密级：_____

中国科学院大学
University of Chinese Academy of Sciences

# 硕士学位论文

## 基于GPU的非线性有限元柔体形变仿真模拟


作者姓名：郭煜

指导教师：王平安　教授

香港中文大学

学位类别：工程硕士

学科专业：计算机技术

研究所：中国科学院深圳先进技术研究院



2013年5月

# GPU-based Soft Body Deformation Simulation with Nonlinear Finite Element Method

By

Yu Guo

A Thesis Submitted to

The University of Chinese Academy of Sciences

In partial fulfillment of the requirement

For the degree of

Master of Engineering

Shenzhen Institutes of Advanced Technology

Chinese Academy of Sciences

May,2013

# 声　　明

我声明本论文是我本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，本论文中不包含其他人已经发表或撰写过的研究成果。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

作者签名：　　日期：

# 论文版权使用授权书

本人授权中国科学院深圳先进技术研究院可以保留并向国家有关部门或机构送交本论文的复印件和电子文档，允许本论文被查阅和借阅，可以将本论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编本论文。

（保密论文在解密后适用本授权书。）

作者签名：　　导师签名：　　日期：

# 摘　　要

柔体结构是一种非线性、各向异性、近似不可压缩的粘弹性材料体。其中的软组织变形仿真是虚拟手术系统中最重要的研究方向之一，同时也是难点之一。本文提出了一种非线性柔体形变的有限元方法，并对其进行了仿真。

本文首先对虚拟手术做了大致的介绍，然后重点分析和总结了软组织形变的各种方法。从几何方法到物理方法，从网格模型到无网格模型，并列举了最近几年的研究情况。其中质点-弹簧模型和有限元模型是最基础也是最常见的方法，其他的很多方法都是以此为基础产生的。其次讨论和实现了有关网格的预处理方法，其中包括表面网格的简化、表面网格的细分以及四面体网格的剖分。采用 QEM 算法对原始冗余网格进行简化，用 Loop 细分的方法对网格进行细化光滑，用Delaunay四面剖分的方法将物体划分空间单元用于有限元计算。接着针对非线性的形变问题，进行有限元分析，提出了两种方法分别将几何非线性和物理非线性两个不能直接求解的难题转化成简单迭代问题。用中心差分的显示形式得到位移的迭代方程。然后详细说明了该非线性有限元方法如何在 CUDA 上面实现，并通过 OpenGL 将结果快速绘制出来。同时还提出了一种快速碰撞模型，不仅使物体发生变形，还模拟了不同的碰撞摩擦效果。此外用变形实验数据和 ABAQUS 的结果做比较，得到十分相近的结果。并分析了不同参数下面的变形情况，以证明本文方法的可靠性和稳定性。最后对本文进行了总结以及对未来发展方向进行了展望。

**关键词：**虚拟手术，软组织形变，有限元方法，GPU

**GPU-based Soft Body Deformation Simulation with Nonlinear Finite Element Method**

Yu Guo (Computer Science)
Directed by Professor Heng Pheng-Ann

Human soft tissues generally exhibit complex material perperties such as nonlinearity, anisotropy, incompressibility and viscoelastictity. Soft tissue deformation is one of the most important yet difficult research tasks in virtual surgery.

This paper presents a comprehensive nonlinear simulation of soft tissue deformation in virtual surgery. We first give an introduction of the virtual surgery system and summarize the various methods of soft body deformation. From geometrically-based methods to physically-based methods, from mesh-based models to meshless models, we give a detailed research situation in recent years. Finite Element Method (FEM) model turn to be the best model of deformation and lots of other method are based on them. Then we give an introduction of mesh precondition which include simplification of the surface mesh, subdivision of surface and tetrahedralization of the soft body. Quadric Error Metric based method is used to simply the redundance surface mesh. Loop Subdivision method is used to make the surface more meticulous and smooth. Constrained Delaunay Tetrahedralizaion method used for meshing the object to later use in FEM. In the finite element method processing, we propose two tricks for the nonlinear deformation of soft body with hyperelastic material. The tricks transform the hard calculating problem about geometry and physical nonlinear to the linear iteration. To use central difference equation to achieve explicit iteration for the displacement of nodes. Then we illustrate how could we achieve the method in the GPU with CUDA API and OpenGL API. As the same time, a fast contact model has been brought out, with this contact method, the soft body perform different kinds of friction deformation. After that, we give an efficient simulation and a fast rendering. Compared the results from ABAQUS and our method, a nice matched curve can be found. Our experiments show that our system is efficient and well present the stability of the deformation. Finally, we make a summary and describe some promising research directions on this topic.

**Keywords:** Virtual surgery, Soft body deformation, Finite element method (FEM), GPU

# 目　　录


**摘要** **I**

**目录** **V**

**图目录** **IX**

**表目录** **XI**

**第一章　绪论** **1**

1.1　虚拟手术系统结构 . . . . . 1

1.1.1　组织器官的三维重建 . . . . . 2

1.1.2　碰撞检测 . . . . . 2

1.1.3　力反馈交互 . . . . . 3

1.1.4　GPU 加速、显示 . . . . . 3

1.2　软组织变形的各种方法 . . . . . 3

1.2.1　几何方法 . . . . . 4

1.2.2　物理方法 . . . . . 4

1.2.3　物理方法小结 . . . . . 10

1.3　本文的研究工作 . . . . . 10

1.4　本文结构安排 . . . . . 11

**第二章　网格处理** **13**

2.1　表面网格简化 . . . . . 13

2.1.1　网格简化算法 . . . . . 14

2.1.2　网格简化结果 . . . . . 15

2.2　表面网格细分 . . . . . 16

2.2.1　网格细分方法 . . . . . 16

2.2.2　网格细分结果 . . . . . 17

2.3　体网格剖分 . . . . . 18

2.3.1　网格剖分方法 . . . . . 19

2.3.2 网格剖分结果 . . . . . 20
2.4 本章小结 . . . . . 21

**第三章 非线性有限元形变模型** **23**
3.1 有限元方法概述 . . . . . 23
3.2 有限元方程的建立 . . . . . 23
3.2.1 力学基本概念 . . . . . 24
3.2.2 应变和应力 . . . . . 25
3.2.3 位移和应变的关系：几何方程 . . . . . 26
3.2.4 应变和应力的关系：物理方程 . . . . . 28
3.2.5 应力和外力的关系：平衡方程 . . . . . 31
3.3 有限元方程的求解 . . . . . 31
3.3.1 直接刚度矩阵法 . . . . . 31
3.3.2 间接刚度矩阵法 . . . . . 33
3.3.3 中心差分的显式积分法 . . . . . 34
3.4 本章小结 . . . . . 35

**第四章 非线性有限元方法的具体实现** **37**
4.1 OpenGL和CUDA简介 . . . . . 37
4.2 有限元算法在 GPU 上的实现 . . . . . 38
4.2.1 显式迭代的并行化 . . . . . 38
4.2.2 Kernel1：单元循环 . . . . . 39
4.2.3 Kernel2：节点循环 . . . . . 39
4.3 快速碰撞模型 . . . . . 41
4.4 CUDA-OpenGL 互操作 . . . . . 43
4.5 算法流程 . . . . . 44
4.6 本章小结 . . . . . 44

**第五章 实验结果分析** **47**
5.1 变形过程的模拟 . . . . . 48
5.2 准确性评估 . . . . . 49
5.3 不同材质的模拟分析 . . . . . 51
5.4 不同碰撞模型的模拟分析 . . . . . 53

5.5 网格剖分程度对变形的影响 . . . . . 54
5.6 计算效率 . . . . . 55
5.7 本章小结 . . . . . 55

**第六章 总结与展望** **57**
6.1 本文工作总结 . . . . . 57
6.2 今后研究展望 . . . . . 57

**参考文献** **59**

**致谢** **i**

**作者简历** **iii**

# 图 目 录


1.1 虚拟手术系统结构简图 . . . . . 2
1.2 二维质点弹簧模型（三角形单元） . . . . . 5
1.3 二维有限元模型（三角形单元） . . . . . 6

2.1 网格操作流程 . . . . . 13
2.2 网格简化方法 . . . . . 14
2.3 Stanford 兔子简化 . . . . . 15
2.4 人的大脑简化 . . . . . 15
2.5 人的牙齿简化 . . . . . 15
2.6 Loop 细分方法中几何点的生成规则 . . . . . 16
2.7 Stanford 兔子细分 . . . . . 17
2.8 人的大脑细分 . . . . . 18
2.9 人的肾脏细分 . . . . . 18
2.10 Delaunay 准则的空球特性 . . . . . 19
2.11 圆环剖分 . . . . . 20
2.12 细分后圆环的剖分 . . . . . 20
2.13 肾脏的剖分 . . . . . 20
2.14 胃的剖分 . . . . . 21
2.15 肝的剖分 . . . . . 21

3.1 $\Omega$ 从 $K_0$ 变形为 $K_t$ . . . . . 25

4.1 整体内力矩阵的合成 . . . . . 40
4.2 不同的四面体剖分结构 . . . . . 40
4.3 内投影碰撞模型，${}^{t+\Delta t}\mathbf{x}$ 是节点的最终碰撞位置 . . . . . 42
4.4 外投影碰撞模型，${}^{t+\Delta t}\mathbf{x}$ 是节点的最终碰撞位置 . . . . . 42

5.1 圆环体变形比较 . . . . . 47
5.2 圆环下落碰撞的变形模拟 . . . . . 48
5.3 $x$ 方向的位移 . . . . . 49

5.4 $y$ 方向的位移 . . . . . 50
5.5 $z$ 方向的位移 . . . . . 50
5.6 整体位移 . . . . . 51
5.7 本文方法与 ABAQUS 所得结果的位移偏差 . . . . . 51
5.8 不同杨氏模量的圆环产生的最大变形 . . . . . 52
5.9 泊松比对变形结果的影响 . . . . . 52
5.10 时间步长与泊松比和杨氏模量的关系 . . . . . 53
5.11 两种不同的摩擦效果 . . . . . 53
5.12 网格剖分程度不同的模型仿真结果与 ABAQUS 结果比较 . . . . . 54
5.13 圆环变形过程 . . . . . 55
5.14 龙模型 . . . . . 56
5.15 胃模型 . . . . . 56

# 表 目 录

3.1 不同组织器官的物理参数 . . . . . 35

4.1 有限元变形算法步骤 . . . . . 44

5.1 GPU 算法单次迭代中各项数据计算时间 . . . . . 55

# 第一章　绪论

随着计算机技术的发展，虚拟现实和医学物理的融合交叉，外科诊断治疗手段正在发生巨大的变化。近年来出现的例如计算机辅助手术系统，虚拟外科手术系统等，就是信息科学应用于医学领域而产生的成果。外科医生通过这些先进的技术手段在术前、术中、术后对手术进行辅助支持。使外科手术越来越安全、可靠、精确，创伤越来越小。

一个完整的虚拟手术系统，具有以下两个特点：虚拟场景复杂、人机交互性强。其所涉及的技术也非常广泛，涉及到多学科交叉，是一项综合性的研究方向。虚拟手术中手术交互操作有：触摸、按压、提拉、抓取、切割、撕扯、剪断、针穿刺、灼烧、缝合、打结等；同时也需要模拟手术过程中的流血、灼烧烟雾等特殊效果，需要用到的技术有：医学数据可视化、计算机图形学、牛顿动力学、生物力学、连续介质力学、流体力学、数值计算、物理建模与几何算法等领域，对计算机编程能力要求极高，是一项研究难度很大的课题。

生物组织一般是固态和液态两种形态构成，并且表现出不同程度的粘弹性。医学中固体分为硬组织（比如软骨、骨骼、头发和指甲等）和软组织（如皮肤、内脏、血管、气管、肌肉等）。硬组织通常不发生形变；而软组织在物理行为上表现出明显的非线性形变特性。因此人体软组织模拟难度较大，通常采用软组织的弹性模量、阻尼系数、密度等物理特性表征软组织的粘弹性、各向异性、非均匀性等特性。手术交互操作中最基本的就是软组织形变。触摸、按压等操作属于弹性形变的范畴，不会改变组织的固有结构，主要需要研究的就是形变特性，是手术仿真中最基本的交互方式。切割、撕扯等属于非线性形变，这种操作使软组织的几何结构发生改变。涉及到拓扑结构重组、断裂力学理论等，仿真难度较大，但基础理论还是软组织形变。

