all the time especially when collision occurs. When the collided area is subject to large deformation, the circumsphere's size should be larger for the collided area to approximate the deformed surface better and provide more accurate collision response. Thus a physical quantity which used to describe the material property is needed. The deformation can be measured by the amount of energy stored inside the deformed area. The energy accumulated in the deformed object is determined by the deformation gradient which provides a good reflection of how much the object deformed. The deformation gradient can be defined as

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} \tag{7}$$

where $\mathbf{x}$ and $\mathbf{X}$ are the coordinate of vertex in deformed state and rest state respectively. How to calculate the deformation gradient for triangle primitive can be found in [37]. How to define the form of energy is dependent on the constitutive model used, such as St.Venant-Kirchoff model, Neohooken model, corotated model etc. Due to the corotated model and Neohookean model both need the expensive computation of SVD. Here we used the St.

Venant-Kirchoff model to define the energy:

$$\mathbf{E} = \mu\|\varepsilon\|^2 + \frac{\lambda}{2}tr^2(\varepsilon) \tag{8}$$

Where ε is the strain $\varepsilon = \frac{1}{2}(\mathbf{F}^T\mathbf{F} - \mathbf{I})$, λ and μ are $l\acute{a}me$ cofficient. The energy reflects the tendency of recovering to initial state so that we used a scalar energy factor $\epsilon = \mathbf{E}$ to dynamically adjust the size of circumsphere according to the material property. Theoretically, the energy can be infinite large or small. We add is a scalar which can reflect the change of energy rather than the accurate energy. So we applied a user defined scalar factor ς to scale the influence of the energy factor on the position of circumsphere (see Figure 9).

$$\mathbf{C}_{cir} = \mathbf{x}_c - \mathbf{N}_c(\varphi R_b cot\theta H_3(K) + \varsigma\epsilon) \tag{9}$$

4.5 Updating of Circumsphere

Instead of updating each frame, we update the circumsphere structure only when the shape change of its underlying primitive exceeds a threshold δ compared to the last frame. The determinant of deformation gradient reveals the shape change. We donate $h_{shape} = |\mathbf{F}|$ where $\mathbf{F}$ is the deformation gradient. However, for a triangle mesh, $\mathbf{F} \in R^{3\times2}$ is not a square matrix. Due to the use of St. Venant Kirchhoff model, we reflect the shape change using $h_{shape} = |\mathbf{F}^T\mathbf{F}|$. If $\|h_{shape} - 1\| < \delta$ where δ is a user defined threshold, it means the shape change ratio is less than a certain threshold then we just update the centre of corresponding circumsphere using the same curvature factor and material factor in last frame. Otherwise, when $\|h_{shape} - 1\| \geq \delta$, we update both the curvature factor and energy factor. There is no

denying that such updating method will affect accuracy but it does not have obvious influence on the visual plausibility. The tradeoff is the improvement of real-time performance. The analysis of the choosing of δ is made in section 4.7.

4.6 Collision Resolution

When collision happened, we bounced the collided circumsphere pair and its underlying triangle primitive away. Firstly, we find the state which collision just happened. Secondly, update the velocity for each sphere using the principle of linear impulse and momentum. Finally, along the direction of new velocity, move each sphere to final position. Then update the position of sphere's underlying triangle primitive using the translation of sphere.

4.7 Method Analysis

We compare our method with polygon based method [13] and bounding sphere based method [18] from accuracy, stability and efficiency perspectives. In order to compare with ground truth, we devised a special experiment of collision detection between a rigid cylinder and a deformable soft tissue as can be seen in Figure 10. Thus the exact distance for collided part and the main axis of rigid cylinder should be the cylinder's radius. This exact distance d_{exa} will be compared in the following experiments.

For accuracy comparison, we use the average distance d_{ave} from collided vertices to the main axis of the cylinder to measure accuracy. In Figure 11, we show the record of d_{ave}

for each method during the simulation and compare them with d_{exa}. When the iteration is low, although the polygon based method is the most accurate, such accuracy is achieved at the cost of introducing artefact because some part of the curve fall below the exact result. This means the collided primitive has run into collision object. Although such artefacts can be avoided by using collision margin, safety bound or sweep volume, they provide bad approximation compared to our method, as can be seen in Figure 12. Bounding sphere method is the most inaccurate method because its poor approximation of the mesh surface.

The fluctuation of the vertex position reflects stability. We therefore measure the average position change for all the collided vertices in each frame to evaluate the stability. From Figure 13, the polygon and bounding sphere based methods are subject to dramatic fluctuation when the number of iteration is small. For the polygon based method, the high variability of polygon shape is the main cause of the instability. The high variability of polygon shape could easily increase the chance to break the already solved constraint. For the bounding sphere based method, the instability is caused by its poor approximation of mesh surface. Our circumphere structure, on the contrary, can not only provides good approximation of the mesh surface, but also adds fewer unstable factors to the constraint solving process due to its uniform shape, making our method the best performer especially when the number of iteration is low.

Time consumed by collision detection and resolution in each frame is a good measurement of algorithm efficiency. As can be seen in Figure 14, the bounding sphere method is the most efficient method because it only needs to update the centre, radius and perform overlap

test when collision happens. Our method needs re-calculation of circumsphere when the shape change of underlying primitive exceeds a given threshold. The polygon intersection based method is the most time consuming one because polygon intersection test should be performed for each potentially collided pair in each frame.

As to the update threshold δ proposed in Section 4.5, we test accuracy, stability and efficiency for different δ. As can be seen in Figure 15, the accuracy measured by the relative error of d_{ave}, stability measured by the variance of d_{ave}. When $\delta = 0$, circumsphere will be updated in each frame which results in instability and inefficiency. The cause of instability is that the ever changing size of circumsphere constantly add unstable factor to the constraint solving process. And the cause of inefficiency is the re-calculation of circumsphere in each frame. Relative error of accuracy evolves with the increasing of δ because the approximation becomes inaccurate. The larger δ is, the less time is needed for circumsphere updating. The extreme situation is no update for circumsphere. Then the circumsphere's approximation of deformed mesh is the worst which can result in instability. As shown in Figure 15, when δ is less than around 70%, stability improves with the increase of δ. For different types of simulation purposes, different updating rate should be utilized.

5 Soft Tissue Dissection with Implicit Shape

There are two types of dissection procedures in laparoscopic surgery: electrosurgical dissection and scalpel dissection. Electrosurgical dissection mainly used to split thin soft tissue

like membrane (surface mesh). Scalpel dissection is used to cut solid soft tissue (volumetric mesh). In the field of computer graphics, tearing and cutting are two major methods for dissection simulation. Tearing splits the originally connected vertices apart. Cutting changes the topology of the original mesh by a cutting geometry. In this section, we proposed a generalized dissection method based on implicit shape.

