Energized Soft Tissue Dissection in Surgery Simulation

Kun Qian[1], Tao Jiang[1], Meili Wang[2], Xiaosong Yang[1], and Jianjun Zhang[1]

[1]National Center for Computer Animation, Bournemouth University
[2]Northwest A&F University

Abstract

With the development of virtual reality technology, surgery simulation has become an effective way to train the operation skills for surgeons. Soft tissue dissection, as one of the most frequently performed operations in surgery, is indispensable to an immersive and high fidelity surgery simulator. Energized dissection tools are much more commonly used than the traditional sharp scalpels for patient safety. Unfortunately, the interaction of such tools with the soft tissues has been largely ignored in the research of surgical simulators. In this paper, we have proposed an energized soft tissue dissection model. We categorize the soft tissues into three types (fascia, membrane and fat) and simulate their physical property accordingly. The dissection algorithm we propose employs an edge-based structure, which offers an effective mechanism for the generation of incisions dissected with energized tools. The mesh topology will not be changed when it is dissected by an energized tool, rather it is controlled by the heat transfer model. Our dissection method is highly compatible and efficient to the physically based simulation resolved by a pre-factorized linear system.

Keywords: energized dissection, deformation simulation, strain limiting, surgery simulation

1 Introduction

Soft tissue dissection, which is defined as the separation of tissues with haemostasis, is one of the most frequently performed operations in surgery. It mainly consists of three steps: exposure, stabilization and division [1]. Current virtual reality based surgical simulators usually treat the soft tissues as homogeneous elastic materials and the cutting tools as a geometric plane which simply dissects the intersected meshes in a straightforward manner. Unfortunately, this simplification is not computationally efficient and does not reflect the reality.

Figure 1: Fascia and fat dissection in real surgery

To reduce the risk to the patient, the contact between soft tissue and sharp instruments should be minimized purposefully. Mechanical energy based dissection systems have been incorporated into modern surgery, such as electricity, diathermy, ultrasound etc. With the energized tools, dissection is performed depending on the amount of energy the soft tissue receives and its material property. As the material of soft tissue is composed of fat, fascia, lymph, nerve etc., some of the dissected area may still connect after dissection because different tissues react differently to high energy, see figure 1. Such complex incision generated by energized tools is challenging for real-time simulation.

In computer graphics research, the energized

tool based dissection has not been widely studied yet. Most of the researches focus on accurate cutting which is not suitable for the energized dissection. The newly generated degree of freedom caused by topology changed method will heavily influence the performance of physics simulation which is dependent on the solving of pre-factorized linear system.

In this paper, we propose an energized soft tissue dissection model. Our main contributions are:

- Proposing different physically based simulation strategies for three kinds of soft tissues: fascia, membrane and fat, rather than treating the soft tissues as uniform elastic materials.
- Proposing a computationally efficient edge based dissection model.
- Proposing an energized tool based dissection using heat transfer model.

2 Related Works

Deformation simulation is very important for the modeling of soft tissue. For a complete survey, readers can refer to [2][3][4]. Currently, the real-time deformation techniques such as mass spring method [5], finite element method [6], position based dynamics [7][8], projective dynamics [9] have been widely used in surgery simulation [10] and anatomy modeling [9][11].

The force based method such as the mass spring and finite element methods suffers from the instability problem. Mass spring method can not capture the volume effect of soft tissue. Finite element method can provide more accurate physics simulation but it is computationally expensive because it contains complex matrix operations [12]. Position based dynamics (PBD) enjoyed wide application due to its unconditional stability and high efficiency. However, PBD suffers from low convergence rate and inaccuracy. To improve accuracy, Jan Bender et al.[13] incorporated continuous material energy into PBD framework based on the energy reduction. To improve the convergence, a hierarchy multigrid structure [14] and unified solver [15] has been proposed. The accuracy of PBD is dependent on the iteration counts. However, the linear momentum of the object will be washed away by using large literation count [9].

The projective dynamics [14] provides a fast convergent, efficient and accurate solver based on a two steps numerical optimization. As a general form of PBD, it conserves the momentum by finding a compromise between momentum and internal elastic energy. However, the efficiency of projective dynamics solver will be influenced by the topology change because the solution is dependent on a pre-factorized linear system. In this paper, we proposed an efficient dissection method which can reduce such influence.

Wu et al. [16] gives a comprehensive introduction to the recent cutting techniques used in physics based simulation. Sifakis et al.[17] proposed a cutting strategy for arbitrary tetrahedron. However, such topology changed methods [18][19] are not efficient for physical simulation especially for the global solving based method. To improve the influence caused by topology change, Wu et al. [20][21][22] proposed fast topology modification optimization. To improve the simulation efficiency, the hybird dissection methods [23][24] and implicit shape based dissection [8] is developed. To simulate multi-layer material cutting, spring based multilayer soft object dissection has been proposed in [25] which uses implicit adhesive springs to connect each layer. However, those techniques can not well simulate the incision pattern from real surgery shown in figure 1. That is the problem we want to solve in this paper.

3 Soft Tissue Structure Overview

We categorize the soft tissue into three groups: fascia, membrane and fat. The fascia refers to a band of connective tissue beneath the skin that attaches, stabilizes, encloses, and separates muscles and other internal organs [26], such as ligaments, aponeuroses, and tendons. The fascia is mainly composed of fibrous connective tissue which encloses packed bundles of collagen fibers parallel to the direction of pull. The elastin confers stiffness to the fascia and store most of the energy. Due to the inextensibility of the collagen fibers, the tension caused by the taut fascia will grow when the collagen is gradually

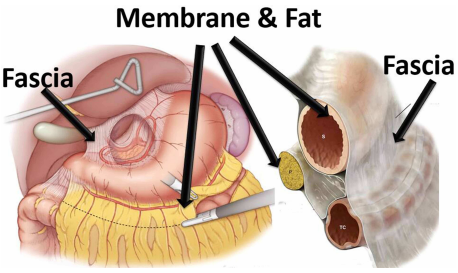

Figure 2: The connectivity between different types of soft tissues

pulled and stretched in one direction.