## 1.1　虚拟手术系统结构

虚拟手术整体系统结构简图 1.1 可以看出，这是一个系统较为完备，涵盖多学科的技术系统。大致分为以下几个层面：组织器官的三维重建，人机交互，效果显示，GPU 加速，以及衔接这几个层面的计算层。其中最核心的就是软组织变形，这一问题顺利解决了才能使整个系统运作起来。这章简单介绍其他几个层面的工作，之后着重介绍软组织变形的各种方法。

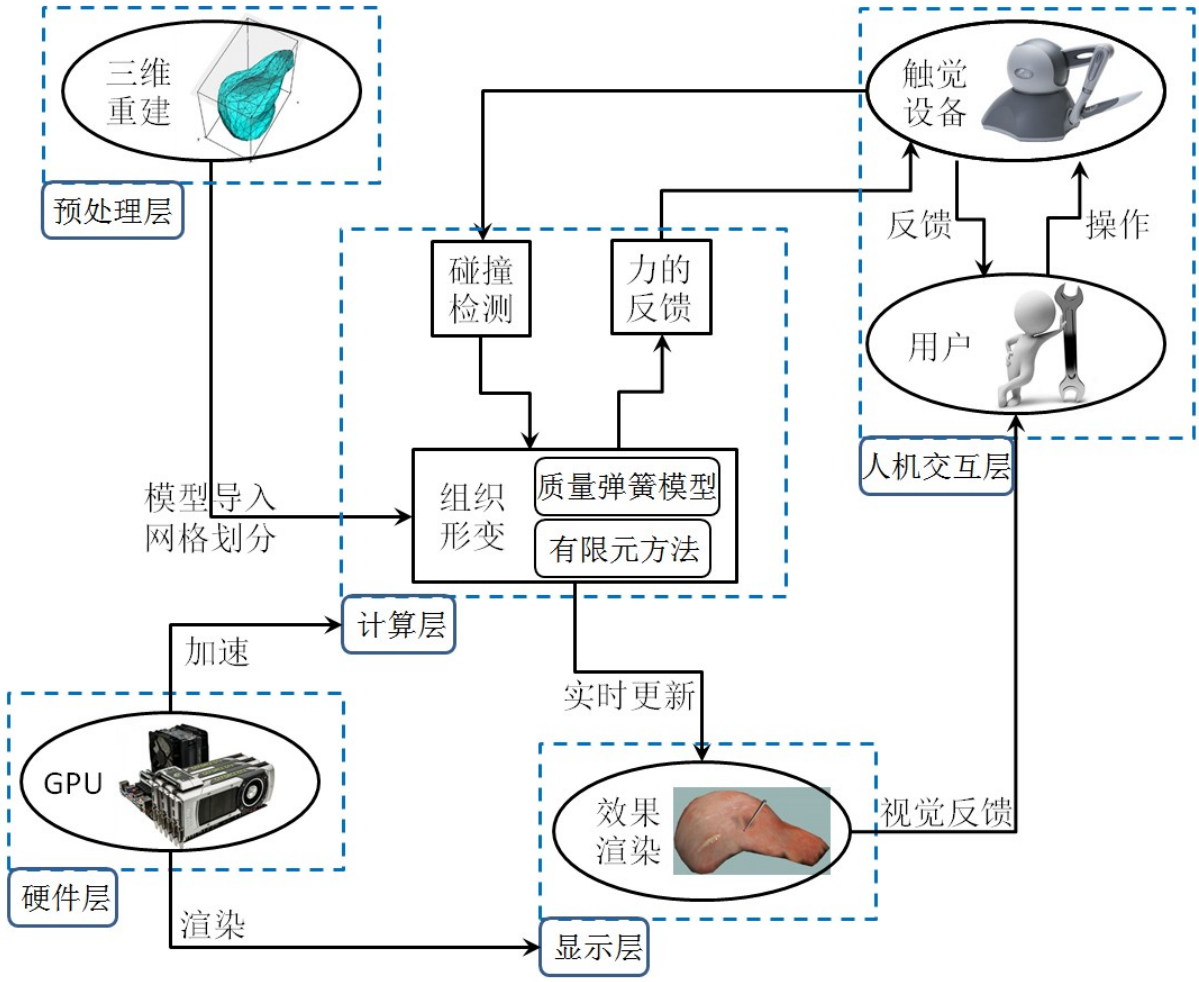



图 1.1: 虚拟手术系统结构简图

### 1.1.1 组织器官的三维重建

人体组织的获取一般是通过 CT 、核磁共振、超声、切片扫描等方法获得，他们的共同特点是把通过二维图像建立三维模型。常用的步骤有重采样、旋转图像、配准、分割、平滑处理、模型制作等。

### 1.1.2 碰撞检测

软组织变形由两种因素导致，一种是身体自身的充气以及心脏跳动引起的自然变形；一种是器官之间相互碰撞挤压以及手术工具的接触，多见于虚拟手术。碰撞检测是有限元变形中外力部分，同时也是力反馈的成因，所以是非常重要的一部分。Stavness 等[1]分析了下颌、舌头与舌骨的碰撞过程，其中包括了软组织和硬组织的碰撞情况。 Johnsen 等[2] 在有限元基础上成功仿真了肝脏与横膈膜之间的碰撞情况。Hirota 等[3]则在可视人数据集平台上做了碰撞变形的模拟。

### 1.1.3　力反馈交互

力反馈的难点是实现多种人机交互模拟。挤压是变形的最简单形式，此外还有针刺、切割等需要破坏模型拓扑结构的操作。系统在运行时使用具有力反馈功能的输入设备模拟手术器械，首先通过输入设备从用户处获得包含速度和位移的输入信息，然后进入碰撞检测阶段。当物体发生形变之后，再计算出交互力传回力反馈设备，让操作者感受到力的反馈。 Wang 使用用力反馈设备，成功模拟了骨骼手术中的“锯”[4]和“磨”[5]两种效果，并且在实时交互的基础上通过实际病人的手术数据验证了模拟的真实性。同时在该平台下还完成了实时下颌角整形手术的模拟[6]。

### 1.1.4　GPU 加速、显示

计算机硬件的飞速发展，越来越多的医学计算开始依靠图形处理器 (Graphics Processing Units, GPU) 的强大并行能力[7]。由于计算量的原因，之前模拟形变的软件大多属于离线计算，只能对变形过程中的某一瞬间结果进行显示，不能实现全程可交互的显示。正因为强调的是人的互动，我们的目标不是变形的最终结果，而是实时的交互变形。所以快速显示是一个很重要的课题。近几年来，随着计算机显卡的发展，模型形变计算大多会依靠 GPU 。模型数据已经存储到了显存，那么做显示的时候，利用显卡的新特性直接从显存把变形模型显示出来，减少数据传输，提高显示速度。 Li 等[8]指出，鉴于目前不同 NVIDIA 显卡的不同特性，形变计算和显示通常不在一张显卡中进行，他们利用 GPUDirect 技术，实现了快速的显卡间数据传输。

## 1.2　软组织变形的各种方法

软组织行为的计算是现代生物医学仿真中的核心问题，在过去的一段时间内，人们提出了许多软组织建模的方法。其中最关键的是要权衡实时性和变形的精确性。真实性和实时性这对矛盾是软组织建模中最为主要的问题。当需要高逼真度来实现真实性时，复杂的模型通常会导致计算时间的大幅增加。此外，仿真过程的稳定性以及交互过程中软组织拓扑特性的改变也是需要考虑的问题。

对于新产品的设计，生物医学材料和手术工具的科学分析需要一个比较高的精度，所以，高计算量的离线计算有时候是非常必要的。与此相对的是，在手术训练中这些需求却可以被降低要求。这样的话由于整体精度的降低，我们就可以增加手术训练系统中的可交互性。这样做才更符合我们的实际意义，因为不同病人的组织器官物理属性不完全一样，我们只知道一个大概的标准，所以基于虚拟现实的手术模拟需要实时的变形显示，而变形精度只需要一个较为粗糙的近似就可以了，这被称为准物理过程。所以一般的手术模拟遵循这样一个标准：我们的目标通常是实现训练效果，而不是高度精确的软组织材质属性。不过，在手术模拟中如何用一个变形模型来实现既定的手术训练过程依然是一个没有被完全解决的问题，较大的不准确性依然会导致一个错

误的手术训练结果。因此，建立仿真系统时，选择合适的形变模型是很关键的一个步骤。

真实的器官组织行为是很复杂的，而且我们通常只能了解到其中的一部分行为。正因为软组织的这种复杂特性，找到一个合适的数学模型是非常困难的事情。要达到交互式实时仿真，需要对特定的模型进行简化和优化。此外模型的稳定性和切割变形是后续需要考虑的问题。在选择了一个合适的数学模型去仿真软组织的生物力学特性后，我们还应该去测量和分析真实的软组织变形参数用以修正仿真模型。在模型被完整的表达出来之后，我们也只能用这个模型去做很有限的分析。

### 1.2.1 几何方法

早期模拟物体变形的方法不是采用物理力学分析的方法，而是直接操作模型的几何属性，改变曲面形状从而达到期望的变形效果。通常会直接修改物体节点位置，让我们看起来模型确实发生了形变。几何方法不考虑组织的力学本构关系，也不考虑物体的质量、材质等自身属性，缺乏生物特性。这种变形方法目前多用于动画中，因为动画可以夸张的表现一些不符合实际情况的形变。直接通过控制点来控制物体形状的自由形变技术 (Free-form Deformation, FFD) ，通过控制点来控制曲线曲面的贝塞尔曲线、 B 样条曲线、非均匀有理 B 样条等曲线曲面技术以及蒙皮法（现在也有基于物理模型的蒙皮法[9]）。此外，隐式曲面在模拟比较软的物质比如血液、粘液、粘土比较有效。 Kravtsov 等[10]使用隐式曲面与多边形网格相结合的方法，创造了视觉逼真的非物理动态模型。 Gibson 等[11]曾提出一种非物理模型，名叫锁环甲模型阵 (ChainMail) 。它把物体离散为环环相扣的锁环，每个锁环与周围八个相连，并且为每个锁环设置最大移动距离。当一个发生运动，并且达到最大移动距离，就会带动相邻的锁环。这个模型原理简单，运算速度快。在早些年用这种几何方法可以弥补计算能力的不足，但是缺点还是很明显的。在虚拟手术中，软组织形变不仅仅要满足视觉上的变形效果，更重要的是能都真实的还原生物力学的特性。同时计算机性能的提升也使人们的关注点放到了如何精确模拟软组织变形。因此，目前基于几何建模的方法在虚拟手术中应用较少，已被物理方法取代。

### 1.2.2 物理方法

目前基于物理方法的技术已经日渐成熟，物体动态建模技术从仅仅包含速度、加速度的运动学方程发展成为包括应力、应变的动力学方程。这种进步使得虚拟手术对象具有了可交互性的功能，能够再现真实的行为属性以及更加逼真的虚拟对象。虚拟手术模拟中遇到的软组织变形的问题，相对于刚体结构模型来说，软体结构最主要的区别是在挤压和撕拉的模拟中会有能量损失导致形状的改变，现有的物理方法已经可以把能量的变化考虑其中。

模拟软组织的变形现在有多种较为成熟的方法。它们大致可以分为两类：一种是基于网格结构的方法，另一种是不依靠网格进行模拟。可以看出，如何建立模型结构是选择模拟方法的重要前提。

基于网格的方法主要有质点-弹簧模型和有限元方法，此外还有长单元模型，有限差分法，有限体积法，边界单元法和张量-质点模型。不使用网格的方法主要有粒子系统，有限球方法，光滑粒子流体动力学方法以及无网格方法。