5.1 Implicit Shape for Surgical Instrument

Rather than a simple cutting plane, the real surgical instruments used for dissection have various shapes, such as column, sphere, polyhedron etc. Considering the applicability of implicit shape, we employ a combination of implicit shapes to represent the dissection part of the surgical instrument. Then dissection simulation with different surgical instruments can be unified into a general framework. During dissection, we move the surgical instrument and generate a sweep volume by connecting the implicit shapes at different times. The surface mesh or the volumetric mesh is dissected by the sweep volume. The details of the dissection method will be introduced in the following sections.

5.2 Surface Mesh Dissection

For the surface mesh dissection, we integrate both cutting and tearing. We employ implicit shape to generalize the cutting procedure. The surface mesh is composed of triangle faces. The algorithm is summarized in Algorithm 3. First, we generate the sweep volume V_{sweep}.

Then we compute intersections and remove all the faces inside V_{sweep} (line 3-5). For each intersected face f_j, we compute the area A_j and subdivide it into a set of small triangles $\{f_{j_k}\}$. From line 10 -15, we remove f_{j_k} if it is inside V_{sweep} or it does not satisfy a user defined threshold $\upsilon_{threshold}$ (see Algorithm 3). Figure 16 illustrates the process of surface cutting. In Figure 16, the purple triangles are the primitives intersected with sweep volume. After cutting, the green triangles which are contained in the sweep volume are deleted, also the yellow primitives which are too small are deleted.

In real surgery environment, the most commonly used shape for surgical instruments in surface dissection is column. A sweep surface is generated by connecting the shapes at different times. The intersections on the sweep surface are intuitive. There may be more than one intersections on the shape's boundary. If there are more than one boundary intersections and no sweep surface intersection, we combine these intersections into a median point to represent the intersection on the boundary as Figure 17 shows. Then we subdivide the intersected triangle. Compared with previous works, our method is suitable for mesh at any resolution. For the tearing of the surface, we utilize the method proposed by Müller [7].

5.3 Volumetric Mesh Dissection

We use volumetric mesh dissection to exsect a polyp on the tissue surface. The volumetric mesh is composed of tetrahedral elements. When dissecting the volumetric mesh, line is selected as the implicit shape for surgical instrument, and the sweep volume turns into a

sweep surface. The dissection algorithm for volumetric mesh is similar to surface cutting: finding the intersections and subdividing the original primitives. There are two types of intersections between the tetrahedral mesh and sweep surface: edge intersections and face intersection. We encode the edges and faces of tetrahedron in binary, and the intersecting state can be represented by a tuple of edge code and face code when there is an intersection. Then the intersected tetrahedra are split into small elements according to the intersecting state. We employed the method in [3] to subdivide the tetrahedron. Figure 18 shows the dissection of fascia and volumetric mesh in the simulator.

6 Realistic Rendering

There are several models of surface BRDFs. The empirical model (Blinn, Phong etc.) and microfacet model are the most widely used. The empirical model is a fast computational model adjustable by parameters, but without considering the physics behind it. The microfacet model is inspired by real physical process. Due to the complex anatomical structures and their glistening effect under light, our material rendering is based on the microfacet model.

6.1 Material Model

The microfacet model can be decomposed into the diffuse part (f_d) and specular part (f_r). This model can be described in the general Equation 10, each component and parameter can

be modified according to practical needs:

$$f_{d/r} = \frac{\int_\Omega f_m(\mathbf{v},\mathbf{l},\mathbf{h},\mathbf{S})G(\mathbf{v},\mathbf{l},\mathbf{S})D(\mathbf{S},\alpha)(\mathbf{v}\cdot\mathbf{S})(\mathbf{l}\cdot\mathbf{S})d\mathbf{S}}{(\mathbf{n}\cdot\mathbf{v})(\mathbf{n}\cdot\mathbf{l})} \tag{10}$$

n is the surface normal , **l** is the lighting direction, **v** is the viewing direction, **h** is the half vector, the integration $\int_\Omega d\mathbf{S}$ indicates an integral over a hemisphere of all directions, $D(\mathbf{S},\alpha)$ is the normal distribution (NDF) function, $G(\mathbf{v},\mathbf{l},\mathbf{S})$ is the geometry attenuation term and $f_m(\mathbf{v},\mathbf{l},\mathbf{h},\mathbf{S})$ is the Fresnel term. The difference between f_d and f_r is dependent on the Fresnel term f_m. For the specular term f_r, it can be written as the general Cook-Torrance form:

$$f_r(\mathbf{v},\mathbf{l},\mathbf{h},\alpha) = \frac{F(\mathbf{v},\mathbf{h})G(\mathbf{v},\mathbf{l},\mathbf{h},\alpha)D(\mathbf{h},\alpha)}{4(\mathbf{n}\cdot\mathbf{v})(\mathbf{n}\cdot\mathbf{l})} \tag{11}$$

where $F(\mathbf{v},\mathbf{h})$ is the Fresnel term for specular light. The term D determines the appearance of surfaces. To capture the real world better, we use the GGX distribution [38] which can produce realistic "long tailed" effect.

$$D(\mathbf{h},\alpha) = \frac{\alpha^2}{\pi((\mathbf{n}\cdot\mathbf{h})^2(\alpha^2-1)+1)^2} \tag{12}$$

Where $\alpha = roughness^2$. The term G is used for describing how much the microfacet is occluded by others. Heitz et al.[39] point out that approximating Smith visibility function using Schlick model [40] and Disney's modification [38] are not accurate enough. The heigh-correlated Smith function which can model the correlation between the masking and shadowing according to the height of the microfacet is a good choice.

$$G(\mathbf{v},\mathbf{l},\mathbf{h},\alpha) = \frac{\chi^+(\mathbf{v}\cdot\mathbf{h})\chi^+(\mathbf{l}\cdot\mathbf{h})}{\Lambda(\mathbf{v})+\Lambda(\mathbf{l})} \tag{13}$$

Where $\Lambda(\mathbf{x}) = \frac{-1+\sqrt{1+\alpha^2 \tan^2(\theta_x)}}{2}$, θ_x is the angle between normal and $\mathbf{x}$, $\chi^+(a)$ is the Heaviside function: return 1 if $a > 0$, return 0 if $a \leq 0$.

The typical choice for the Fresnel term F is Schlick's approximation [40]. We utilize a spherical Gaussian approximation method proposed in [41] which is more computationally efficient for hardware. We adjust the term F as:

$$F(\mathbf{v}, \mathbf{h}) = F_0 + (1 - F_0) \times 2^{-6.745372(\mathbf{v}\cdot\mathbf{h})^2 + 35.324156(\mathbf{v}\cdot\mathbf{h})} \tag{14}$$

For the diffuse term f_d, we take the form of Disney's empirical method which takes the roughness of material into account and generates retro-reflection at grazing angles. This model can approximate the main features of the MERL database's material.

$$\begin{aligned} &f_d(\mathbf{v}, \mathbf{l}, \mathbf{h}, \alpha) = \\ &\frac{(1 + (F_{D90} - 1)(1 - (\mathbf{n} \cdot \mathbf{v}))^5)(1 + (F_{D90} - 1)(1 - (\mathbf{n} \cdot \mathbf{l}))^5)}{\pi/\rho} \end{aligned} \tag{15}$$

Where $F_{D90} = 0.5 + \cos(\mathbf{l} \cdot \mathbf{h})^2 \alpha$. In real world, the energy received by material is no less than the reflected energy. The energy conservation can cope with the effect at grazing angle where the light is tend to be scattered more. To keep the computation efficiency, we adopt the Disney method which multiply the Fresnel reflectance $(1 - F(\mathbf{n}, \mathbf{l}))(1 - F(\mathbf{l}, \mathbf{v}))$ to Equation 15.