For the fatty tissue, it mainly includes water, carbohydrate and protein which makes it like a hyperelastic object combined with damping resistance. When the stretch of fatty soft tissue exceed certain limit, it will not recover to its initial shape. Figure 2 shows the overall structure of our multi-layer soft tissue.

The physical property of membrane is similar to fascia. Therefore,

- the fascia and membrane is simulated using anisotropic strain limiting for normal and shear strain. The strain direction is dependent on the direction of the tissue fibers.
- the fatty tissue is simulated using hyperelastic and plastic model.

4 Deformation Model

Before introducing the deformation model, we give the definition of some variables in table 1.

Integration scheme is important for a dynamics system because it influences the stability and accuracy of the system. Explicit integration will cause instability under large time step or high stiffness. Implicit integration scheme can overcome this issue. The implicit integration update the positions and velocities as:

$$\mathbf{v}^{t+1} = \mathbf{v}^t + t\mathbf{M}^{-1}(\mathbf{f}_{int} + \mathbf{f}_{ext}) \quad (1)$$

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \Delta t \mathbf{v}^{t+1} \quad (2)$$

where Δt is the time step. The following equation can be derived from equation 1 and 2:

$$\mathbf{M}(\mathbf{x}^{t+1} - \mathbf{x}^t - \Delta t \mathbf{v}^t) = \Delta t^2(\mathbf{f}_{ext} + \mathbf{f}_{int}) \quad (3)$$

Name	Description
n	The number of vertices. $n \in R$
$\mathbf{v}$	Velocity state vector in current configuration, which assembles all vertices' velocity into one column vector. $\mathbf{v} \in R^{3n}$
$\mathbf{x}$, $\mathbf{X}$	Position state vector in current configuration ($\mathbf{x}$) and rest configuration ($\mathbf{X}$), which assembles all vertices into one column vector. $\mathbf{x}, \mathbf{X} \in R^{3n}$
$\mathbf{x}_i$, $\mathbf{X}_i$	The ith vertex position. $\mathbf{x}_i, \mathbf{X}_i \in R^3$
$\mathbf{x}^t$	The position state at time t. $\mathbf{x}^t \in R^{3n}$
$\mathbf{f}_{int}(\mathbf{x})$, $\mathbf{f}_{ext}(\mathbf{x})$	The position independent internal force ($\mathbf{f}_{int}(\mathbf{x})$) and external force ($\mathbf{f}_{ext}(\mathbf{x})$) state vector. $\mathbf{f}_{int}, \mathbf{f}_{ext} \in R^{3n}$
$\mathbf{C}$	The constraint set, which contains N constrains $(C_1, C_2 ... C_N)$
$\mathbf{M}, \mathbf{M}^i$	$\mathbf{M}$ is the mass matrix. $\mathbf{M} \in R^{3n}$. $\mathbf{M}^i$ is the mass matrix for the ith node.
$\mathbf{M}_m$	$\mathbf{M}_m$ is the mass matrix for the nodes involved in the mth constraint $C_m \in \mathbf{C}$
$\widetilde{\mathbf{x}}_m$	Vertices involved in the mth constraint $C_m \in \mathbf{C}$.
$\mathbf{E}(\mathbf{x})$	Total internal energy generated by $\mathbf{f}_{int}(\mathbf{x})$, $\mathbf{E}(\mathbf{x}) \in R$, $\mathbf{f}_{int}(\mathbf{x}) = -\nabla_{\mathbf{x}}\mathbf{E}(\mathbf{x})$
$\mathbf{E}_m(\mathbf{x})$	The internal energy generated by the mth constrain C_m, $\mathbf{E}_m(\mathbf{x}) \in R$
$\mathbf{x}_m^p$	Auxiliary position variable for constraint $C_m \in \mathbf{C}$, $\mathbf{x}_m^p$ is the projection of $\widetilde{\mathbf{x}}_m$ onto its corresponding undeformed constraint manifold.

Table 1: Mathematic definitions

Solving this equation equals to find the critical point of the following minimization problem:

$$\min_{\mathbf{x}^{t+1}} \frac{1}{2\Delta t^2} \|\mathbf{M}^{\frac{1}{2}}(\mathbf{x}^{t+1} - \mathbf{s}^t)\|_F^2 + \mathbf{E}(\mathbf{x}^{t+1}) \quad (4)$$

where $\mathbf{s}^t = \mathbf{x}^t + \Delta t \mathbf{v}^t + \Delta t^2 \mathbf{M}^{-1}\mathbf{f}_{ext}$ actually reflects the momentum under the influence of external forces. The total internal energy equals to summation of the energies generate by all the constraints $\mathbf{E}(\mathbf{x}^{t+1}) = \sum_{C_m \in \mathbf{C}} E_m(\mathbf{x}^{t+1})$. To make the notation easy to understand, we will represent $\mathbf{x}^t$, $\mathbf{s}^t$ using $\mathbf{x}$ and $\mathbf{s}$.