#### 1.2.2.1　网格结构方法

**长单元模型 (Long Element Model, LEM)**

长单元模型是一种基于物理的高度简化的模型，它首先将模型离散化为充满着不可压缩液体的长六面体单元，并在此基础上对满足 Pascal 定理及体积不变性的物体建立求解方程，每个模型的单元数量与边长的平方成比例。由于长六面体的单元结构使得该方法的网格数量比四面体或立方体网格数量少近一个数量级，模型的物理参数可以使用体积变量而不必做预计算、压缩等处理，保证了较高的视觉与力觉刷新速度。但是该方法对模型进行了高度的抽象和简化，因此计算精度较低，目前主要适用于模拟实时动画，但对软组织建模也有一些应用。

**质点-弹簧模型 (Mass-Spring Model, MSM)**

质点弹簧模型是把变形模型的质量节点用弹簧连接起来（见图 1.2 ），当一个弹簧试图伸长，周围其他弹簧就限制它让其有一种收缩的趋势，反之亦然。经过一段的时间，整个弹簧系统达到动态平衡，或者说软组织形变的过程就是一种向平衡过度的过程，直到被外界的外力打破这种平衡，继而再次寻找平衡的过程。

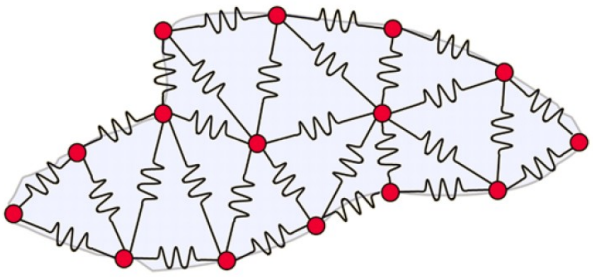


图 1.2: 二维质点弹簧模型（三角形单元）

当一个力作用到某个质量节点时，弹簧长度由 $l^0$ 变为 $l$，弹性系数为 $k$，根据胡克定律 $f = k(l^0 - l)$，在质量点 $i$ 处的弹簧弹力可以由他周围的 $j$ 点对其共同作用而成：

$$\mathbf{f}_i = \sum_j k_{ij}(\|\mathbf{x}_j - \mathbf{x}_i\| - l_{ij}^0)\frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_j - \mathbf{x}_i\|} \tag{1.1}$$

自从 Terzopoulos [12]的开创性工作之后， MSM 在手术仿真中的应用越来越广。我们很容易得到系统的解，因为运动方程不需要显示构造。 MSM 是一种离散的方法，对于一个弹性体来说，他只能近似的估计真实的物理情况。而且对于真实的组织模型物理参数在这里是不能直接用的，需要我们进行人工调整，同时物理过程的仿真结果和质量点的选择有很大的关联，建立三维弹簧的最优网格拓扑结构是一个很复杂的过程，网格的优劣与数量对精度和模型的动态稳定性有较大的影响。然而拓扑结构的改变却很难能影响到 MSM 方法。 San-Vicente 等[13]提出了基于拉力形变和非线性材料的 MSM 方法，专门处理类似于软组织的超弹性材料，得到的结果比传统 MSM 要更加精确。 Etheredge [14]给出了 MSM 模型的并行算法，并对肺部和胳膊的变形进行交互式模拟，实现了快速 MSM 方法。 Hammer 等[15]对心脏瓣膜的形变模拟运用了 MSM 方法，在结构规则的时候，准确性和有限元方法相近，但是遇到剪切模拟时，应变的错误率会快速的增加。 Qin 等[16]提出了硬件加速的基于 MSM 的多层次软组织变形模拟框架，在中国可视人 (Chinese Visible Human, CVH) 数据集上面建立了一套可交互的虚拟手术训练系统。

**有限元方法 (Finite Element Method, FEM)**

有限元方法是求解弹性力学问题的经典方法。将其应用于虚拟手术，并通过恰当的参数调节，可以构造出几何与物理上都较为精确的模型。有限元法的基本思想是将连续的求解区域离散为一组有限个、且按一定方式相互联结在一起的单元的组合体（见图 1.3 ）。由于单元能按不同的联结方式进行组合，且单元本身有可以有不同形状，因此可以精确地模型化几何形状复杂的求解域。在虚拟手术中，用有限元法模拟器官形变的基本思想是：通过推导得到一个模型所有结点处外力与位移关系的方程组，然后实时代入不断变化的已知外力与位移，进行方程组的求解。有限元方法针对连续的模型提供了很严密的软组织物理模拟，同时考虑到了边界情况。在这个处理过程中，结构模型被划分为一定数量的有限单元结构，三维模型被划分成四面体或者六面体单元，二维模型被划分为三角形或者四边形单元，每个单元的位移和位置都是由节点值插值而得到。 Foteinos 等[17]给出了多组织网格模型的生成方法，该方法生成的网格具有良好的几何结构，同时不同组织之间的接触面也做了很好的网格处理，非常适合进行有限元处理。

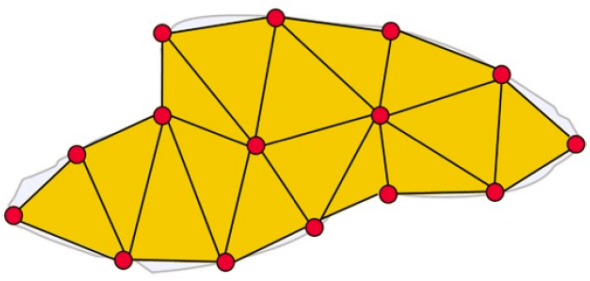


图 1.3: 二维有限元模型（三角形单元）

FEM 的一个优点是在描述这个物理系统时，我们只需要较少的材料参数，这些参数可以直接从软组织测量中得到并整合到计算中。然而较长的计算时间对于实时性的要求难以达到。如果把切割过程加入有限元过程，那么将需要修改网格的拓扑结构，网格模型在模拟过程中也需要被不断的重新计算，计算量不断加大。预计算以及模型简化将是解决这一问题的可行手段。

对于有限元方法中的四面体网格，常采用的 Lagrange 方法已经被广泛应用。西澳大学医学智能系统实验室的 Karol Miller 小组提出用全局拉格朗日显示动力学 (Total Lagrange Explicit Dynamic, TLED) 方法计算大脑形变[18]，随后 Zeike Taylor 小组把 TLED 方法在 GPU 上面实现加速[19]，之后不断改进，成功的把模型降维[20]，减小了计算量，目前已成为一个比较热门的方法。随后 Joldes 等[21]对比了不同网格结构的划分在 GPU 上运行的效率，他们的实验结果对有限元模型的选择起着宝贵的参考意见。Noe [22]运用该方法成功模拟了放射疗法中三维网格的配准。 Niroomandi 等[23]用该方法对肝脏器官进行了仿真。

**边界元方法 (Boundary Element Methods, BEM)**

和 FEM 比较类似的另一种数值求解方法就是边界元法。该方法只需要在求解域的边界处采用分段多项式近似的方法对控制方程离散化，通过 Green-Gauss 法则将运动方程转换为表面积分表达式，因此与 FEM 不同之处就在于 BEM 只对模型的边界进行离散，进而使计算由于降低了问题求解的自由度而得到了简化。最早将边界元法用于形变物体仿真的时候是基于 BEM 建立了准静态形变模型，并通过采用简单的结构再分析技术实现了实时的仿真特性，对一些简单的模型有一定的优势。但是此类方法只限定在对内部为线性的、各项同性均质的物体建模，对一些复杂的、具有丰富细节的模型失真较大；在处理拓扑逻辑结构变化的问题时比 FEM 方法更困难，不适合虚拟手术切割建模，但是在处理手术器械尖端力时[24]，却要比 FEM 法表现出更好的精确性。另外， BEM 还无法表达体数据，而在很多工程应用中，物体内部的力及位移是很重要的，这也是边界元法的缺点之一。

**张量-质点模型 (Tensor-Mass Model, TMM)**

张量-质点模型是 FEM 的一种简化，它把对象实体内部离散化为一系列的四面体有限单元，四面体的各顶点作为质点，四面体的边作为弹簧，与 FEM 不同的是 TMM 不用组装整体刚度矩阵，而是把各单元刚度矩阵分离后累加到各个顶点和边上。质点所受的作用力由所有与它相邻的四面体及四面体的顶点共同决定，将对象的质量及阻尼集中到四面体的顶点上并使用牛顿定律，在对象所受外力和形变之间建立线性或非线性的应力-应变关系。因此，该模型的实质就是利用 FEM 定义应变能函数，然后由应变能函数导出作用在质点上的内力计算公式，最后根据 MSM 建立形变控制方程。形变的计算不再完全依赖所在单元的拓扑结构，因而比 MSM 模型具有更强的体特性的表现能力[25]。

#### 1.2.2.2 网格模型优缺点比较

上述有代表性的物理建模方法都各有优缺点，像非迭代的 BEM 计算速度和鲁棒性好于其他模型，但要涉及复杂的预计算；具有简单、灵活拓扑结构的 MSM 更容易渲染，但稳定性不佳；高精度的 FEM 计算量庞大；具有体描述能力、易于实现大形变的 TMM 要涉及复杂的四面体网格。

总的来说主流的两种方法是：质点弹簧模型和有限元模型。其他方法基本都是这两种方法的衍生品或简化版。

MSM 的最显著的优势就是通常速度很快，足以达到实时计算。此外，在模拟过程中加入几何修正也是不难的事情。缺点是在标准方程中不能保持体积的不变性，因为软组织几乎都是不可压缩的，所以这将会产生不真实结果。而且变形是依靠拓扑结构的，我们很难去选择未知模型的参数。

FEM 的优势在于，它对网格的几何结构以及外部载荷都是独立的，所以用 FEM 可以解决各种不同的复杂情况。而且它还有坚实的理论基础。另一方面，潜在的劣势就是较长的计算时间，尤其是对于非线性有限元。此外，该方法经常难于实行，需要一些经验来定义一些其他问题，比如网格的生成。如果遇到类似于切割的问题，拓扑结构发生了改变， FEM 就会变得比较复杂，通常需要增加重新划分网格这一步骤。

FEM 通常被认为比 MSM 具有更高的精确性，但相反速度方面有很大的劣势。美国南加州大学的 Barbič 教授带领团队开发的非线性 FEM 模拟器 Vega [26]，成功模拟了多种 FEM 模型以及 MSM 模型，并且对这几种方法产生的效果做了比较。在摩尔定律的今天，硬件发展飞速，并行化已成趋势，尤其是 GPU 通用计算的诞生，让快速的 FEM 成为可能。

#### 1.2.2.3 无网格方法

基于网格的方法用来模拟软组织变形已经较为成熟并且应用广泛，但是在手术模拟过程中，经常会遇到血液或者其他近似流体的运动和变形，网格模型的劣势在这里被凸显出来。首先网格的划分是一项很复杂的工作，划分结果的好坏会决定变形的效果。同时，遇到大变形或者拓扑结构的改变，甚至需要不断重新划分。再次，网格结构只适用于连续体，还有很多变形是网格结构所不能完成的。在此情形下，很多人将研究方向转向了无网格结构，也就是用一种更自由的方式来模拟更多样的形变。

**粒子系统 (Particle System)**

一个粒子系统是由大量的随时间变化的粒子作为物体造型的基本单元，每个粒子都有一组物理和时间属性，包括位置、速度、颜色和生命周期等，并在每个生命周期中，都要经历粒子的产生、属性的更新、死亡的消除三个阶段，采用随机过程的方法对这三个阶段中与粒子有关的每一个参数进行随机控制，使得粒子系统所表现的景物具有随机性。因此，粒子系统并不是一个简单的静态系统，其本质是一种随机模型，

此随机模型可根据给定的平均期望值和最大方差确定其属性参数的变化范围。这样的模型可以产生大的变形，所有粒子的位移共同形成了物体的形变。这种方法一般用来模拟没有固定形状的物体，比如火焰，爆炸，烟，云等自然景观，在虚拟手术模拟中一般用来模拟流血的效果。