6.2 Lighting Model

Under the effect of light, some soft tissues will reflect the surrounding environment. Image based lighting (IBL) allows us to represent the incident light surrounding a point by an

environment map, making the object fit into the environment without expensive lighting computation. To achieve this, incident light should be in consistent with BRDF equation especially for layered material. However, the computation of the interaction between IBL (L) and BRDF material (f) is a costly operation which needs to integrate all the direction of the light:

$$L(\mathbf{v}) = \int_{\Omega} f(\mathbf{l}, \mathbf{v}) L(\mathbf{l}) d\mathbf{l} \tag{16}$$

To improve efficiency, pre-integration can be used for IBL computation. For the specular lighting computation, the Equation 16 can be approximated using the method proposed in [42]:

$$L_r(\mathbf{v}) \approx \sum_{i}^{N} (F(\mathbf{v}, \mathbf{h}) G(\mathbf{l}, \mathbf{v}, \mathbf{h}, \alpha)(\mathbf{v} \cdot \mathbf{h})) \sum_{i}^{N} (L(\mathbf{l})(\mathbf{n} \cdot \mathbf{l})) \tag{17}$$

Where N is the number of light probes. We can see that $L_r(\mathbf{v})$ is approximated by two summation term, called DFG term and LD term respectively,both can be precomputed separately. Similarly, for diffuse material, it can be calculated as:

$$L_d(\mathbf{v}) \approx \sum_{i}^{N} \frac{f_d(\mathbf{v}, \mathbf{h}, \mathbf{l}, \alpha)}{\pi} \sum_{i}^{N} (L(\mathbf{l})(\mathbf{n} \cdot \mathbf{l})) \tag{18}$$

$\sum_{i}^{N} (L(\mathbf{l})(\mathbf{n} \cdot \mathbf{l}))$ can be efficiently calculated using spherical harmonic method [43]. Spherical harmonics uses a frequency space to represent an image over a sphere, which is continuous and rotationally invariant. The usage of this representation can produce accurate diffuse

reflection from a surface using nine spherical harmonic basis functions:

$$c_1 L_{22}(x^2 - y^2) + c_3 L_{20} z^2 + c_4 L_{20} - c_5 L20 + \\ 2c_1(L_{2-2}xy + L_{21}xz + L_{2-1}yz) + 2c_2(L_{11}x + L_{1-1}y + L_{10}z)$$

where L_i are the nine lighting coefficients and coefficient c_i are constant value which can be found in [43]. Figure 19 is the demonstration of objects using different material with different roughness.

6.3 Rendering Pipeline

For the rendering component of a game engine, the most time consuming part is the post-processing stage, such as SSAO, depth of field, HDR, soft shadow and motion blur etc. Those post-processing techniques have one thing in common: expensive pixel operation. Excessive pixel sampling operation may cause the sampler stall phenomenon which means that GPU is too busy to handle all the sampling instruction. Considering the practical scene and lighting complexity of laparoscopic surgery, our reduced deferred shading based rendering pipeline focuses on the lighting and material rendering. Our pipeline will not consume much computational resource but provide high quality rendering effect for laparoscopic surgery. Figure 20 shows an example of the rendering procedure and the structure of the whole rendering pipeline.

7 Implementation and Results

Implementing the above discussed techniques, we have built a rectum cancer surgery simulator using C++, OpenGL and OpenHaptics which runs on a desktop with a Xeon 5 CPU and Quadro K2000 GPU. We use two Omni Phantoms haptic devices running at the rate of 1KHz, as can be seen in Figure 21. To determine the optimal values of the control parameters of the physical simulation, we have tested our system with the help of experienced practitioners. Our aim was to achieve the same or very similar effect in the simulation of the operations. In table 1, parameters used for the simulator are listed.

To evaluate the computational cost, we have designed an experiment to include all the key operations for the rectum surgery. We recorded the time cost while performing the surgical operation (see Figure 21). We analyzed the cost distribution of each component such as graphical rendering, collision detection, deformation and haptic rendering. In detail, we also measured the cost of major deformation algorithms such as relaxed LRA, truss and goal limiting shape matching. The computational cost for deformation is dependent on the current surgical operation and operative site. In the initial stage of the simulation (frame 1 to 500), the main interaction is between the large volumetric organ and surgical tools so that the goal limiting shape matching is the major deformation algorithm. From frame 500 to 1500, the operative site focused more on the separation of tubiform and glandular soft tissues. The truss based deformation algorithm begins to play its role at this stage. From frame 1500 to 2146, mobilizing bilateral sides of the colon from its connected fascia is the main operation,

which is mainly simulated by relaxed LRA. Successive over-relaxation technique [44] has been used to average each kind of deformation constraint in a soft manner. Such over-relaxation propagates constraint corrections faster which will benefit convergence speed.

8 Future Works

To further improve our system, there are still a few aspects needed to be developed in the future. Currently, our constraint solving procedure consumes most of the computation resources. Migrating this procedure onto GPU using parallel computing could further improve the performance of the system. Yet, this system does not support smoke and fluid simulation which can further enhance the visual effects making the simulation more immersive and realistic.

9 Acknowledgement

This work is partially supported by National Natural Science Foundation of China (61402025) and Centre for Digital Entertainment which is an EPSRC funded centre for doctoral training. We thank Prof.Ladislav Kavan and Tiantian Liu for the helpful discussions.

References

[1] Gábor Székely, Ch Brechbühler, R Hutter, Alex Rhomberg, Nicholas Ironmonger, and P Schmid. Modelling of soft tissue deformation for laparoscopic surgery simulation. *Medical Image Analysis*, 4(1):57–66, 2000.

[2] Timothy Coles, Dwight Meglan, and Nigel W John. The role of haptics in medical training simulators: a survey of the state of the art. *Haptics, IEEE Transactions on*, 4(1):51–66, 2011.

[3] AndrewB. Mor and Takeo Kanade. In *Medical Image Computing and Computer-Assisted Intervention C MICCAI 2000*, volume 1935, pages 598–607. Springer Berlin Heidelberg, 2000.

[4] Jérémie Allard, Stéphane Cotin, François Faure, Pierre-Jean Bensoussan, François Poyer, Christian Duriez, Hervé Delingette, and Laurent Grisoni. Sofa-an open source framework for medical simulation. In *MMVR 15-Medicine Meets Virtual Reality*, volume 125, pages 13–18. IOP Press, 2007.

[5] Andrew Nealen, Matthias Mller, Richard Keiser, Eddy Boxerman, and Mark Carlson. Physically Based Deformable Models in Computer Graphics. *Computer Graphics Forum*, 25:809–836, 2006.