In the projective dynamics [9], the projective means it projects the vertices which involved in a constraint (current state) back to the undeformed manifold of this constraint. The energy

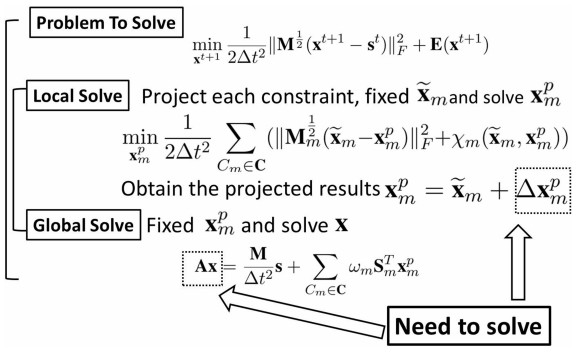

Figure 3: The workflow of projective dynamics

$\mathbf{E}(\mathbf{x})$ is formulated as:

$$\frac{1}{2\Delta t^2} \sum_{C_m \in \mathbf{C}} (\|\mathbf{M}_m^{\frac{1}{2}}(\widetilde{\mathbf{x}}_m - \mathbf{x}_m^p)\|_F^2 + \chi_m(\widetilde{\mathbf{x}}_m, \mathbf{x}_m^p)) \tag{5}$$

$\widetilde{\mathbf{x}}_m$ can be obtained through the constant selection matrix $\mathbf{S}_m$ where $\widetilde{\mathbf{x}}_m = \mathbf{S}_m\mathbf{x}$. Assuming the mth constraint $C_m \in \mathbf{C}$, it involves M vertices so $\mathbf{S}_m \in R^{3M\times 3n}$. $\chi_m(\widetilde{\mathbf{x}}_m, \mathbf{x}_m^p)$ is a function to penalize $\mathbf{E}(\mathbf{x})$ when the projection of $\widetilde{\mathbf{x}}_m$ is not on the undeformed constraint manifold which means $\mathbf{E} \to \infty$. When $\widetilde{\mathbf{x}}_m$ is projected to the undeformed constraint manifold, $\mathbf{E}(\mathbf{x}) = 0$ and $\mathbf{x}_m^p$ stores the projected results of the vertices involved in the mth constraint.

How to project $\widetilde{\mathbf{x}}_m$ onto the undeformed manifold can be found in [7][9]. The main idea is to project the constraint in the negative direction of its gradient and linearize the constrain in this direction until the constraint vanish in this direction, then we get the $\Delta\mathbf{x}_p^m$ which is used to update the current state ($\widetilde{\mathbf{x}}_m$) to the projected state ($\mathbf{x}_m^p$).

$$\mathbf{x}_m^p = \widetilde{\mathbf{x}}_m + \Delta\mathbf{x}_m^p \tag{6}$$

The derivation of the equation 4 can be formulated as:

$$\mathbf{Ax} = \frac{\mathbf{M}}{\Delta t^2}\mathbf{s} + \sum_{C_m \in \mathbf{C}} \omega_m \mathbf{S}_m^T \mathbf{x}_m^p \tag{7}$$

where $\mathbf{A} = (\frac{\mathbf{M}}{\Delta t^2} + \sum_{C_m \in \mathbf{C}} \omega_m \mathbf{S}_m^T \mathbf{S}_m)$. The whole algorithm is generalized in figure 3.

In equation 7, $\mathbf{A}$ is a constant matrix which can be precomputed in the initial stage, making the solve of this sparse linear system very efficient. However, when the mesh topology changed, the left hand side matrix is needed to be changed which affects the performance significantly. Our method will fix this problem.

5 Modeling of Three Different Soft Tissues

5.1 Strain Limiting for Fascia and Membrane

To model the strain limiting for fascia and membrane, finding the direction of limiting is important. The texture of fascia and membrane can well reflect the fibers direction so we limit the strain in the direction of its material coordinate.

According to the general Hook's Law, stress is dependent on strain (ϵ). ϵ is a function of the deformation gradient $\mathbf{F}$. For a surface mesh, let $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)$ be the world positions of a triangle element's vertices, the deformation gradient can be calculated as:

$$\mathbf{F} = \mathbf{D}_s\mathbf{D}_m^{-1} \tag{8}$$

where $\mathbf{D}_s = [\mathbf{x}_j - \mathbf{x}_i, \mathbf{x}_k - \mathbf{x}_i]$, $\mathbf{D}_s \in \mathbf{R}^{3\times 2}$, $\mathbf{D}_m = [\mathbf{X}_j - \mathbf{X}_i, \mathbf{X}_k - \mathbf{X}_i]$, $\mathbf{D}_m \in \mathbf{R}^{3\times 2}$. However, $\mathbf{D}_m^{-1}$ is not defined because it is not a squared matrix. Let $\mathbf{u}_i \in R^{2\times 1}$ be the texture coordinate of $\mathbf{x}_i$. To project $\mathbf{D}_m$ to the material coordinate, the tangent vectors $(\mathbf{t}_u, \mathbf{t}_v)$ in texture UV direction can be calculated.

$$(\mathbf{t}_u, \mathbf{t}_v) = \mathbf{D}_m(\mathbf{u}_j - \mathbf{u}_i, \mathbf{u}_k - \mathbf{u}_i)^{-1} \tag{9}$$

Then by orthogonalizing $(\mathbf{t}_u, \mathbf{t}_v)$, the new $\mathbf{t}_u$ and $\mathbf{t}_v$ is perpendicular to each other. Then projecting the world coordinate to $(\mathbf{t}_u, \mathbf{t}_v)$ direction to get the material coordinate $\mathbf{D}'_m \in R^{2\times 2}$.

$$\mathbf{D}'_m = (\frac{\mathbf{t}_u}{|\mathbf{t}_u|}, \frac{\mathbf{t}_v}{|\mathbf{t}_v|})^T(\mathbf{x}_j - \mathbf{x}_i, \mathbf{x}_k - \mathbf{x}_i) \tag{10}$$

The deformation has become $\mathbf{F} = \mathbf{D}_s\mathbf{D}'^{-1}_m$. In figure 4, the left column show the material coordinate does not conform to the mesh topology so the strain follows the material coordinate direction. The right column shows material coordinate conform to the mesh topology.