Müller 等[27]使用定向粒子，储存了旋转以及一些常规的线性属性，比如位置和速度。在常规的粒子系统中加入这些特有属性之后，可以物理模拟更为精确的碰撞效果和带皮类结构的动画显示。 Qian [28]通过构建点云模型来分析应力情况，并对主动脉膨胀以及颅骨撞击进行了仿真模拟。

**有限球方法 (Method of Finite Spheres, MFS)**

有限球法是一种不基于网格的物理建模方法，最早在 2000 年由麻省理工大学的 De 和 Bathe 提出[29]。该方法实际上是无网格局部彼得洛夫-伽辽金法的一种特例，对局部积分方案和边界条件的引入方式进行了改进。后来， De 等[30]将此方法应用于对虚拟微创手术系统的研究中，采用这种基于配置点的有限球方法 (PCMFS) 建立软组织的形变物理模型，由对线性模型的研究逐步深入到非线性模型，并灵活地运用多分辨率技术和快速再分析策略加快运算速度，实现了手术触诊与切割操作的仿真；通过与有限元方法相比较，体现出 PCMFS 较高计算效率。但是该方法实际上是一种基于配点法的无网格技术，必然会遇到不稳定和精度低的问题[31]。

**光滑粒子流体动力学方法 (Smoothed Particle Hydrodynamics, SPH)**

SPH 是最近 30 年出现的一种真正意义的无网格方法，该方法的基本思想是，将连续的流体（或固体）用相互作用的质点组来描述，每个质点上加载各种物理变量，包括质量、速度等，通过求解质点组的动力学方程，并且跟踪每个质点的运动轨道，得到整个系统的力学行为。理论上讲，只要质点的数目足够多，就能精确地描述力学行为。虽然在 SPH 方法中，解的精度也依赖于质点的排列，但它对点阵排列的要求远远低于网格的要求。由于质点之间不存在网格关系，因此它可避免大变形时网格扭曲而造成的精度破坏等问题，并且也能较为方便的处理不同介质的交界面。由于粒子点的运动和流体或者气体的运动非常相似，因而在流体流动中应用较多。该方法对计算资源要求较低，但是对液体表面的跟踪和重构较难，光滑核函数难以选取，求解系统容易是病态的，需要特殊的稳定方案，因此模拟的真实程度、稳定性及精度方面还比不过基于网格的模拟效果。

苏黎世联邦理工学院的 Solenthaler [32]等人提出了基于 SPH 的动量守恒双向耦合方法，用流体动力学来模拟流体以及任意刚体物质。这种方法有利于模型初始化，以及可以实现多个物体的动态模拟。 Akinci 等[33]采用不可压缩 SPH 方法，成功的将边界粒子化，实现了刚体-流体的相互耦合。

**无网格方法 (Meshless Method)**

无网格模型是一种快速的而且很流行的方法用来生成软组织结构。它最大的优势

在于不需要很多计算，所以常被人们用于实时可视化。无网格模型是一种比较稳定的方法，不会在仿真过程中出现奇怪的变化而导致模拟失败。它的结果是简单并且直观的。

无网格法是由机械工程领域提出的一种数值求解方法，该方法基于一组分散点来离散求解区域，不需要网格的初始划分和重构，直接用这些离散点为每一个点构造形函数，从而在没有指定任何连接关系的情况下用求解域上的场函数在这些离散点上的值来逼近或者插值这些函数。由于它弱化了基于网格算法的分片单元的特性，强化了紧支域内的整体误差最小，从而保证了整体计算精度，而且可以减少计算难度。正是由于基于网格的数值方法在处理某些问题时存在的固有缺陷，使得无网格方法得到了迅速发展。

西澳大学的 Jin 等[34]为了克服基于网格方法的缺点，尤其是在切割手术中，由于网格被切开时需要重新划分网格而带来的高计算消耗，提出了自适应无网格 Lagrange 方法 (MTLADR)，不光效率有所提高，精度也不比 ABAQUS 软件计算的结果差。Horton 等[35]尝试用 Lagrange 方法来处理无网格结构，取得一些理论进展。对一些内部不容易划分网格的物体，表面采取网格化，内部采用无网格化，这种大胆的创新结合，也许正是未来发展的一种趋势。 Gills 等[36]更是把无网格结构与基于框架的蒙皮法结合，也就是物理方法与几何方法相结合，得到的这种新的变形方法可以创造稀疏的无网格模型[37]，并适用于多种本构模型的大位移以及大变形。

### 1.2.3 物理方法小结

基于物理的建模方法倍受欢迎，一方面因为物理方法具有遵循物体客观运动规律的特点，所以该方法更能逼近真实的形变等动态效果；另一方面，在一些复杂的形变和运动分析中，此类模型依赖于真实物体的属性，更易于获取参数并进行明确的设定，增强了对形变控制的灵活性与便利性；更重要的是，虚拟手术训练系统的反馈力可以直接获得或者由公式推导得到，同时可以融入实际对象物理属性；而在非物理的模型中，反馈力需要单独计算，且与对象真实属性无关。此外，由于物理模型与工程领域的仿真具有一致性，从而更加容易与虚拟医学辅助诊断等工程系统融合。

## 1.3 本文的研究工作

本文重点介绍了非线性软组织形变的有限元计算这一关键技术问题，并从事解决在实现虚拟手软组织形变的过程中所遇到的相关问题。

本文首先介绍了虚拟手术平台的整体框架设计，平台模块的划分。之后介绍了形变模块的处理方法，详细回顾了当前物理形变过程中的各种模型。接着对网格的处理做了一些实现，包括简化、细分和剖分。本文重点建立了非线性有限元的处理过程，并在 GPU 上完成这一工作。同时在形变过程中，提出一种能快速并且有利于后续形变

计算的碰撞处理方法，为提高整体模拟速度做出贡献。最后通过详细的实验对该算法进行仿真，表明算法的可行性与有效性。

## 1.4　本文结构安排

本文章节结构安排如下：

第一章，绪论。

简要叙述了本课题的研究意义，研究背景以及当前国际从事该课题研究的进展与成果情况。

第二章，网格处理。

介绍了有限元变形的前操作，主要以网格的处理为主。对表面网格简化、表面网格细分以及体网格剖分进行阐述以及实现。

第三章，非线性有限元形变模型。

详细介绍了课题所从事的形变理论的思想，并对现有有限元形变模型进行了详细的介绍，对非线性有限元提出了两个处理办法，是该方法可以更好的在 GPU 上运行。

第四章，非线性有限元方法的具体实现。

本章详细介绍了非线性有限元方法在 GPU 上算法的实现过程，并提出一种适合 GPU 计算的快速碰撞模型。

第五章，实验结果分析。

本章着重介绍了上述章节形变的仿真效果，并对其中的一些细节问题通过实验的方式进行讨论。

第六章，总结与展望。

总结本课题所做的工作，并对后续的工作的进一步研究作了展望。

# 第二章　网格处理

虚拟手术中被操作的软组织模型一般来自于 CT 或者 MRI 的切片图像，然后重建出三维图像数据，经过图像平滑和数学形态学等预处理操作，分割出我们所需要的部分，接着进行表面重建。此时重建出的表面网格冗余程度很高，在保持网格拓扑结构以及原始网格特性的基础上，需要最大程度的减少网格顶点数目同时保持误差最小。网格简化之后需要对网格进行光滑处理，对网格质量进行评估和检测。最后进行面网格的体剖分，将组织模型转化为可以进行有限元计算的体网格单元，见图 2.1 。此外还有一种方法是直接从医学图像中生成体网格，这种方法得到的网格结构不理想，故在此不予考虑。

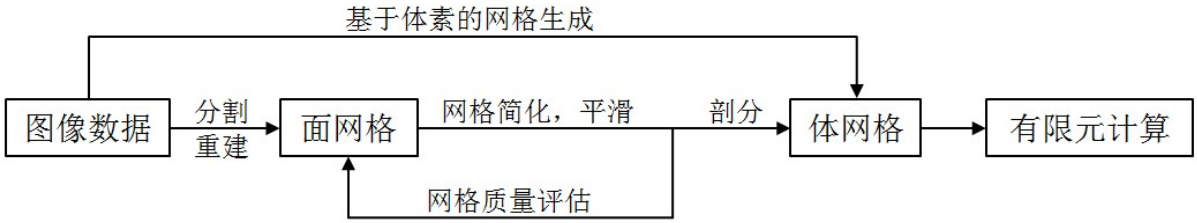



图 2.1: 网格操作流程

为了提高网格计算效率，我们在有限元模拟过程之前进行了网格的简化，但是在有限元模拟过程中，物理形变较大的地方往往需要较密的面网格来实现良好的形变效果。这样我们还需要用到自适应的网格细分算法。

网格的处理，在虚拟手术模拟过程中有这承上启下的作用，是连接医学与物理动力学之间的重要桥梁。在本章节中，讲分别阐述网格简化，网格细分，以及网格划分等方法。

## 2.1　表面网格简化

不论是医学三维 CT数据经过 Marching Cube 的方法重建，还是三维激光/投影扫描设备所采集到的点云数据重建，生成的三角面片网格结构是相当庞大的，基本都在百万数量级以上。这是因为三维扫描的过采样，以及三角化时过度细分造成的。这样的三维模型不用说进行有限元处理，就连划分四面体空间结构都是一件非常困难的事情。然而这其中的三角面片大部分都是冗余的，不管是对于模型本身，还是后续的计算，模型的简化都是很重要的。

由于网格模型大部分由三角面片表示，而且即使原始模型不是三角面片，也可以对其进行三角化，因此网格模型简化的本质是：在尽可能保持原始模型特征的情况下，最大限度地减少原始模型的三角形和顶点的数目。它通常包括两个原则：顶点最少原则，即在给定误差上界的情况下，使得简化模型的顶点数最少，误差最小原则，给定简化模型的顶点个数，使得简化模型与原始模型之间的误差最小。网格简化应用广泛，存在多种算法，按照拓扑结构可分为拓扑结构保持[38, 39]和非拓扑结构保持[40, 41]；根据简化过程可分为逐步求精[42, 43] 和几何简化[44–46]；根据误差控制可分为误差受限[47]和误差不受限[44]；根据视点特性分为视点相关[48, 49]和视点无关[38, 44]等。

### 2.1.1 网格简化算法

Garland [50]的基于二次误差测度 (Quadric Error Metric, QEM) 的网格简化算法，是一种基于边收缩的简化算法。它利用顶点到与其相关联平面距离的平方和作为二次误差估计来度量简化模型与原始模型的近似程度，是网格简化领域一个十分经典的算法。边收缩操作过程如下（如图 2.2 ），每次边收缩操作，减少一个点、两个面和三条边。

（1）去除原始网格中所有与顶点 $v_1$ 和顶点 $v_2$ 相关联的三角网格；

（2）将顶点 $v_1$ 和 $v_2$ 移至新的位置 $\bar{v}$ ；

（3）将与顶点 $v_1$ 和 $v_2$ 相关的边和三角网格的信息添加到顶点 $\bar{v}$ 的信息中。

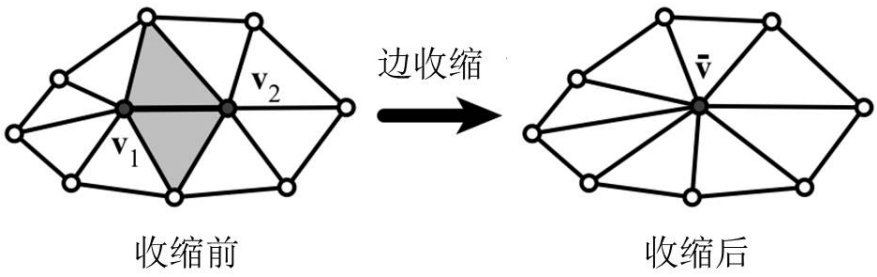



图 2.2: 网格简化方法

要得到较好的简化模型，重点在于选取合适的边进行收缩，使得边收缩最小程度地影响模型的视觉效果。在边收缩操作中，一条边的收缩与否是通过评价这条边收缩后对模型所带来的误差的大小决定的。 Garland 将简化带来的误差定义为收缩代价。