[6] Jan Bender, Kenny Erleben, and Jeff Trinkle. Interactive simulation of rigid body dynamics in computer graphics. *Comput. Graph. Forum*, 33(1):246–270, 2014.

[7] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. *J. Vis. Comun. Image Represent.*, 18(2):109–118, April 2007.

[8] Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. Projective dynamics: fusing constraint projections for fast simulation. *ACM Transactions on Graphics (TOG)*, 33(4):154, 2014.

[9] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 43–54. ACM, 1998.

[10] Jan Bender, Dan Koschier, Patrick Charrier, and Daniel Weber. Position-based simulation of continuous materials. *Computers & Graphics*, 44:1–10, 2014.

[11] Matthias Teschner, Stefan Kimmerle, Bruno Heidelberger, Gabriel Zachmann, Laks Raghupathi, Arnulph Fuhrmann, M-P Cani, François Faure, Nadia Magnenat-Thalmann, Wolfgang Strasser, et al. Collision detection for deformable objects. In *Computer graphics forum*, volume 24, pages 61–81, 2005.

[12] René Weller. A brief overview of collision detection. In *New Geometric Data Structures for Collision Detection and Haptics*, pages 9–46. Springer, 2013.

[13] Christer Ericson. *Real-Time Collision Detection (The Morgan Kaufmann Series in Interactive 3-D Technology)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[14] Oren Tropp, Ayellet Tal, and Ilan Shimshoni. A fast triangle to triangle intersection test for collision detection. *Computer Animation and Virtual Worlds*, 17(5):527–535, 2006.

[15] Gareth Bradshaw and Carol O'Sullivan. Sphere-tree construction using dynamic medial axis approximation. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 33–40. ACM, 2002.

[16] Gareth Bradshaw and Carol O'Sullivan. Adaptive medial-axis approximation for sphere-tree construction. *ACM Transactions on Graphics (TOG)*, 23(1):1–26, 2004.

[17] Rene Weller and Gabriel Zachmann. A unified approach for physically-based simulations and haptic rendering. In *Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games*, pages 151–159. ACM, 2009.

[18] Cesar Mendoza and Carol OSullivan. Interruptible collision detection for deformable objects. *Computers & Graphics*, 30(3):432–438, 2006.

[19] Doug L James and Dinesh K Pai. Bd-tree: output-sensitive collision detection for reduced deformable models. *ACM Transactions on Graphics (TOG)*, 23(3):393–398, 2004.

[20] Junjun Pan, Chengkai Zhao, Xin Zhao, Aimin Hao, and Hong Qin. Metaballs-based physical modeling and deformation of organs for virtual surgery. *The Visual Computer*, pages 1–11, 2015.

[21] Ehsan Arbabi, Ronan Boulic, and Daniel Thalmann. Fast collision detection methods for joint surfaces. *Journal of biomechanics*, 42(2):91–99, 2009.

[22] Anderson Maciel, Ronan Boulic, and Daniel Thalmann. Efficient collision detection within deforming spherical sliding contact. *Visualization and Computer Graphics, IEEE Transactions on*, 13(3):518–529, 2007.

[23] A. Maciel, T. Halic, Z. Lu, L.P. Nedel, and S. De. Using the physx engine for physics-based virtual surgery with force feedback. *International Journal of Medical Robotics and Computer Assisted Surgery*, 5(3):341–353, 2009.

[24] Jun J Pan, Jian Chang, Xiaosong Yang, Jian J. Zhang, Tahseen Qureshi, Robert Howell, and Tamas Hickish. Graphic and haptic simulation system for virtual laparoscopic rectum surgery. *International Journal of Medical Robotics and Computer Assisted Surgery*, 7(3):304–317, 2011.

[25] Hadrien Courtecuisse, Jrmie Allard, Pierre Kerfriden, Stphane P.A. Bordas, Stphane Cotin, and Christian Duriez. Real-time simulation of contact and cutting of heterogeneous soft-tissues. *Medical Image Analysis*, 18(2):394 – 410, 2014.

[26] Junjun Pan, Junxuan Bai, Xin Zhao, Aimin Hao, and Hong Qin. Real-time haptic manipulation and cutting of hybrid soft tissue models by extended position-based dynamics. *Computer Animation and Virtual Worlds*, 26(3-4):321–335, 2015.

[27] Jun Wu, Rüdiger Westermann, and Christian Dick. Physically-based simulation of cuts in deformable bodies: A survey. In *Eurographics 2014 - State of the Art Reports, Strasbourg, France, 2014*, pages 1–19, 2014.

[28] Mohamed A ElHelw, Benny P Lo, Ara Darzi, and Guang-Zhong Yang. Real-time photo-realistic rendering for surgical simulations with graphics hardware. In *Medical Imaging and Augmented Reality*, pages 346–352. Springer, 2004.

[29] Yi-Je Lim, Wei Jin, and Suvranu De. On some recent advances in multimodal surgery simulation: A hybrid approach to surgical cutting and the use of video images for enhanced realism. *Presence*, 16(6):563–583, 2007.

[30] Naty Hoffman. Background: Physically-based shading. In *SIGGRAPH '10: ACM SIGGRAPH 2010 Courses*. ACM, 2010.

[31] Charles de Rousiers Sébastian Lagarde. Moving frostbite to physically based rendering. In *SIGGRAPH '14: ACM SIGGRAPH 2014 Courses*. ACM, 2014.

[32] Matthias Müller, Bruno Heidelberger, Matthias Teschner, and Markus Gross. Meshless deformations based on shape matching. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 471–478. ACM, 2005.

[33] Tae-Yong Kim, Nuttapong Chentanez, and Matthias Müller-Fischer. Long range attachments - A method to simulate inextensible clothing in computer games. In *Pro-*

ceedings of the 2012 Eurographics/ACM SIGGRAPH Symposium on Computer Animation, SCA 2012, Lausanne, Switzerland, 2012, pages 305–310, 2012.

[34] Jeffrey Smith, Jessica Hodgins, Irving Oppenheim, and Andrew Witkin. Creating models of truss structures with optimization. In *ACM Transactions on Graphics (TOG)*, volume 21, pages 295–301. ACM, 2002.

[35] Chunlin Wu and Xuecheng Tai. A level set formulation of geodesic curvature flow on simplicial surfaces. *Visualization and Computer Graphics, IEEE Transactions on*, 16(4):647–662, July 2010.

[36] Tatiana Surazhsky, Evgeni Magid, Octavian Soldea, Gershon Elber, and Ehud Rivlin. A comparison of gaussian and mean curvatures estimation methods on triangular meshes. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 1, pages 1021–1026. IEEE, 2003.

[37] Matthias Müller, Nuttapong Chentanez, Tae-Yong Kim, and Miles Macklin. Strain based dynamics. In *The Eurographics / ACM SIGGRAPH Symposium on Computer Animation, SCA '14, Copenhagen, Denmark, 2014.*, pages 149–157, 2014.

[38] Brent Burley. Physically-based shading at disney. In *SIGGRAPH '12: ACM SIGGRAPH 2012 Courses*. ACM, 2012.