For the strain measurement, we use the rotational invariant and computationally efficient St. Venant-Kirchhoff model to capture the nonlinear material property. Here we limit the strain as:

$$\epsilon = \mathbf{F}^T\mathbf{F} - \mathbf{T} = \mathbf{K} - \mathbf{T} = \begin{bmatrix} k_{00} - t_{00} & k_{01} - t_{01} \\ k_{10} - t_{10} & k_{11} - t_{11} \end{bmatrix} \tag{11}$$

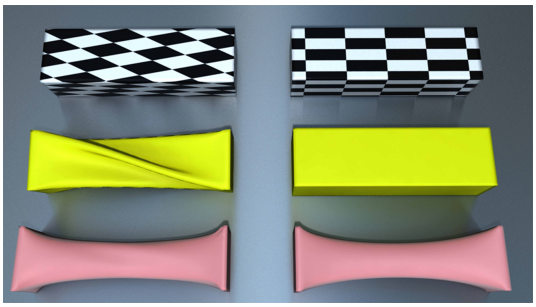

Figure 4: Strain limiting comparison. First row, no strain. Second row: strain limiting in x-axis. Third row: strain limiting in both x and y axis

The diagonal entries and off-diagonal elements of $\mathbf{T} \in \mathbf{R}^{2\times2}$ will limit the normal and shear strain respectively. The strain limiting constraint can be defined as:

$$C_{m_{ab}} = k_{ab} - t_{ab} \tag{12}$$

where k_{ab} and t_{ab} $(a, b = 1 \text{ or } 2)$ are the entries of $\mathbf{K}$ and $\mathbf{T}$ respectively. According to the workflow of projective dynamics described in figure 3, the gradient of the constraint $\nabla_{\mathbf{x}_i}\mathbf{C}_{m_{ab}}, \nabla_{\mathbf{x}_j}\mathbf{C}_{m_{ab}}, \nabla_{\mathbf{x}_k}\mathbf{C}_{m_{ab}}$ need to be computed to obtain the projected result. For the details of the computation, please refer to the **Appendix A**. Finally the position update $\Delta\mathbf{x}^p_{m_t}$ for $\mathbf{x}_t$, $(t = i, j, k)$ which involved in $\widetilde{\mathbf{x}}_m$ can be calculated as:

$$-\sum_{a=1}^{2}\sum_{b=1}^{2}\frac{(\mathbf{M}^t)^{-1}\nabla_{\mathbf{x}_t}C_{m_{ab}}C_{m_{ab}}}{\sum_{w=i,j,k}\|(\mathbf{M}^\omega)^{-\frac{1}{2}}\nabla_{\mathbf{x}_w}C_{m_{ab}}\|_F^2} \tag{13}$$

$\Delta\mathbf{x}^p_{m_t}$ updates $\mathbf{x}_t$ to its projected position. When $a = b$ the strain limiting is caused by normal stress, otherwise, it is caused by shear stress.

5.2 *Modelling of the Fatty Tissue Beneath Membrane*

In this part, we model the elasticity, strain limiting and plasticity properties of fatty tissue beneath the fascia. The fatty tissue is composed of the surface membranes and the fat warped inside. We use tetrahedron as the basic element to represent the fatty tissue. We treat the boundary of the tetrahedral mesh as the surface membranes which we apply the physics properties described in section 5.1. For the fatty tissue inside surface membrane, we describe the modeling method in the following part.

Assuming the tetrahedral element is composed by $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4)$. Now $\mathbf{D}_s = (\mathbf{x}_1 - \mathbf{x}_4, \mathbf{x}_2 - \mathbf{x}_4, \mathbf{x}_3 - \mathbf{x}_4)$, $\mathbf{D}_m = (\mathbf{X}_1 - \mathbf{X}_4, \mathbf{X}_2 - \mathbf{X}_4, \mathbf{X}_3 - \mathbf{X}_4)$. The deformation gradient $\mathbf{F} = \mathbf{D}_s\mathbf{D}_m^{-1} \in \mathbf{R}^{3\times3}$. The energy exist in the nth tetrahedral element τ_n can be calculated as [6]: $\mathbf{E}_{\tau_n} = \Psi(\mathbf{F})$ where $\Psi(\mathbf{F})$ is the energy density function which is dependent on the constitutive model used. We used the St.Venant Kirchhoff model so that $\Psi(\mathbf{F}) = \mu\epsilon : \epsilon + \frac{\lambda}{2}tr^2(\epsilon)$. Here we want to minimize the energy stored inside each element, so we add a energy constraint for each element τ_n as:

$$C_m = \mathbf{E}_{\tau_n} \tag{14}$$

The gradient of the energy constrain is:

$$\nabla_{\mathbf{x}_u}C_m = \frac{\partial\mathbf{E}_{\tau_n}}{\partial\mathbf{F}}\frac{\partial\mathbf{F}}{\partial\mathbf{x}_u} = \mathbf{P}(\mathbf{F})\frac{\partial\mathbf{F}}{\partial\mathbf{x}_u} \tag{15}$$

where $u = 1, 2, 3$, $\mathbf{P}(\mathbf{F})$ is the first Piola-Kirchhoff stress tensor. The above equation can be caluclated as (the details can be referred in **Appendix B**):

$$\nabla_{\mathbf{x}_u}C_m = \begin{bmatrix}\mathbf{e}_1^T\mathbf{P}(\mathbf{F})\mathbf{D}_m^{-1}\mathbf{e}_u \\ \mathbf{e}_2^T\mathbf{P}(\mathbf{F})\mathbf{D}_m^{-1}\mathbf{e}_u \\ \mathbf{e}_3^T\mathbf{P}(\mathbf{F})\mathbf{D}_m^{-1}\mathbf{e}_u\end{bmatrix} \tag{16}$$

where $\mathbf{e}_j \in \mathbf{R}^{3\times1}$ is a selection vector $j = 1, 2, 3$, the jth element of $\mathbf{e}_j$ is 1, other elements equals to 0. So $\nabla_{\mathbf{x}_4}C_m = -(\nabla_{\mathbf{x}_1}C_m + \nabla_{\mathbf{x}_2}C_m + \nabla_{\mathbf{x}_3}C_m)$. Then the position update $\Delta\mathbf{x}^p_{m_t}$ for $\mathbf{x}_t$ (t=1,2,3) which involved in $\widetilde{\mathbf{x}}_m$ can be calculated as:

$$\Delta\mathbf{x}^p_{m_t} = \frac{-(\mathbf{M}^t)^{-1}\nabla_{\mathbf{x}_t}C_mC_m}{\sum_{\omega=1,2,3}\|(\mathbf{M}^\omega)^{-\frac{1}{2}}\nabla_{\mathbf{x}_\omega}C_m\|_F^2} \tag{17}$$

The first Piola-Kirchhoff stress tensor of St.Venant Kirchhoff is [6]: $\mathbf{P}(\mathbf{F}) = \mathbf{F}(2\mu\epsilon + \lambda tr(\epsilon)\mathbf{I})$ where μ, λ are *Láme* coefficients, ϵ is the strain.