边收缩简化算法首先计算所有边进行收缩时的代价，然后将所有满足收缩条件的边按收缩代价的大小进行排序，再从代价最小的边开始收缩操作，删除在排序队列中于被删除边相关联的边，同时计算与新顶点相关联边的收缩代价，并按大小插入收缩代价队列中。如此重复操作，每次都选取收缩代价最小的边进行收缩操作，这样得到的简化模型与原始模型的误差最小。 Garland 算法在将三角网格中的两个顶点收缩为

一个新顶点时，首先计算它的二次误差测度，即计算该顶点到与原始网格中被收缩的两个顶点相关联的三角网格间距离的平方和，并据此求解方程得到新顶点的坐标位置。

### 2.1.2　网格简化结果

图 2.3、2.4、2.5 展示了一些网格简化的结果。

图 2.3: Stanford 兔子。从左到右点/面数依次是：35947/69451；3596/6945；362/693

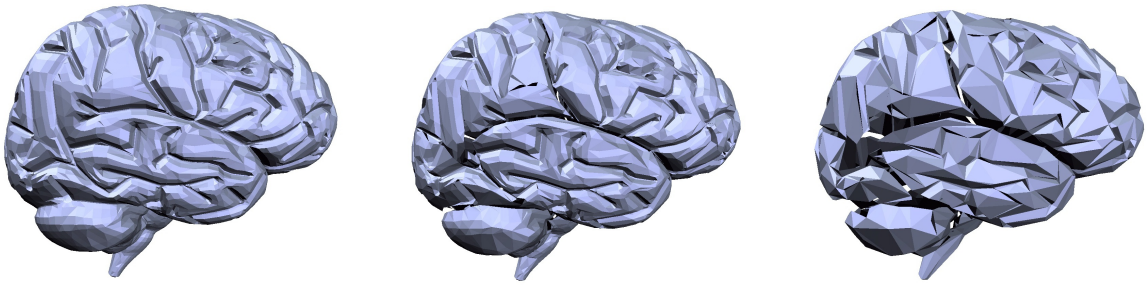


图 2.4: 人的大脑。从左到右点/面数依次是：14780/28390；7439/14195；1639/2839

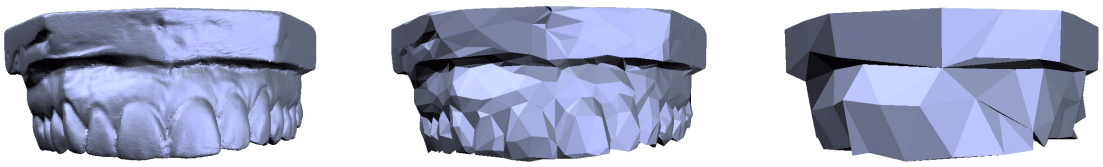


图 2.5: 人的牙齿。从左到右点/面数依次是：116604/233204；1238/2331；119/233

## 2.2 表面网格细分

如何能更精细的显示一个模型的变形结果，尤其是在形变程度较大的弯折处。近年来，细分曲面 (Subdivision Surface) 被越来越广泛的应用。其主要的原因在于细分方法可以由任意形状网格出发来自动快速的构造光滑曲面，不会影响运行速度。表面细分的模式有很多种。根据细分极限曲面与初始控制网格的位置关系分类，细分曲面一般可以分为逼近型细分方法和插值型细分方法。对于初始网格中的任意一个控制顶点，在不同的细分层次上均有一个控制点与其对应。如果这些点均相同，则这种细分方法称为插值型的，否则称为逼近型的。对于插值模型细分方法，初始网格的每一个控制顶点都存在于生成曲面当中，定义曲面的初始控制点也是极限曲面上的点，这使得可以更加直观地控制曲面。因此在曲面造型应用中，插值曲面具有容易控制生成曲面外形的特点。对于逼近模式，其生成的曲面的点不经过初始控制网格，因而生成的细分曲面比插值型细分产生的曲面质量更高。成熟应用的方法有 Catmull-Clark[51]、Doo-Sabin[52]、Loop[53]、Butterfly[54] 等。随着 GPU 的发展，这些方法已经可以在硬件上面实现了[55–59]。

### 2.2.1 网格细分方法

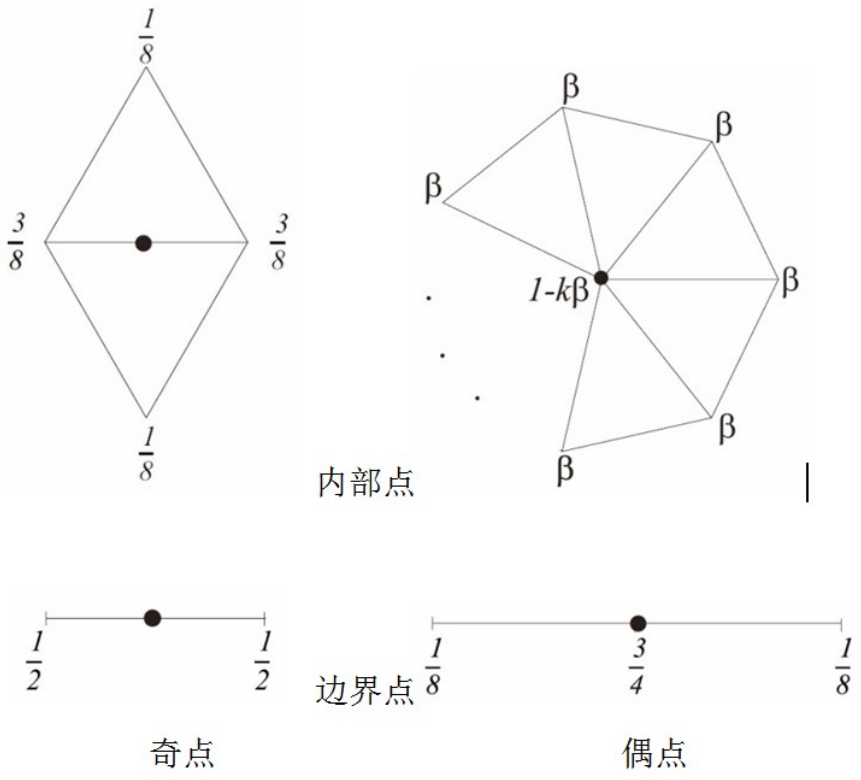



图 2.6: Loop 细分方法中几何点的生成规则

图 2.6 所示的是 Loop 的细分规则，也就是新顶点坐标的生成方法。其中粗黑的点是新生成的点，这些点都是由周围原来的旧点加权平均得到的，那些数字代表的就是权重值。其中棱上的点是完全新生成的（左图），而顶点处的点是对原来的顶点进行了位置更新（右图），以满足光滑性。这些新生成的点又分为内部点和边界点，所以共有四种情况生成新的点。

内部的奇点：假设有两个三角形 $(v_0, v_1, v_2)$ 和 $(v_0, v_1, v_3)$，共享边的两个顶点分别为 $v_1, v_2$，有

$$v = \frac{3}{8}(v_0 + v_1) + \frac{1}{8}(v_2 + v_3) \tag{2.1}$$

内部的偶点：假设点 $v$ 的度为 $k$，其周围的邻接点为 $v_i (i = 1, \ldots, k)$，权重为 $\beta$，则该偶点为其本身与其邻接点的加权平均和

$$v = (1 - k\beta)v + \beta \sum_{i=0}^{k-1} v_i \tag{2.2}$$

边界的奇点：假设一条边的两个顶点为 $v_0, v_1$，有

$$v = \frac{1}{2}(v_0 + v_1) \tag{2.3}$$

边界的偶点：若边界顶点在边界上的两相邻顶点为 $v_0, v_1$，那么该偶点为其本身与周围两点的加权平均和

$$v = \frac{3}{4}v + \frac{1}{8}(v_0 + v_1) \tag{2.4}$$

### 2.2.2　网格细分结果

图 2.7、2.8、2.9 展示了一些网格细分的结果

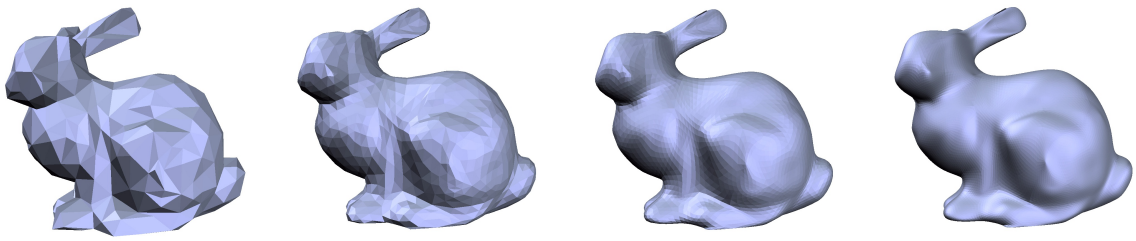


图 2.7: Stanford 兔子。点/面数依次：362/693；1416/2772；5603/11088；22293/44352

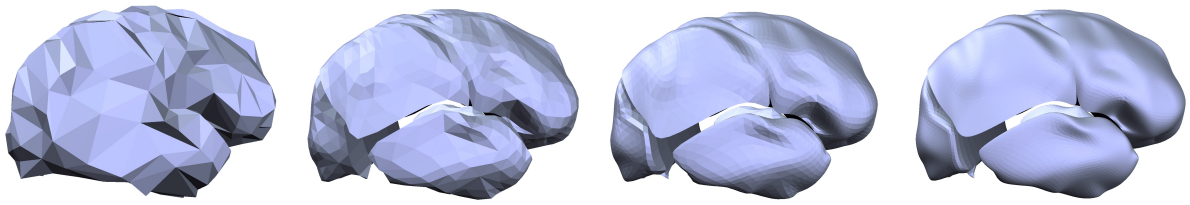


图 2.8: 人的大脑。 点/面数依次：509/928；1937/3712；7574/14848；29972/59392

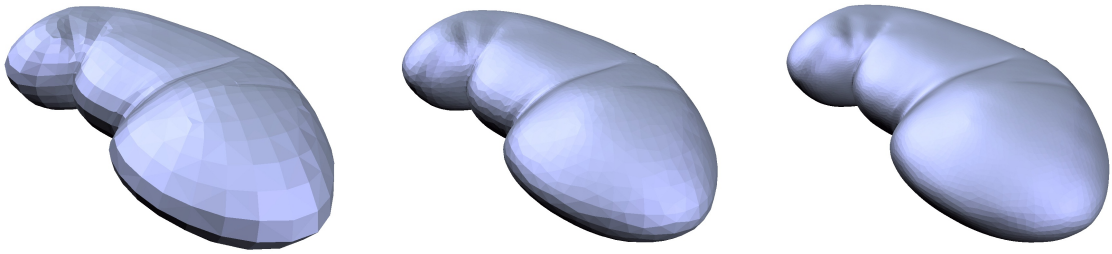


图 2.9: 人的肾脏。 点/面数依次：1212/2420；4832/9680；19302/38720

## 2.3 体网格剖分

有限元网格的生成是工程领域的一个重要课题，该技术在有限元分析、计算机辅助设计、计算机图形学、科学计算可视化、生物医学和地理信息系统等领域有着广泛的应用。作为有限元计算的前处理技术，网格剖分的工作量在有限元计算分析过程中占了相当大的比重，生成网格质量的好坏将直接影响到后续的工作结果。因此，网格生成技术在有限元分析中占着举足轻重的地位。人工划分网格相当费力、耗时，并且容易出错，划分具有复杂边界的三维实体更是如此，因此网格单元的自动划分技术是研究的热点。

在体网格剖分算法中，具有代表的有 Delaunay 算法[60, 61]，源于 1934 年俄国的数学家 B.Delaunay 提出的 Delaunay 准则；八叉树 (Oc-tree) 算法[62, 63]以及前沿推进法 (Advancing Front Technique, AFT)[64] 。 Delaunay 四面体剖分方法由于其成熟的理论及其判断准则，非常适用于三维实体的网格剖分。具有其他算法不可比拟的优越之处，其优越性主要体现在以下两方面：