[39] Eric Heitz. Understanding the masking-shadowing function in microfacet-based brdfs. *Journal of Computer Graphics Techniques*, 3(2):32–91, 2014.

[40] Christophe Schlick. An inexpensive BRDF model for physically-based rendering. *Comput. Graph. Forum*, 13(3):233–246, 1994.

[41] Yu-Ting Tsai and Zen-Chung Shih. All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. *ACM Trans. Graph.*, 25(3):967–976, 2006.

[42] Nikolas Kasyan. Playing with real-time shadows. In *SIGGRAPH '13: ACM SIGGRAPH 2013 Courses*. ACM, 2013.

[43] Ravi Ramamoorthi and Pat Hanrahan. An efficient representation for irradiance environment maps. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 497–500. ACM, 2001.

[44] Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. Unified particle physics for real-time applications. *ACM Trans. Graph.*, 33(4):153:1–153:12, 2014.

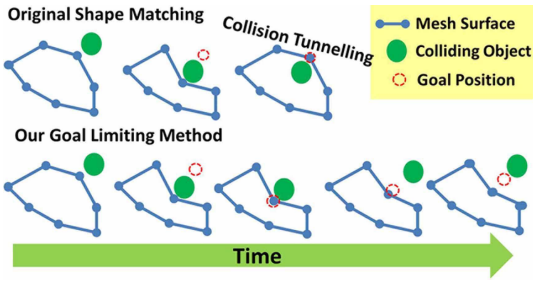

Figure 1: Comparison between our method and shape matching. The goal position of our method is limited to the position of collided vertex and then gradually recover to the position of original shape.

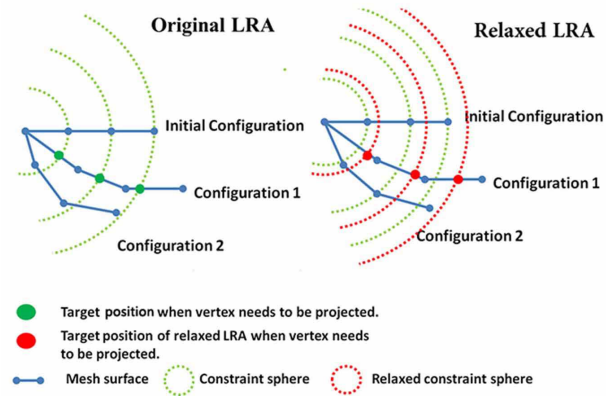

Figure 2: Comparison between original LRA and relaxed LRA

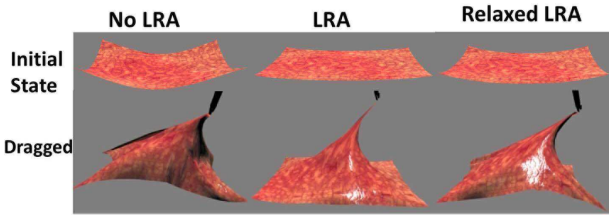

Figure 3: The improvement by using relaxed LRA

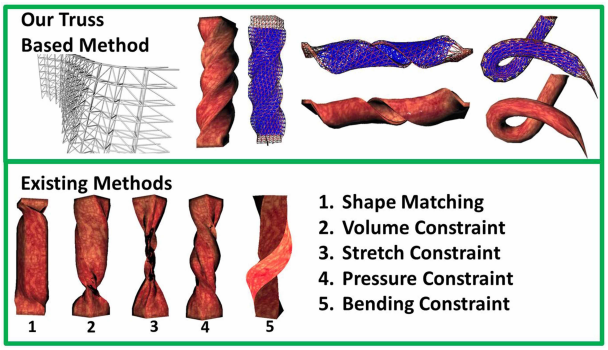

Figure 4: Truss structure based deformation and comparison of the deformation results using existing constraints