For the plasticity property, we use the von Mises yield criterion proposed in [27], they decomposed the strain into elastic part and plastic part $\epsilon = \epsilon_e + \epsilon_p$. When the elastic strain deviation $\epsilon' = \epsilon_e - \frac{Tr(\epsilon_e)}{3}\mathbf{I}_3$ exceed the elastic limit

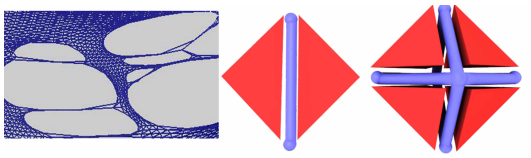

Figure 5: Edge based structure

(γ_1) and within the plastic limit (γ_2), plastic deformation happens.

For tetrahedral element, strain limiting techniques mentioned in section 5.1 can also be used but $\mathbf{F}$ now becomes $\mathbf{F} = (\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3)$.

For the connectivity between fascia and its attach membrane or fatty tissue, we used the truss based structure proposed in [8] to connect the space between fascia and membrane.

6 Dissection Model

6.1 Edge Based Structure

In this section, we presented an edge-based mesh structure which is different from the polygon (triangle, tetrahedral, hexahedral etc.) based structure. We treat the edge as the most basic element for the mesh. Each edge maintains the information of triangles or tetrahedrons which share this edge, see figure 5. Our method can efficiently incorporate the dissected mesh into the physics computational model and produce more realistic incision pattern which generated by energized tools (see figure 7).

In the initial stage, each polygon primitive keeps the counts it has been shared by the edges, we denote the counter for the kth face as $share\_count[k]$. When an edge is dissected, the shared count of this edge's surrounding polygons should be subtracted by one. When rendering the mesh, there is no need to reorganize the index buffer and change the size of vertex buffer. In each frame, we only feed the rendering buffer with the polygon faces whose $share\_count$ remains the same as the initial stage. Further analysis is in section 7.

6.2 Heat Transfer Model

Due to the high energy in the tool, most of the soft tissue will be dissected immediately when touched, so we assume that there is no heat transfer on the soft tissue. The heat transfer is only between soft tissue and energized tools. We build a local $N \times N \times N$ uniform grid coordinate based on the heat source. We take dl as the length step of the grid coordinate and dt as the time step. We donate the temperature at a grid node $\mathbf{n} = (n_x, n_y, n_z)$ as $u(\mathbf{n})$. According to the parabolic heat equation:

$$\frac{\partial u}{\partial t} - \alpha \nabla^2 u = 0 \tag{18}$$

where α is the thermal diffusivity. ∇^2 is the Laplace operator. $\nabla^2 u(\mathbf{n}) = (\frac{\partial^2 u}{\partial x^2}, \frac{\partial^2 u}{\partial y^2}, \frac{\partial^2 u}{\partial z^2})$. To solve this parabolic PDE, the finite difference method can be used. At $u(\mathbf{n})$, the $\frac{\partial^2 u}{\partial x^2}$ can be solved as:

$$\frac{u(n_x + 1, :, :) + u(n_x - 1, :, :) - 2u(n_x, :, :)}{dl^2} \tag{19}$$

It is the same for $\frac{\partial^2 u}{\partial y^2}, \frac{\partial^2 u}{\partial z^2}$. The temperature inside the grid can be achieved by interpolation. We use a conductivity parameter ξ to control how much heat the soft tissue can absorb. Due to our dissection method is based on edge, we calculate the temperature (τ_e) of an edge (e) by using the average temperature at the edge's nodes.

$$\tau_e = (u_a + u_b)\xi \Delta t / 2 \tag{20}$$

where Δt equals to the sum of the time the edge e intersect with the implicit shape proxy of the tool and the time the edge e completely lay inside the implicit shape. When τ_e excess the ignition temperature δ, the edge will be dissected. For different soft tissue, δ have different values. Following are some reference data for the ignition temperature of soft tissues [28].

Ignition Temperature (δ)	
Name	Description
peritoneum (fascia)	172°C ± 17 °C
mesentery (membrane)	96.4°C ± 4.1 °C
liver (fatty)	76°C ± 2.9 °C

6.3 Dissection Area Modelling

Due to the high energy of the energized tool, the newly generated edges and surfaces of the dissected area will shrink because the fibers and