(1) Delaunay 由于具有比较完善的数学理沦基础，能够确保对于任何复杂的输入模型进行四面体化时算法的收敛性；

(2) 算法效率高，所生成的单元质量好。

### 2.3.1　网格剖分方法

我们一般需要使用的是带约束的 Delaunay 四面体剖分 (Constrained Delaunay Tetrahedralization, CDT)，定义为在给定的限定（边界）条件下，将待剖分区域分解成满足 Delaunay 准则的四面体单元集合。而 Delaunay 有两大准则：

(1) 空球特性，即Delaunay 四面体的外接球内部不包含任何其他顶点，如图 2.10 所示，$q$ 点在 $abcd$ 形成的外接圆内，但是却在边界 $F$ 之外；而 $p$ 点在边界上，但却在外接圆的外部。

(2) 最小角最大化特性，即在点集可能形成的四面体剖分中，Delaunay 剖分所形成四面体的最小二面角最大。从这个意义上讲，Delaunay 剖分是最接近于规则化的的四面体。然而在边界处则是允许出现不满足 Delaunay 准则的四面体，这样的好处是避免在边界处产生大量微小体积的四面体网格。

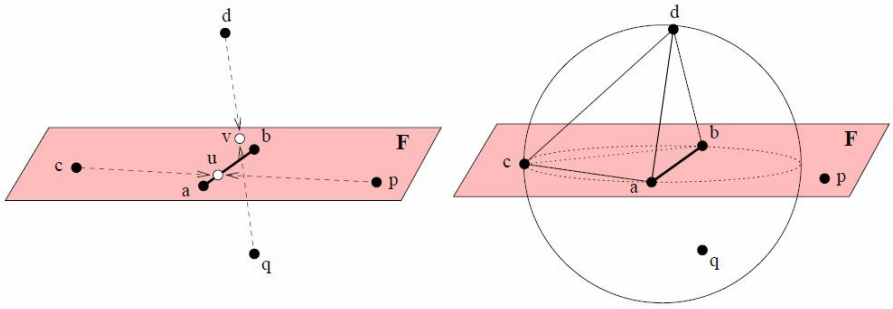



图 2.10: Delaunay 准则的空球特性

Hang [65]提出的三维 Delaunay 网格生成方法可以支持多种模型格式，被广泛使用。一个分段线性组合体 (Piecewise Linear Complex, PLC) $X$ 可以近似的表示任意复杂表面形状，因为它包含了顶点、片段和面。设 $X_0$ 是一个 PLC 的初始形态，$\rho_0$ 是半径-边长比，$T$ 是 Delaunay 四面体化之后的结果。算法如下：

(1) 初始化一个 Delaunay 四面体 $D_0$ 包含 $X_0$ 中的所有顶点；

(2) 根据半径边长比 $\rho$，通过插入新的点，在 $D_0$ 中恢复 $X_0$ 的结构，从而更新 $X_0 \to X_1$ 以及 $D_0 \to D_1$；

(3) 加入新的点从而去掉 $X_1$ 中退化的部分；从而更新 $X_1 \to X_2$ 以及 $D_1 \to D_2$；

(4) 复原 $D_2$ 中的子表面 $X_2$，从而得到最终的 $T$；(5) 加入 Steiner 点来提高网格质量并优化网格。

其中需要注意的是：$D_0$ 可以通过任何一种标准算法来建立；插入点必须遵守 Delaunay 的生成规则；$T$ 中任何四面体的半径边长比不能超过 $\rho_0$。CDT 生成的四面体网格能保持很好的稳定性，并确保二面角都在合适的范围内。

### 2.3.2 网格剖分结果

图 2.11、2.12 给出的是圆环细分前后的剖分结果。图 2.13、2.14 和 2.15 是简化的组织器官模型剖分结果。

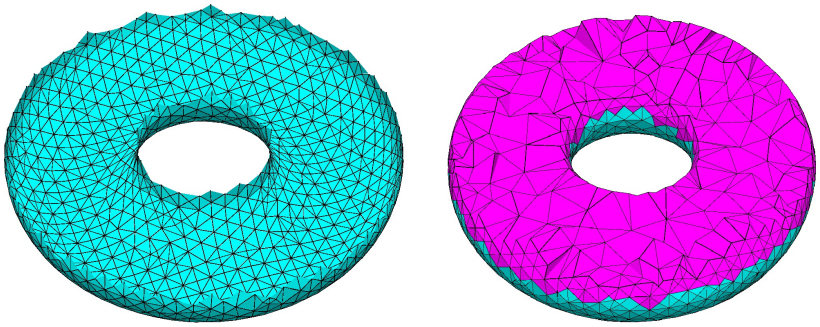


图 2.11: 圆环左：剖分前，点1729，面3458；右：剖分后，点2145，四面体7456

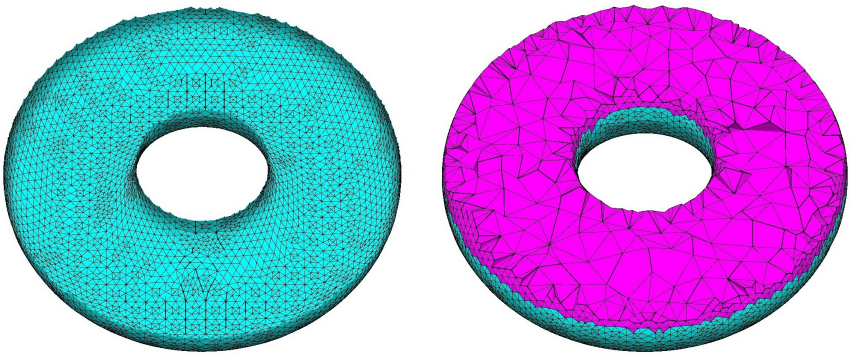


图 2.12: 细分圆环左：剖分前，点6916，面13832；右：剖分后，点8395，四面体27918

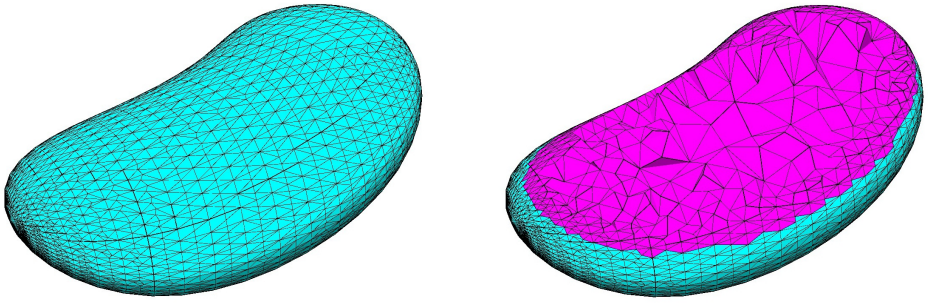


图 2.13: 肾左：剖分前，点1202，面2400；右：剖分后，点5922，四面体24369

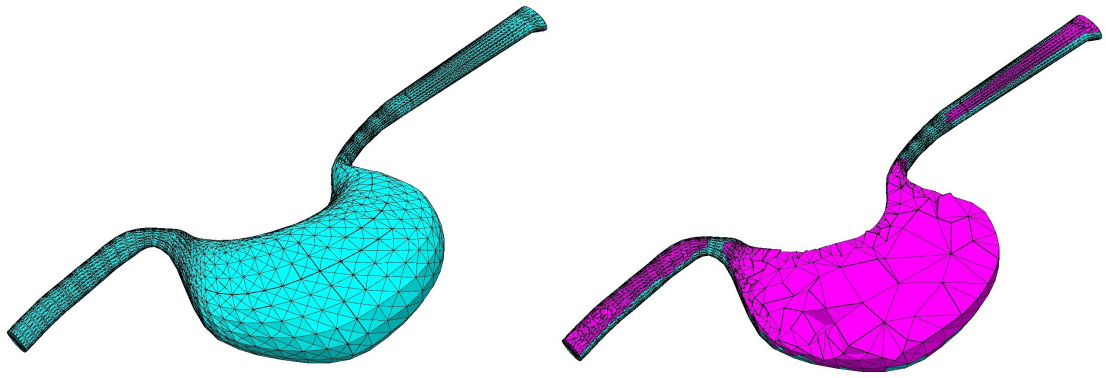


图 2.14: 胃：左：剖分前，点1226，面2448；右：剖分后，点9714，四面体37609

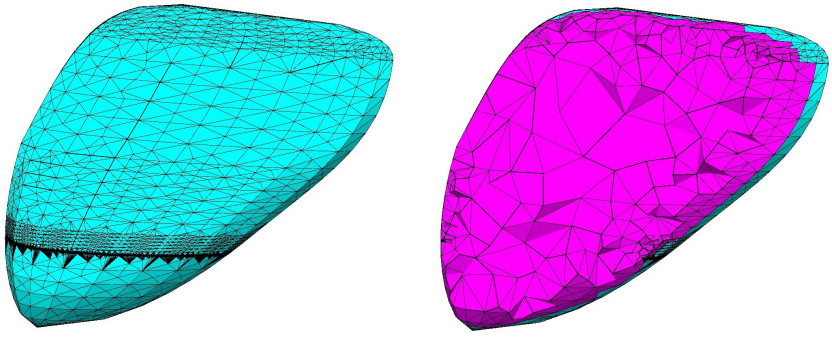


图 2.15: 肝：左：剖分前，点666，面1328；右：剖分后，点32717，四面体137426

## 2.4　本章小结

网格处理过程基本上属于成熟的技术，但为了适应不同的需求，还是有很多值得探索的地方，在这里不做过多的介绍。本章就网格处理展开讨论，先后介绍了网格简化，网格细分以及网格剖分的算法理论，并且给出了部分测试结果图。网格的处理是为了更好的进行有限元算法，下一章将详细讲述非线性有限元的处理方法。

# 第三章　非线性有限元形变模型

## 3.1　有限元方法概述

有限元方法是解决各种工程问题的常用方法。基本思想是将连续体机构离散成若干子单元，单元间通过边界上的结点相连接，形成一个组合体；然后用每个单元内的近似函数分段表示求解域内待求的未知变量，其中的近似函数用未知变量函数在每个单元结点上的值和与其对应的插值函数表示；最后通过基于变分原理的数学模型，建立求解未知量的代数方程组或者微分方程组，并通过数值方法求解方程组。

随着有限元技术的不断发展，有限元在医学应用中的发展也逐渐开始。由于生物软组织的不可压缩性、粘弹性、塑形、各向异性等非线性力学属性的影响，我们很难构造其精确的模型，并且由于计算的复杂性，同时也很难达到实时性。

虽然工程中有限元的应用已经非常成熟，但是生物力学中的应用才刚刚起步。目前国际上还没有类似于 ANSYS 或者 ABAQUS 等的商业成熟软件用于生物医学领域。

近些年来，很多研究者运用有限元理论，在虚拟手术软组织变形中做了许多模拟仿真。法国国家信息与自动化研究所 (INRIA) 在虚拟手术仿真应用领域一直保持着领先世界的优势。他们的开源工具 SOFA 是目前优秀的模拟软组织变形的平台，给出了精准快速的眼球变形模拟[66]，为白内障手术带来福音。 Joldes 等[67]采用自适应动态松弛的方法解决非线性有限元的问题，并应用到大脑变换的模拟中。 Boonma 等[68]模拟了微型手术针对皮肤的针刺效果，对“刺”前和“刺”后的软组织分开建模，加强了刺前的模拟速度，提高了刺后仿真精度。 Qin 等[69]提出了一个基于有限元的模型框架来估计腹部血管瘤的压力分布，实验结果与临床观察所得到的结果保持一致。 Li 等[70]用有限元成功模拟了插管手术中可交互导管的运动变形情况。 Dick 等[71]在 GPU 上面，测试了六面体网格的有限元运行效率，面对十万量级的网格单元，达到了每秒 11 个时间步长的高效运算速度。以上研究都出于实验室阶段，真正能投入市场应用的还没有出现。