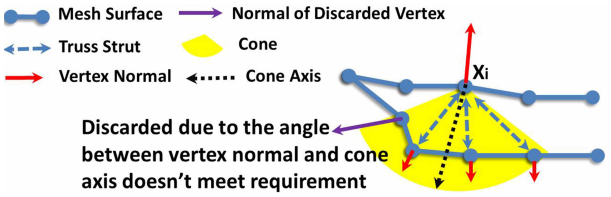

Figure 5: Truss based structure initialization

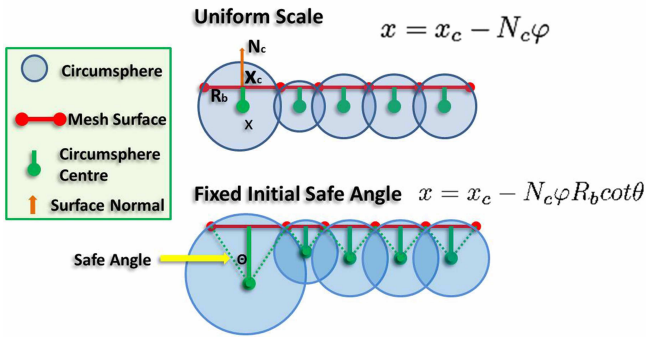

Figure 6: The generation of the initial circumpshere

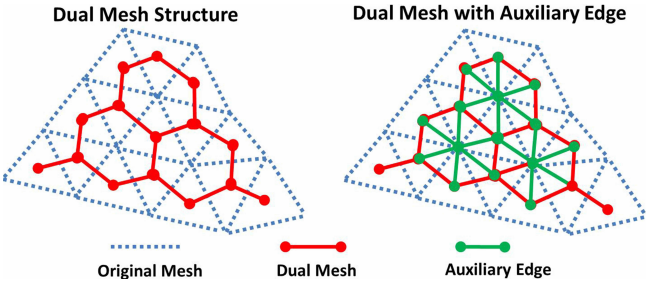

Figure 7: The structure of dual mesh and auxiliary edges

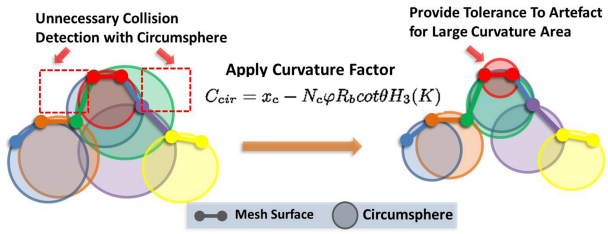

Figure 8: Adjust the size of circumsphere according to curvature

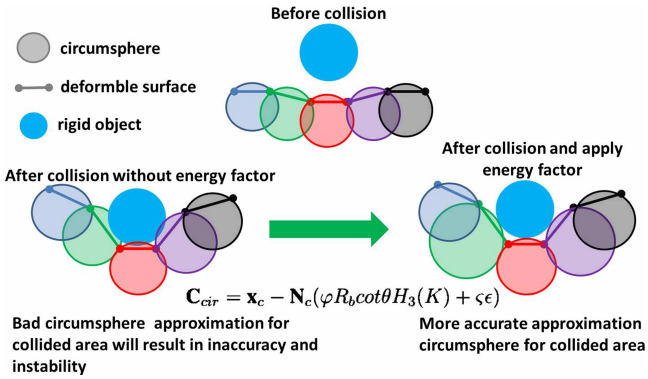

Figure 9: Adjust the size of circumsphere according to material property

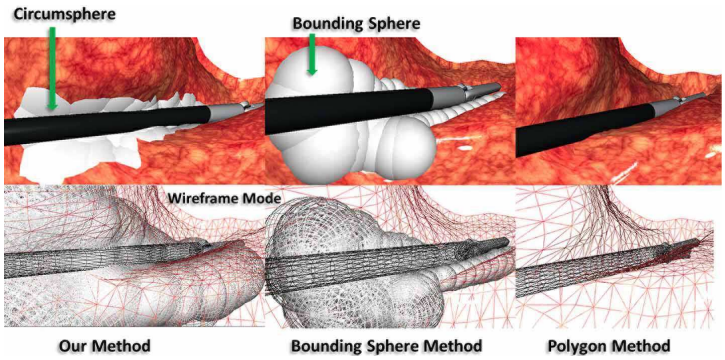

Figure 10: Overview of the comparison between our method and other methods

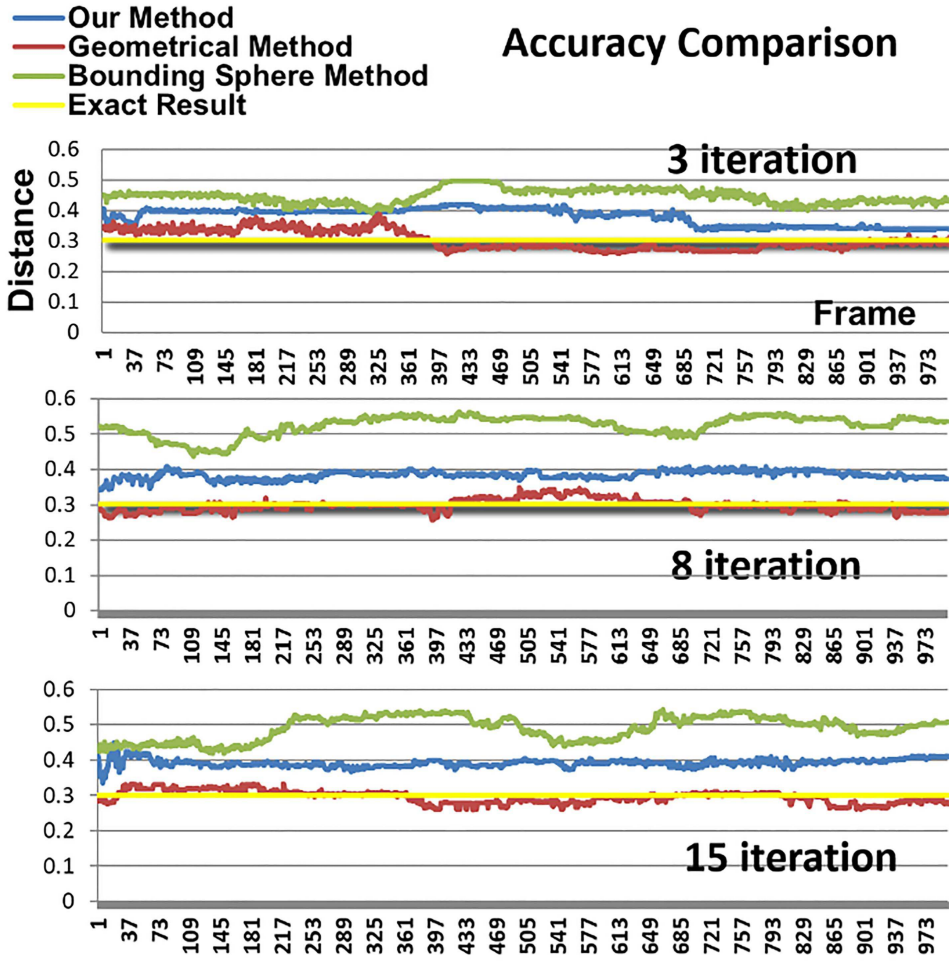

Figure 11: Method analysis from accuracy perspective

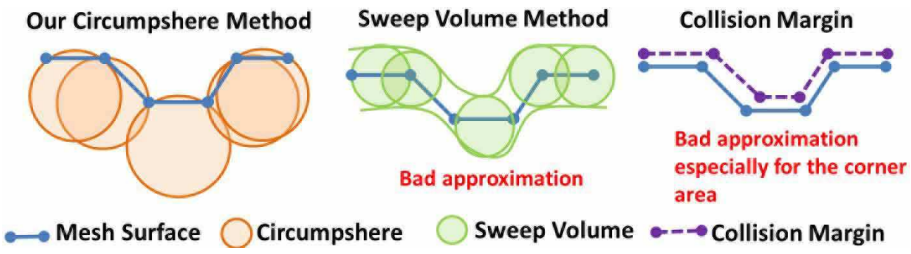

Figure 12: Compare our method with the collision margin and sweep volume

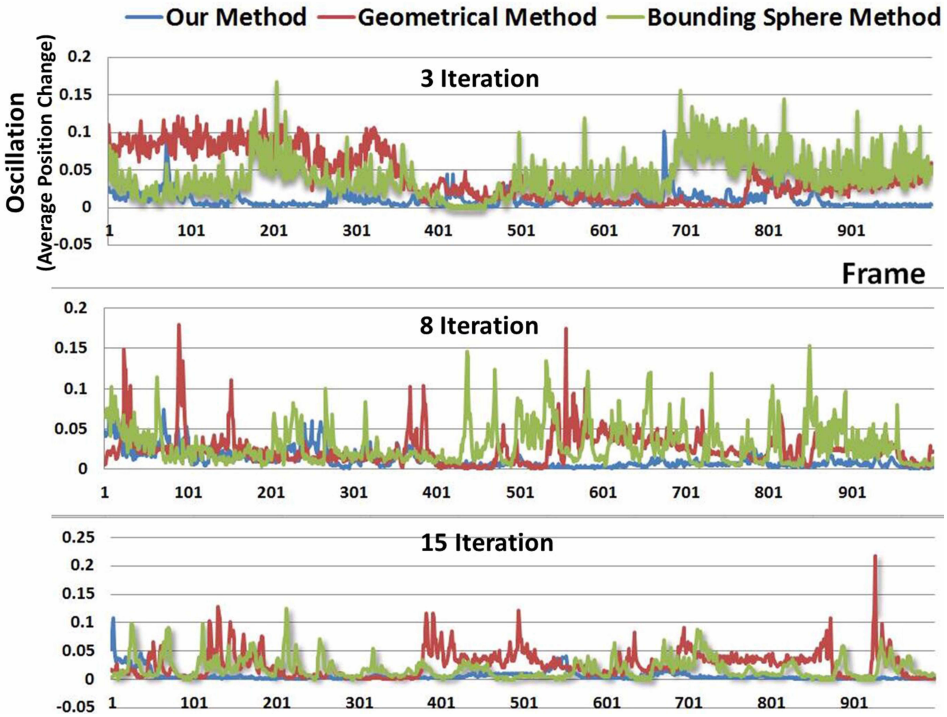

Figure 13: Stability comparison under different iteration number