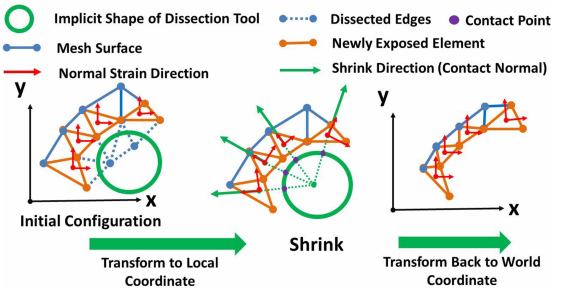

Figure 6: Dissected area modeling

protein inside the tissue is burnt [28]. According to the equation 8, the hyperelastic deformation can be conceived as the process that the deformable object from the $\mathbf{D}_s$ state gradually go back to $\mathbf{D}_m$ state, which equals to the process that $\mathbf{F}$ gradually recover to the identity matrix. If we change the target shape that $\mathbf{F}$ goes back to, the shrink effect can be achieved. The newly exposed surface should shrink perpendicular to the contact surface. To get the contact normal, we approximate the contact point ($\mathbf{x}_c$) as the intersection between the implicit shape of the tool and the lines which determined by the center of the newly exposed element and the center of the implicit shape, see figure 6. Assuming the implicit shape function represents the dissection tool is $f(\mathbf{x}) \in R$, the normal ($\mathbf{N}_c$) at the contact point is $\partial \mathbf{f}/\partial \mathbf{x}_c$. To build the local coordinate for the newly generated primitive, we use the direction of $\mathbf{N}_c$ as one axis. Other two axis ($\mathbf{N}_a$, $\mathbf{N}_b$), we compute them use the face normal ($\mathbf{N}_f$) of the newly generate face and $\mathbf{N}_c$. $\mathbf{N}_a = \mathbf{N}_f \times \mathbf{N}_c$, $\mathbf{N}_b = \mathbf{N}_c \times \mathbf{N}_a$.

To find the direction for the shrinking, we transform the initial normal strain direction to the local coordinate of the contact point. Then applying the strain limiting to the transformed normal strain direction. Finally rotate the current shrinked state back to the original coordinate, to get the shrinkage in world space. We use the St. Venant-Kirchhoff model:

$$\epsilon = \mathbf{F}^T\mathbf{F} - \mathbf{R}^{-1} \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{bmatrix} \mathbf{R} \qquad (21)$$

where $\mathbf{R}$ is the rotation matrix which transforms the initial normal strain direction to the local coordinate of the contact point. S_x, S_y, S_z are the shrink ratio in each direction.

7 Method Comparison and Results

7.1 Complexity Comparison

Before making comparison, we categorized the dissection methods into topology-changed and topology-unchanged. Different from the topology-unchanged methods, the topology-changed methods contain the newly generated vertices. Our method belongs to the topology-unchanged method.

For the buffer update, the topology-changed methods need to change the vertex and index buffers size, which require frequent inserting and deleting operations. We donate the complexity of inserting as $O(ins)$ and deleting as $O(del)$. There is no need for our method to update both buffers. We reference the original vertex and index buffers and render the faces whose *share_count* equals to initial configuration. The complexity of the buffer operation for our method is $O(n)$ because we only need to iterate all the faces once for each simulation step.

For the newly generated primitive, the topology-changed methods need to perform mesh refinement ($O(re)$) and find the neighbors for the newly generated primitives, deleting ($O(N\_del)$) and inserting ($O(N\_ins)$). Our method only needs one iteration of *share_count* to update the neighbour information for vertices and faces ($O(n)$).

For the ill shape handling, the topology-change method needs to concern whether the newly generated polygons are in ill shape, assuming the complexity as $O(ill)$. Our method has not change the topology so that ill shape test is not needed.

The overall complexity of our method is $O(n)$ and the topology changed method is $O(ins) + O(del) + O(N\_ins) + O(N\_del) + O(re) + O(ill)$. Comparing with the topology-unchanged methods, our method has the same complexity as others. However, our edge based structure can generate more patterns than other polygon based dissection methods (figure 5).

7.2 Efficiency Comparison

The governing equation of our simulation system is described in equation 7. The left side

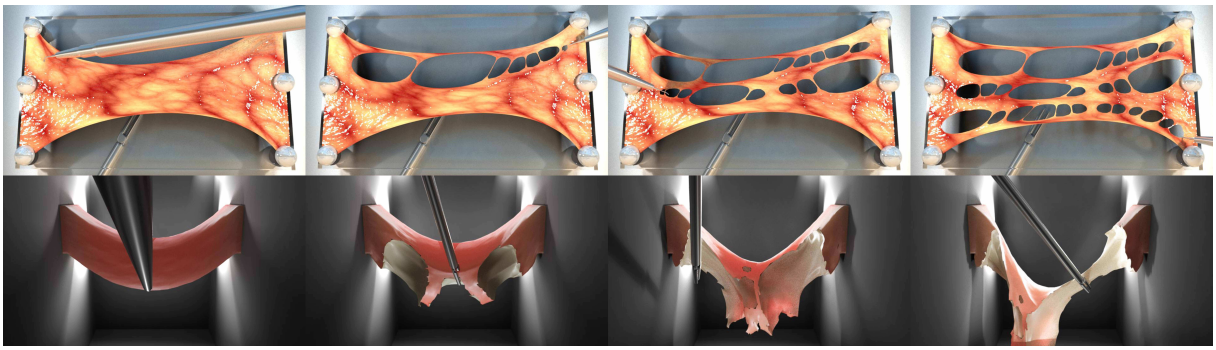

Figure 7: Energized tool dissection using our method

matrix $\mathbf{A} = (\frac{\mathbf{M}}{t^2} + \sum_{m \in C} \omega_i \mathbf{S}_m^T \mathbf{S}_m)$ is constant when there is no topology changes. Thus, topology changed method has great influence on the efficiency of the solving process.

Assuming a mesh is composed of n vertices, a constraint $C_m \in \mathbf{C}$ which contains k vertices so $\mathbf{S}_m \in \mathbf{R}^{3k \times 3n}$, $\mathbf{S}_m^T \mathbf{S}_m$ is a diagonal $\mathbf{R}^{3n \times 3n}$ matrix. The non-zero entries represent the the vertices involved in constraint C_m. Thus, the entries of $\sum_{C_m \in C} \omega_i \mathbf{S}_m^T \mathbf{S}_m$ represents the times a vertex has been involved in the constraint. For our method, what we need to do is to find the dissected edges and elements, ignoring the row and columns of matrix represent those deleted vertices and modifying corresponding entries of $\sum_{m \in C} \omega_m \mathbf{S}_m^T \mathbf{S}_m$ by subtracting the times each entry has been involved in the deleted constraints. We do not change the matrix size. For the topology-changed method, the size of matrix is needed to be changed due to new vertices. Also, the relationship between vertices has changed, so $\sum_{C_m \in C} \omega_m \mathbf{S}_m^T \mathbf{S}_m$ needs to be computed again, making the solving of this linear system very inefficient.