## 3.2　有限元方程的建立

在连续介质力学中，物体的行为被假设为连续的物质而不是离散粒子的运动。为了展开有限元方法的应用，我们先从弹性力学理论的相关概念开始介绍，比如位移、应变、应力等，接着谈论超弹性模型的本构方程，引入非线性有限元平衡方程，最后给出求解方法。

### 3.2.1 力学基本概念

假设一个物体 $\Omega$ 从 $t=0$ 时刻的初始构型 $K_0$ 变形到 $t>0$ 的现时构型 $K_t$。 $\Omega$ 上一点 $P$，在参考坐标系内，未变形时的位置 (position) 向量为 $\mathbf{X}$，由于变形，点 $P$ 移动到新的位置 $\mathbf{x}$。也就是该点产生了位移 (displacement) $\mathbf{u}$，我们定义 $\mathbf{u}=\mathbf{x}-\mathbf{X}$。 $\Omega$ 上面所有的点由于形变而产生的位移场记为 $\mathbf{U}$。用有限元计算形变的关键就在于计算出 $\mathbf{U}$ 的值。

描述运动的方法有两种，用初始构型作为参考构型的方法称为物质描述（即 Lagrange 方法）；用现时构型作为参考构型的方法成为空间描述（即 Eular 方法）。参考构型不随着时间变化而变化的称为完全 Lagrange 方法，而随着时间变化不断更换初始构型的称为更新 Lagrange 方法。 Lagrange 方法以物质点为观察对象，在现时构型上创建基本方程，多用于固体问题中； Eular 方法从物体占据的空间点上来观察运动，多用来模拟流体的运动。我们这里针对软组织的变形采用的是 Lagrange 方法。

我们从位移的定义可以得到一个三维空间的变形映射函数 $\varphi$，满足

$$\mathbf{x}=\varphi(\mathbf{X})=\mathbf{X}+\mathbf{u}(\mathbf{X}) \tag{3.1}$$

物体内部发生变形以及产生变形的内力分别称为应变 $\boldsymbol{\varepsilon}$ 和应力 $\boldsymbol{\sigma}$。对于点 $P$ 我们在初始构型为参考构型的坐标下定义工程应变

$$\boldsymbol{\varepsilon}=\frac{\partial \mathbf{u}}{\partial \mathbf{X}}=\frac{\partial}{\partial \mathbf{X}}(\mathbf{x}-\mathbf{X})=\frac{\partial \varphi}{\partial \mathbf{X}}-1 \tag{3.2}$$

把受力点看做小立方体单元，在每个方向上面各有一对正向的应力应变和两对切向的应力应变，自由变量分别都是六个，向量形式为 $\hat{\boldsymbol{\varepsilon}}=[\varepsilon_{xx},\varepsilon_{yy},\varepsilon_{zz},\gamma_{xy},\gamma_{xz},\gamma_{yz}]$ 和 $\hat{\boldsymbol{\sigma}}=[\sigma_{xx},\sigma_{yy},\sigma_{zz},\sigma_{xy},\sigma_{xz},\sigma_{yz}]$， $\varepsilon$ 是正应变， $\gamma$ 是切应变，其矩阵形式分别为

$$\boldsymbol{\varepsilon}=\begin{bmatrix}\varepsilon_{11} & \varepsilon_{12} & \varepsilon_{13}\\ \varepsilon_{21} & \varepsilon_{22} & \varepsilon_{23}\\ \varepsilon_{31} & \varepsilon_{32} & \varepsilon_{33}\end{bmatrix}=\begin{bmatrix}\varepsilon_{xx} & \frac{1}{2}\gamma_{xy} & \frac{1}{2}\gamma_{xz}\\ & \varepsilon_{yy} & \frac{1}{2}\gamma_{yz}\\ sym & & \varepsilon_{zz}\end{bmatrix} \tag{3.3}$$

$$\boldsymbol{\sigma}=\begin{bmatrix}\sigma_{11} & \sigma_{12} & \sigma_{13}\\ \sigma_{21} & \sigma_{22} & \sigma_{23}\\ \sigma_{31} & \sigma_{32} & \sigma_{33}\end{bmatrix}=\begin{bmatrix}\sigma_{xx} & \sigma_{xy} & \sigma_{xz}\\ & \sigma_{yy} & \sigma_{yz}\\ sym & & \sigma_{zz}\end{bmatrix} \tag{3.4}$$

研究弹性力学的有限元分析，关键是三个基本方程：几何方程，物理方程和平衡方程。其中几何方程描述的是位移与应变之间的关系；物理方程描述的是应变与应力之间的关系；平衡方程描述的是应力与外力之间的关系。通过这三个方程，我们才能建立起位移与外力的关系，通常是已知外力，来求解物体位移，得到变形结果。下面几节就是围绕这三个方程展开的。

### 3.2.2　应变和应力

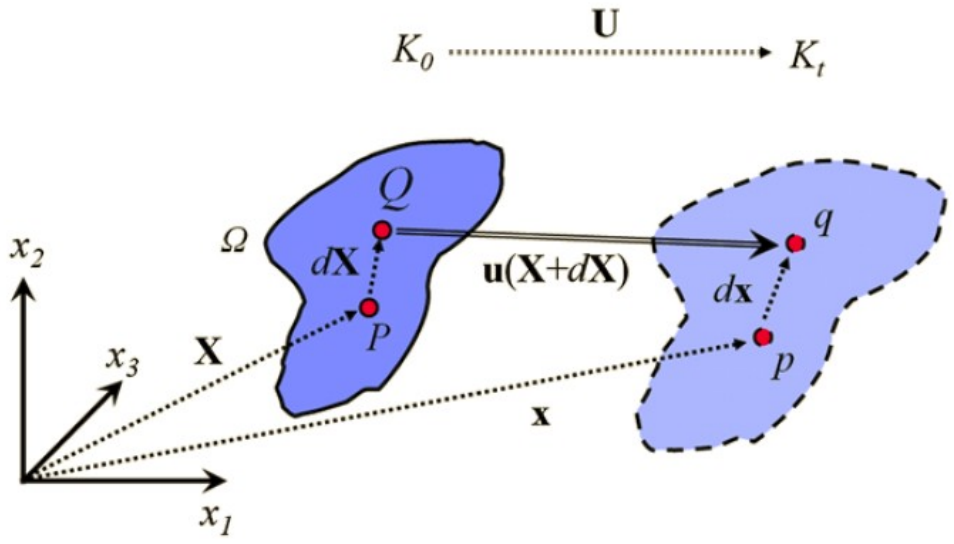



图 3.1: Ω 从 $K_0$ 变形为 $K_t$

我们考虑参考构型上相邻距离为 $d\mathbf{X}$ 的两个点 $P$ 和 $Q$，他们的位置向量分别是 $\mathbf{X}$ 和 $\mathbf{X}+d\mathbf{X}$。分别对应现时构型上的相邻距离为 $d\mathbf{x}$ 的两个点 $p$ 和 $q$，他们的位置向量分别是 $\mathbf{x}$ 和 $\mathbf{x}+d\mathbf{x}$。那么 $P$、$Q$ 两点的发生的位移分别为（见图 3.1）

$$\mathbf{u}(\mathbf{X}) = \mathbf{x}-\mathbf{X} \tag{3.5}$$

$$\mathbf{u}(\mathbf{X}+d\mathbf{X}) = (\mathbf{x}+d\mathbf{x})-(\mathbf{X}+d\mathbf{X}) \tag{3.6}$$

根据 (3.5) 我们可以得到微元 $d\mathbf{x}$ 和 $d\mathbf{X}$ 之间的关系

$$d\mathbf{x} = \left(\mathbf{I}+\frac{\mathbf{u}(\mathbf{X}+d\mathbf{X})-\mathbf{u}(\mathbf{X})}{d\mathbf{X}}\right)d\mathbf{X} = (\mathbf{I}+\nabla_{\mathbf{X}}\mathbf{u})d\mathbf{X} \tag{3.7}$$

$d\mathbf{X}$ 是初始构型中相距很近的两个点的位置差。在刚体运动或者小形变的过程中，这两个点所产生的位移在矢量上是相近的，因为只发生了平移和旋转，以及很小的形变，那么此时 $d\mathbf{x}$ 基本保持不变；但是在大变形过程中，两个点发生的位移相差很大，小的 $d\mathbf{X}$ 可能引起很大的 $d\mathbf{x}$，两者已经不能保持线性的关系，这时就需要设计针对大形变的应力和应变张量。

$\mathbf{H}=\nabla_{\mathbf{X}}\mathbf{u}$ 称为位移梯度张量，有

$$[H_{ij}] = \left[\frac{\partial u_i}{\partial X_j}\right] = \begin{bmatrix} \frac{\partial u_1}{\partial X_1} & \frac{\partial u_1}{\partial X_2} & \frac{\partial u_1}{\partial X_3} \\ \frac{\partial u_2}{\partial X_1} & \frac{\partial u_2}{\partial X_2} & \frac{\partial u_2}{\partial X_3} \\ \frac{\partial u_3}{\partial X_1} & \frac{\partial u_3}{\partial X_2} & \frac{\partial u_3}{\partial X_3} \end{bmatrix} \tag{3.8}$$

$\mathbf{F}=\mathbf{I}+\mathbf{H}$ 称为变形梯度张量，有

$$[F_{ij}]=\left[\frac{\partial x_i}{\partial X_j}\right]=\begin{bmatrix}\frac{\partial x_1}{\partial X_1} & \frac{\partial x_1}{\partial X_2} & \frac{\partial x_1}{\partial X_3}\\ \frac{\partial x_2}{\partial X_1} & \frac{\partial x_2}{\partial X_2} & \frac{\partial x_2}{\partial X_3}\\ \frac{\partial x_3}{\partial X_1} & \frac{\partial x_3}{\partial X_2} & \frac{\partial x_3}{\partial X_3}\end{bmatrix} \tag{3.9}$$

变形梯度张量反映了微元的变形与运动情况，一般为非对称的。 $\mathbf{F}$ 是一个线性变换，把初始微元 $d\mathbf{X}$ 变换成现时微元 $d\mathbf{x}$ 。使 $d\mathbf{X}$ 长度发生改变的同时也旋转一定的角度。

$J=\det(\mathbf{F})$ 称为变形梯度张量的 Jacobian 行列式。它反映了物体体积的变化率，等价于密度的改变，有 $J=\rho_0/\rho$ 。其中 $\rho_0$ 是初始构型的密度， $\rho$ 是现时构型的密度。

我们来看微元 $d\mathbf{x}$ 长度的平方

$$|d\mathbf{x}|^2=d\mathbf{x}^T d\mathbf{x}=d\mathbf{X}^T\mathbf{F}^T\mathbf{F}d\mathbf{X} \tag{3.10}$$

$\mathbf{C}=\mathbf{F}^T\mathbf{F}$ 称为右 Cauchy-Green 变形张量，再来看微元 $d\mathbf{x}$ 长度平方的改变量

$$|d\mathbf{x}|^2-|d\mathbf{X}|^2=d\mathbf{X}^T\mathbf{C}d\mathbf{X}-d\mathbf{X}^T d\mathbf{X}=d\mathbf{X}^T(\mathbf{C}-\mathbf{I})d\mathbf{X} \tag{3.11}$$

$\mathbf{E}=\frac{1}{2}(\mathbf{C}-\mathbf{I})$ 称为 Green-Lagrange 应变张量。

物体在外力作用下产生运动或者达到平衡，这些都是在变形之后发生的，因此原则上应该采用现时构型来描述应力的变化。假设物体在现时构型中有一个微小的面元 $\mathbf{n}dA$ ，上面受到 $dt$ 的作用力，则可推导出面元上的应力 $\mathbf{t}$ 。

$\mathbf{t}=\boldsymbol{\sigma}\mathbf{n}$ 称为 Cauchy 应力张量。该应力是基于现时构型，因此也叫真应力。但在前面的分析中可知，一般已知初始构型，而现时构型是未知的。为了方便计算，不采用 Eular 描述而使用 Lagrange 描述来定义应力张量。