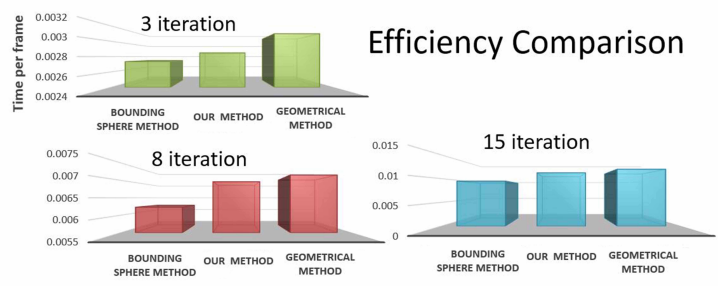

Figure 14: Efficiency comparison between our method and others

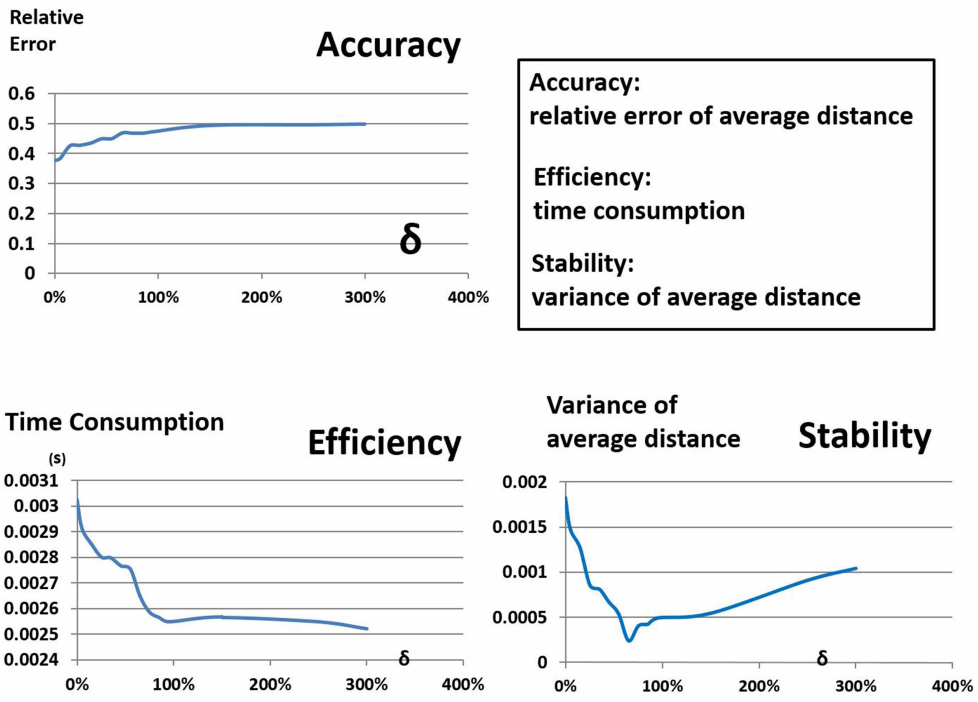

Figure 15: Comparison of accuracy, stability and efficiency for different δ

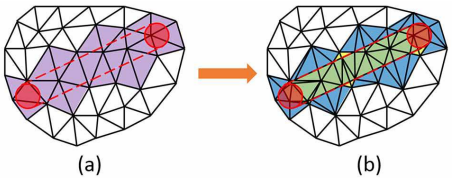

Figure 16: The process of surface cutting. (a)The intersected triangles are in purple color. (b)After cutting, the green triangles (inside V_{sweep}) and yellow triangles (too small) are removed, blue triangles are reserved.

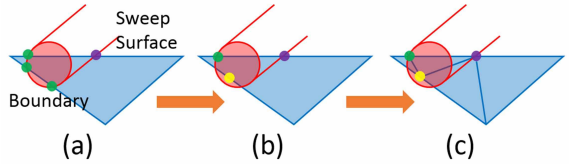

Figure 17: Combination and subdivision. (a)The green points are the intersections on the shape's boundary, the purple point is the intersection on the sweep surface. (b)The boundary intersections on the same edge are combined into a median point. (c)The triangle primitive is subdivide according to these intersections.

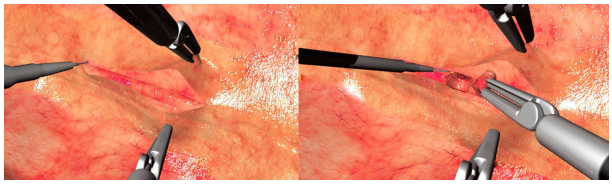

Figure 18: The dissection of fascia and volumetric mesh.