7.3 Experiments

In figure 7, we simulate the fascia using the following parameters: Strain limiting parameters: $t_{00} = 0.1, t_{11} = 0.1$. Energized tool temperature: 170°C. Ignition temperature:160 °C. Shrink ratio:$S_x = 0.1, S_y = 0.3$. Thermal conductivity parameter: 0.4. The fascia mesh is composed of $6K$ triangles, running at 103.2 fps.

For the fatty tissue dissection, we use the following parameters: Strain limiting parameters: $t_{00} = 0.3, t_{11} = 1, t_{22} = 1$. Energized tool temperature: 90°C. Ignition temperature:78 °C. Shrink ratio:$S_x = 0.1, S_y = 1, S_y = 1$. Thermal conductivity parameter: 0.6. Elastic limit: 0.00054. Plastic limit: 0.16. The fatty tissue is composed of $30K$ tetrahedrons and the membrane is composed of $6K$ triangles. The average fps is 15.1 fps. For the energized tool, the thermal diffusivity we used is $4.210^{-6} m^2/s$.

8 Conclusion and Future Works

In this paper, we proposed an energized soft tissue dissection model based on heat transfer and physically based simulation strategies for three kinds of soft tissues. This represents a step forward comparing with the exiting research in surgical simulation. A simplified heat transfer approach was used in our current work in order to favour computing efficiency, which can be improved by incorporating more accurate model [29]. Our implicit shape representation for the heat source of the dissection tool is not accurate enough. More accurate shape representation can be designed to approximate different tools.

9 Acknowledgement

We thank Prof.Ladislav Kavan and Tiantian Liu for the helpful discussions. This work is partially supported by National Natural Science Foundation of China (61402374), EU project AniNex (FP7-IRSES-612627), Dr.Inventor (FP7-ICT-2013.8.1 611383), Centre for Digital Entertainment which is an EPSRC funded centre for doctoral training.

References

[1] SM Shimi. Dissection techniques in laparoscopic surgery: a review. *Journal of the Royal College of Surgeons of Edinburgh*, 40(4):249–259, 1995.

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

[3] Matthias Müller, Jos Stam, Doug James, and Nils Thürey. Real time physics: class notes. In *ACM SIGGRAPH 2008 classes*, page 88. ACM, 2008.

[4] Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, and Mark Carlson. Physically based deformable models in computer graphics. In *Computer Graphics Forum*, volume 25, pages 809–836, 2006.

[5] 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.

[6] Eftychios Sifakis and Jernej Barbic. Fem simulation of 3d deformable solids: a practitioner's guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 Courses*, page 20. ACM, 2012.

[7] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. In *Proceedings of the Third Workshop on Virtual Reality Interactions and Physical Simulations, VRIPHYS 2006*. [7], pages 71–80.

[8] Kun Qian, Junxuan Bai, Xiaosong Yang, JunJun Pan, and Jianjun Zhang. Virtual reality based laparoscopic surgery simulation. In *Proceedings of the 21st ACM Symposium on Virtual Reality Software and Technology*, pages 69–78, 2015.

[9] Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. Projective dynamics: fusing constraint projections for fast simulation. *ACM Trans. Graph.*, 33(4):154:1–154:11, 2014.

[10] Jeffrey Berkley, George Turkiyyah, Daniel Berg, Mark Ganter, and Suzanne Weghorst. Real-time finite element modeling for surgery simulation: An application to virtual suturing. *Visualization and Computer Graphics, IEEE Transactions on*, 10(3):314–325, 2004.

[11] Dicko Ali-Hamadi, Tiantian Liu, Benjamin Gilles, Ladislav Kavan, François Faure, Olivier Palombi, and Marie-Paule Cani. Anatomy transfer. *ACM Trans. Graph.*, 32(6):188:1–188:8, 2013.

[12] Geoffrey Irving, Joseph Teran, and Ron Fedkiw. Invertible finite elements for robust simulation of large deformation. In *Proceedings of the 2004 ACM symposium on Computer animation*, pages 131–140.

[13] Jan Bender, Dan Koschier, Patrick Charrier, and Daniel Weber. Position-based simulation of continuous materials. *Computers & Graphics*, 44(0):1 – 10, 2014.

[14] Matthias Müller. Hierarchical position based dynamics. In *Proceedings of the Fifth Workshop on VRIPHYS 2008, France, 2008.*, pages 1–10.

[15] Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG)*, 33(4):153, 2014.

[16] Jun Wu, Rüdiger Westermann, and Christian Dick. A survey of physically based simulation of cuts in deformable bodies. *Comput. Graph. Forum*, 34(6):161–187, 2015.

[17] Eftychios Sifakis, Kevin G Der, and Ronald Fedkiw. Arbitrary cutting of deformable tetrahedralized objects. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 73–80, 2007.

[18] Daniel Bielser, Pascal Glardon, Matthias Teschner, and Markus Gross. A state machine for real-time cutting of tetrahedral meshes. *Graphical Models*, 66(6):398–417, 2004.

[19] Denis Steinemann, Matthias Harders, Markus Gross, and Gabor Szekely. Hybrid cutting of deformable solids. In *Virtual Reality Conference, 2006*, pages 35–42. IEEE, 2006.

[20] Jun Wu, Christian Dick, and Rüdiger Westermann. Interactive high-resolution boundary surfaces for deformable bodies with changing topology. In *Proceedings of the 8th Workshop on Virtual Reality Interactions and Physical Simulations, VRIPHYS 2011*, pages 29–38, 2011.