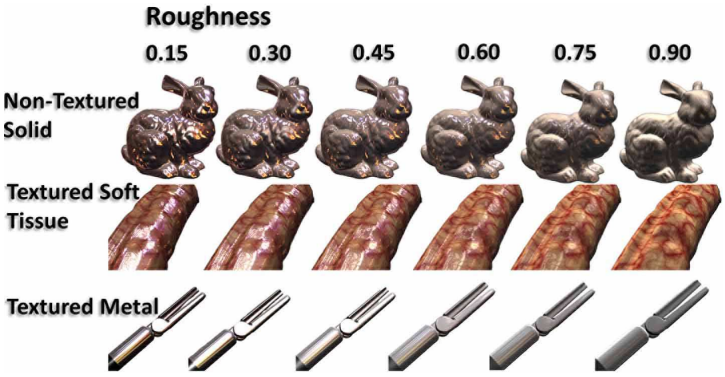

Figure 19: Demonstration of objects using different material

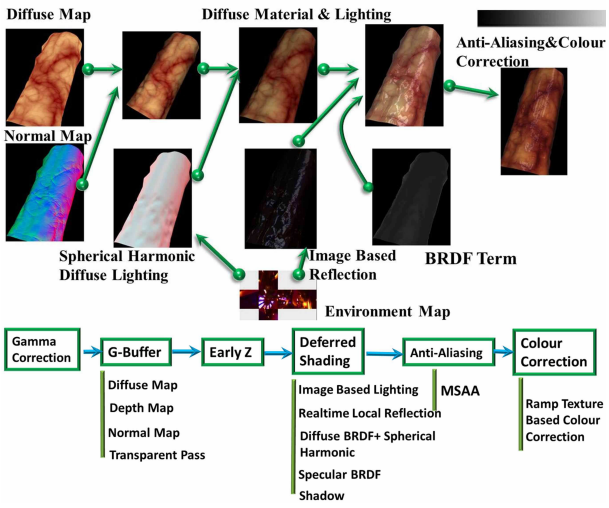

Figure 20: Overview of the rendering procedure and pipeline

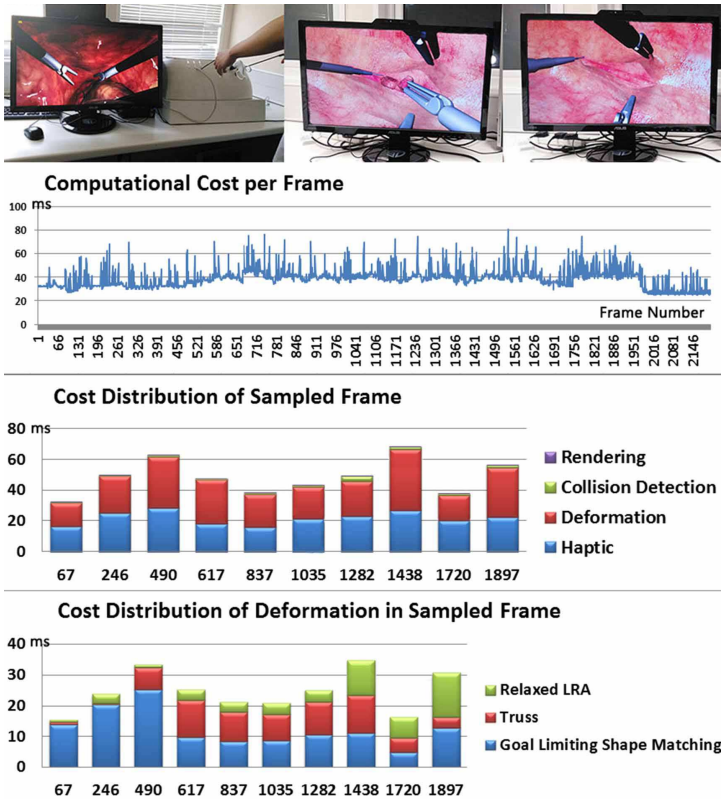

Figure 21: Computatioanl cost distribution analysis

Model	Abdominal Tissue	Rectum	Nerve and Vessel	Fascia
goal limiting factor (η)	0.75	0.15	0.01	-
level of relaxation β	-	-	-	0.1$\sim$0.4
bending threshold $\theta_{threshold}$	-	-	$\pi/3$	-
minimum stiffness factor k_{min}	-	-	0.05	-
safe angle θ	$\pi/6$	$\pi/6$	$\pi/6$	$\pi/6$
global scale factor φ	2	3	1	2
energy scale factor ς	0.9	0.7	0.95	0.85
circumsphere updating threshold δ	1	0.7	0.7	0.65
eliminating threshold $\upsilon_{threshold}$	-	2	2	0.15

Table 1: Parameter settting for the laparoscopic surgery simulator.

Algorithm 1 Goal Limiting Shape Matching

Parameters: time step (h), goal position limiting factor for $\mathbf{x}_i$ (η_i), vertex velocity ($\mathbf{v}_i$), position in rest configuration ($\mathbf{x}_i^0$), recovering rate (R_r), stiffness factor (α)

procedure GOAL_POSITION_LIMITING

  for all Vertex $\mathbf{x}_i$ **do**

    $\mathbf{v}_i = \mathbf{v}_i + \alpha\frac{\mathbf{g}_i - \mathbf{x}_i}{h} + h\mathbf{f}_{ext}/m_i$

    $\mathbf{p}_i = \mathbf{x}_i + \mathbf{v}_i h$

  endfor

  for all cluster $c_i \in C$ **do**

    Compute the mass center $\mathbf{p}_{cm}^{c_i}$ for all $\mathbf{p}_i \in c_i$

    Compute the mass center $\mathbf{x}_{cm}^{0c_i}$ for all vertices $\mathbf{x}_i^0 \in c_i$

    Compute $\mathbf{A}_{c_i} = \sum_i^n m_i(\mathbf{p}_i - \mathbf{p}_{cm}^{c_i})(\mathbf{x}_i^0 - \mathbf{x}_{cm}^{0c_i})$

    Caculate $\mathbf{R}_{c_i}$=$polorDecomposition(\mathbf{A}_{c_i})$

    for all $\mathbf{p}_i \in c_i$ **do**

      if $\mathbf{p}_i$ has been collided **then**

        $\eta_i = 1$

      else

        $\eta_i = max(\eta_i - R_r, 0)$

      $\mathbf{g}_i^{c_i} = R(\mathbf{x}_i^0 - \mathbf{x}_{cm}^{0c_i}) + \mathbf{p}_{cm}^{c_i}$

      $\mathbf{g}_i += \mathbf{g}_i^{c_i} + \eta_i(\mathbf{p}_i - \mathbf{g}_i^{c_i})$

    endfor

  endfor

  for all cluster $c_i \in C$ **do**

    Get the number of clusters (n_c) sharing $\mathbf{g}_i$.

    $\mathbf{g}_i / = n_c$

  endfor

Algorithm 2 Truss Structure Pre-processing

Input: vertex position ($\mathbf{x}_i$), vertex normal ($\mathbf{n}_i$), cone angle (θ), minimum strut stiffness (k_{min})

procedure TRUSS_STRUCTURE_GENERATION

for all vertex $\mathbf{x}_i$ **do**

for all Face $\mathbf{f}_k$ **do**

Caculate face normal $\mathbf{nf}_k$

if $dotproduct(\mathbf{n}_i, \mathbf{nf}_k) < 0$ **then**

Find the nearest face ($\mathbf{f}_n$) intersected with $-\mathbf{n}_i$

endfor

$\mathbf{d} = \mathbf{f}_n^{center} - \mathbf{x}_i$, where $\mathbf{f}_n^{center}$ is the center of $\mathbf{f}_n$

Using $\mathbf{d}$ and user-define θ to generate a cone κ_i

endfor

for all vertex $\mathbf{x}_i$ **do**

for all vertex j inside the cone κ_i **do**

if $dotproduct(\mathbf{n}_i, \mathbf{n}_j) < \theta_{threshold}$ **then**

Connect $\mathbf{x}_i$ with $\mathbf{x}_j$ as strut;

endfor

Find max and minimum strut length D_{max} and D_{min};

for all vertex j inside the cone κ_i **do**

Calculate the stiffness factor

$k_{ij} = (distance(\mathbf{x}_i, \mathbf{x}_j) - D_{max})/(D_{min} -$

$D_{max}) + k_{min}$

endfor

endfor

Algorithm 3 Surface Cutting

procedure SURFACE CUTTING

  Generate the sweep volume V_{sweep}

  Compute intersections

  for all face f_i inside sweep volume V_{sweep} **do**

    Delete f_i from the mesh
  end for

  for all intersected face f_j **do**

    Compute the area A_j for face f_j

    Subdivide face f_j into small triangles $\{f_{j_k}\}$

    for all newly generated face f_{j_k} **do**

      if face f_{j_k} is inside V_{sweep} **then**

        Delete face f_{j_k}

      else

        Compute the area A_{j_k} for f_{j_k}

        if $A_{j_k}/A_j < \upsilon_{threshold}$ **then**

          Delete face f_{j_k}
    end for
  end for