[21] Jun Wu, Rüdiger Westermann, and Christian Dick. Real-time haptic cutting of high resolution soft tissues. *Studies in Health Technology and Informatics (Proc. Medicine Meets Virtual Reality 2014)*, 196:469–475, 2014.

[22] Jun Wu, Christian Dick, and Rüdiger Westermann. A system for high-resolution topology optimization. *IEEE Trans. Vis. Comput. Graph.*, 22(3):1195–1208, 2016.

[23] 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. *Journal of Visualization and Computer Animation*, 26(3-4):321–335, 2015.

[24] JunJun Pan, Junxuan Bai, Xin Zhao, Aimin Hao, and Hong Qin. Dissection of hybrid soft tissue models using position-based dynamics. In *Proceedings of the 20th ACM Symposium on Virtual Reality Software and Technology, VRST 2014*, pages 219–220.

[25] Oleksiy Busaryev, Tamal K Dey, and Huamin Wang. Adaptive fracture simulation of multi-layered thin plates. *ACM Transactions on Graphics (TOG)*, 32(4):52, 2013.

[26] Hoehn Katja Marieb, Elaine Nicpon. *Human anatomy and physiology*. Pearson Education, 2007.

[27] James F O'brien, Adam W Bargteil, and Jessica K Hodgins. Graphical modeling and animation of ductile fracture. *ACM transactions on graphics (TOG)*, 21(3):291–294, 2002.

[28] Fernando J Kim, MF Chammas Jr, Gewehr, E, and Morihisa. Temperature safety profile of laparoscopic devices: Harmonic ace (ace), ligasure v (lv), and plasma trisector (pt). *Surgical endoscopy*, 22(6):1464–1469, 2008.

[29] Nicolas Maréchal, Eric Guérin, Eric Galin, Stéphane Mérillou, and Nicolas Mérillou. Heat transfer simulation for modeling realistic winter sceneries. In *Computer Graphics Forum*, volume 29, pages 449–458, 2010.

Appendix A

We denote the deformation gradient $\mathbf{F}$ as $\mathbf{F} = (\mathbf{f}_1, \mathbf{f}_2)$, $\mathbf{D}_m = (\mathbf{d}_1, \mathbf{d}_2)$, $\mathbf{d}_1 = (d_{1x}, d_{1y})^T$, $\mathbf{d}_2 = (d_{2x}, d_{2y})^T$, where $\mathbf{f}_i, \mathbf{d}_i$ are $\mathbf{R}^{2\times 1}$ vectors ($i = 1, 2$). For $a, b = 1$ or 2, the entries of $\mathbf{K}$ in equation 11 can be expressed as: $k_{ab} = \mathbf{f}_a^T \mathbf{f}_b = (\mathbf{D}_s \mathbf{d}_a)^T (\mathbf{D}_s \mathbf{d}_b)$. $\nabla_{\mathbf{x}_j} \mathbf{C}_{m_{ab}} = \nabla_{\mathbf{x}_j} (\mathbf{f}_a^T \mathbf{f}_b) = \mathbf{f}_b d_{1x} \otimes \mathbf{I}_3 + \mathbf{f}_a d_{2x} \otimes \mathbf{I}_3$ and $\nabla_{\mathbf{x}_k} \mathbf{C}_{m_{ab}} = \nabla_{\mathbf{x}_k} (\mathbf{f}_a^T \mathbf{f}_b) = \mathbf{f}_b d_{1y} \otimes \mathbf{I}_3 + \mathbf{f}_a d_{2y} \otimes \mathbf{I}_3$, which can be represented as:

$$\begin{bmatrix} \nabla_{\mathbf{x}_j} \mathbf{C}_{m_{ab}} \\ \nabla_{\mathbf{x}_k} \mathbf{C}_{m_{ab}} \end{bmatrix} = \begin{bmatrix} \mathbf{d}_1 & \mathbf{d}_2 \end{bmatrix} \otimes \mathbf{I}_3 \begin{bmatrix} \mathbf{f}_b \\ \mathbf{f}_a \end{bmatrix} \tag{22}$$

Then $\nabla \mathbf{x}_i \mathbf{C}_{m_{ab}} = -(\nabla_{\mathbf{x}_j} \mathbf{C}_{m_{ab}} + \nabla_{\mathbf{x}_k} \mathbf{C}_{m_{ab}})$.

Appendix B

Assuming vertex $\mathbf{x}_u = (x_u^{(1)}, x_u^{(2)}, x_u^{(3)})^T$. $\nabla_{\mathbf{x}_u} \mathbf{E}_{\tau_n} = (\nabla_{x_u^{(1)}} \mathbf{E}_{\tau_n}, \nabla_{x_u^{(2)}} \mathbf{E}_{\tau_n}, \nabla_{x_u^{(3)}} \mathbf{E}_{\tau_n})^T$, $\nabla_{\mathbf{x}_u} \mathbf{E}_{\tau_n} \in \mathbf{R}^{3\times 1}$

$$\nabla_{x_u^{(j)}} \mathbf{E}_{\tau_n} = \mathbf{P}(\mathbf{F}) : \mathbf{e}_j \mathbf{e}_u^T \mathbf{D}_m^{-1} = tr(\mathbf{e}_j^T \mathbf{P}(\mathbf{F}) \mathbf{D}_m^{-1} \mathbf{e}_u) \tag{23}$$

$\mathbf{e}_j^T \mathbf{P}(\mathbf{F}) \mathbf{D}_M^{-1} \mathbf{e}_u$ is a scalar, so it can be taken out of the tr as:

$$\nabla_{x_u^{(j)}} \mathbf{E}_{\tau_n} = \mathbf{P}(\mathbf{F}) : \frac{\partial \mathbf{F}}{\partial x_u^{(j)}} = \mathbf{e}_j^T \mathbf{P}(\mathbf{F}) \mathbf{D}_m^{-1} \mathbf{e}_u \tag{24}$$