$\boldsymbol{\tau}=J\boldsymbol{\sigma}$ 称为 Kirchhoff 应力张量。

$\mathbf{P}=J\boldsymbol{\sigma}\mathbf{F}^{-T}$ 称为第一 Piola-Kirchhoff 应力张量。

$\mathbf{S}=J\mathbf{F}^{-1}\boldsymbol{\sigma}\mathbf{F}^{-T}$ 称为第二 Piola-Kirchhoff 应力张量。

对于血流的模拟，可采用 Cauchy 应力张量；第二 Piola-Kirchhoff 应力张量是对称矩阵，比第一 Piola-Kirchhoff 应力张量更适合做数值模拟。

### 3.2.3 位移和应变的关系：几何方程

根据变形梯度张量和位移梯度张量之间的关系可得

$$\begin{aligned}\mathbf{E} &= \frac{1}{2}(\mathbf{F}^T\mathbf{F}-\mathbf{I})=\frac{1}{2}[(\mathbf{I}+\mathbf{H})^T(\mathbf{I}+\mathbf{H})-\mathbf{I}]\\ &= \frac{1}{2}(\mathbf{H}+\mathbf{H}^T+\mathbf{H}^T\mathbf{H})\end{aligned} \tag{3.12}$$

方程 (3.12) 表达的就是位移与应变的关系，也就是几何关系。这个公式明确的表达了应变是软体变形和刚体运动之间的差别。如果是没有发生形变的刚体运动，有 $\mathbf{F}^T\mathbf{F} = \mathbf{I}$，也就说明应变为零。

#### 3.2.3.1　小形变：线性几何方程

对于小形变问题，$\nabla\mathbf{u} \ll 1$，一阶偏导乘积项可以被忽略，我们得到线性小应变张量

$$\boldsymbol{\varepsilon} = \frac{1}{2}(\mathbf{H} + \mathbf{H}^T) \tag{3.13}$$

其分量形式为

$$\varepsilon_{ij} = \frac{1}{2}(\frac{\partial u_j}{\partial X_i} + \frac{\partial u_i}{\partial X_j}) \tag{3.14}$$

其矩阵形式为

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \frac{\partial u_1}{\partial X_1} & \frac{1}{2}(\frac{\partial u_1}{\partial X_2} + \frac{\partial u_2}{\partial X_1}) & \frac{1}{2}(\frac{\partial u_1}{\partial X_3} + \frac{\partial u_3}{\partial X_1}) \\ & \frac{\partial u_2}{\partial X_2} & \frac{1}{2}(\frac{\partial u_2}{\partial X_3} + \frac{\partial u_3}{\partial X_2}) \\ sym & & \frac{\partial u_3}{\partial X_3} \end{bmatrix} \tag{3.15}$$

其向量形式为

$$\hat{\boldsymbol{\varepsilon}} = \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \gamma_{xy} \\ \gamma_{xz} \\ \gamma_{yz} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial X_1} & 0 & 0 \\ 0 & \frac{\partial}{\partial X_2} & 0 \\ 0 & 0 & \frac{\partial}{\partial X_3} \\ \frac{\partial}{\partial X_2} & \frac{\partial}{\partial X_1} & 0 \\ 0 & \frac{\partial}{\partial X_3} & \frac{\partial}{\partial X_2} \\ \frac{\partial}{\partial X_3} & 0 & \frac{\partial}{\partial X_1} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \mathbf{P}\mathbf{u} \tag{3.16}$$

其中 $\mathbf{P}$ 是应变的梯度算子矩阵。

#### 3.2.3.2　大形变：非线性几何方程

对于大形变，由于一阶偏导乘积项 $\mathbf{H}^T\mathbf{H} = \nabla_{\mathbf{X}}\mathbf{u}^T\nabla_{\mathbf{X}}\mathbf{u}$ 不能忽略，所以应变和位移之间是非线性的关系。当形变很大的时候，这个一阶偏导乘积项也会非常的大，此时是典型的几何非线性。

其分量形式为

$$E_{ij} = \frac{1}{2}\left[\frac{\partial u_j}{\partial X_i} + \frac{\partial u_i}{\partial X_j} + \sum_{k=1}^{3}\left(\frac{\partial u_k}{\partial X_i}\right)\left(\frac{\partial u_k}{\partial X_j}\right)\right] \tag{3.17}$$

其向量形式为

$$\hat{\mathbf{E}} = \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \gamma_{xy} \\ \gamma_{xz} \\ \gamma_{yz} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial X_1} & 0 & 0 \\ 0 & \frac{\partial}{\partial X_2} & 0 \\ 0 & 0 & \frac{\partial}{\partial X_3} \\ \frac{\partial}{\partial X_2} & \frac{\partial}{\partial X_1} & 0 \\ 0 & \frac{\partial}{\partial X_3} & \frac{\partial}{\partial X_2} \\ \frac{\partial}{\partial X_3} & 0 & \frac{\partial}{\partial X_1} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} + \begin{bmatrix} \frac{1}{2}\sum_{k=1}^{3}(\frac{\partial u_k}{\partial X_1})^2 \\ \frac{1}{2}\sum_{k=1}^{3}(\frac{\partial u_k}{\partial X_2})^2 \\ \frac{1}{2}\sum_{k=1}^{3}(\frac{\partial u_k}{\partial X_3})^2 \\ \sum_{k=1}^{3}(\frac{\partial u_k}{\partial X_1})(\frac{\partial u_k}{\partial X_2}) \\ \sum_{k=1}^{3}(\frac{\partial u_k}{\partial X_2})(\frac{\partial u_k}{\partial X_3}) \\ \sum_{k=1}^{3}(\frac{\partial u_k}{\partial X_1})(\frac{\partial u_k}{\partial X_3}) \end{bmatrix} = \mathbf{P_X u} + \bar{\mathbf{P}}_{\mathbf{X}}(\mathbf{u}) \tag{3.18}$$

$\bar{\mathbf{P}}_{\mathbf{X}}$ 是应变和位移关系的非线性部分，这对后面的求解是一个非常难的问题。所以我们试用一种线性的关系来近似这种非线性的问题。显然，只用线性部分 $\mathbf{P_X u}$ 来近似 $\mathbf{P_X u} + \bar{\mathbf{P}}_{\mathbf{X}}(\mathbf{u})$ 是不合适的。考虑到 $\mathbf{P_X}$ 是针对初始构型的梯度，而在几何非线性变形中，单元的几何形状随着时间而改变，节点应变梯度也应该随之变化。不妨做如下假设

$$\mathbf{P_X u} + \bar{\mathbf{P}}_{\mathbf{X}}(\mathbf{u}) \approx \mathbf{P_X F}^T\mathbf{u} = \mathbf{P_x u} \tag{3.19}$$

这样就把几何非线性问题转化成了线性可求解问题。

### 3.2.4 应变和应力的关系：物理方程

物体在外力作用下发生产生应力发生形变，外力很小时，撤除外力后应力和形变也随之消失，这种可以恢复的特性就是弹性。若应力只是当前时刻的变形梯度张量 $\mathbf{F}$ 的函数，与之前的无关称为线弹性模型，比如最简单的弹簧模型。而与初始时刻和最后时刻的状态都相关的则称为超弹性模型，简单说，就是物体在外力作用下发生了远远超过弹性极限应变量的应变，但是卸载时应变却也能恢复到原来的状态。

物体应力和应变的关系称为本构关系。它反映了物体材质本身固有的属性。通常是将描述连续介质变形的参量与描述内力的参量联系起来的一组关系式。他们之间的关系式，又称本构方程。对于不同的物质，在不同的变形条件下有不同的本构关系，也称为不同的本构模型。本质上说，就是物理关系，建立的方程称为物理方程，它是结构或者材料的宏观力学性能的综合反映。架起了形变与内部物理学之间的桥梁。

He 等[72]从分子力的角度推导出不可压缩的弹性体的应力与形变的关系，给出了包括内力和表面牵引力的局部形变公式。 Mousavi 等[73]提出的统计有限元模型，考虑到不同个体的同一组织器官在形态属性方面的相似性，将组织器官的几何结构和有限元变形过程通过本构关系关联在一起，利用神经网络或者主成分分析的方法，建立两者的间接关系，从而减轻 FEM 的计算臃肿。

#### 3.2.4.1　线弹性：线性物理方程

一般来说，简化的物理方程是线性的，也就是说应力应变之间的关系可以用一条直线来模拟。这种情况下可以直接建立本构方程，应力张量 $\boldsymbol{\sigma}$ 是关于小应变张量 $\boldsymbol{\varepsilon}$ 的线性函数，即 $\boldsymbol{\sigma} = \mathcal{C}\boldsymbol{\varepsilon}$ 。其中 $\mathcal{C}$ 是描述材料应力应变关系的四阶弹性张量构成的弹性矩阵。组织的生物力学特性不仅仅取决于坐标位置，还取决于空间方向，是一种各向异性的材料，有 21 个不相关的弹性常数，但这种特性很难从图像数据中恢复出来，所以我们近似成各向同性且只有两个独立参数。应力与应变有如下关系

$$\boldsymbol{\sigma} = \lambda(tr(\boldsymbol{\varepsilon}))\mathbf{I} + 2\mu\boldsymbol{\varepsilon} \tag{3.20}$$

其分量形式

$$\sigma_{ij} = \lambda(tr(\boldsymbol{\varepsilon}))\delta_{ij} + 2\mu\varepsilon_{ij} \tag{3.21}$$

其矩阵形式

$$\boldsymbol{\sigma} = \begin{bmatrix} \lambda\sum_{k=1}^{3}\varepsilon_{kk} + 2\mu\varepsilon_{11} & 2\mu\varepsilon_{12} & 2\mu\varepsilon_{13} \\ 2\mu\varepsilon_{21} & \lambda\sum_{k=1}^{3}\varepsilon_{kk} + 2\mu\varepsilon_{22} & 2\mu\varepsilon_{23} \\ 2\mu\varepsilon_{31} & 2\mu\varepsilon_{32} & \lambda\sum_{k=1}^{3}\varepsilon_{kk} + 2\mu\varepsilon_{33} \end{bmatrix} \tag{3.22}$$

其向量形式

$$\hat{\boldsymbol{\sigma}} = \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ & & \lambda+2\mu & 0 & 0 & 0 \\ & & & \mu & 0 & 0 \\ & sym & & & \mu & 0 \\ & & & & & \mu \end{bmatrix} \hat{\boldsymbol{\varepsilon}} = \mathcal{C}\hat{\boldsymbol{\varepsilon}} \tag{3.23}$$

其中 $\lambda$ 和 $\mu$ 称为 Lamé 系数，与弹性模量 $E$ 和泊松比 $\nu$ 的关系

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \quad \mu = \frac{E}{2(1+\nu)} \tag{3.24}$$

#### 3.2.4.2　超弹性：非线性物理方程

但是大多数生物力学模型都属于非线性弹性问题，虽然应力与应变之间依然保持一一映射的关系，但是却不保持线性。我们也可以用 $\mathcal{C}$ 表示这种映射， $\hat{\mathbf{S}}$ 和 $\hat{\mathbf{E}}$ 分别是应力 $\mathbf{S}$ 和应力 $\mathbf{E}$ 的六自由度向量形式，有

$$\hat{\mathbf{S}} = \mathcal{C}(\hat{\mathbf{E}}) \tag{3.25}$$

大多数这样的物理方程是由力学实验拟合出的结果，也有一些经典方程是通过超弹性模型的应变能函数 $W$ 推到而出，用来描述超弹性材料。这里我们仅给出基于 Green-Lagrange 应变张量 $\mathbf{E}$ 的第二 Piola-Kirchhoff 应力张量 $\mathbf{S}$ 与应变能函数 $W$ 之间的关系，同时由 $\mathbf{E}=\frac{1}{2}(\mathbf{C}-\mathbf{I})$ 可得应变能函数与右 Cauchy-Green 变形张量的函数关系

$$\mathbf{S}=\frac{\partial W(\mathbf{E})}{\partial \mathbf{E}}=2\frac{\partial W(\mathbf{C})}{\partial \mathbf{C}} \tag{3.26}$$

对于各向同性的超弹性材料， $W$ 只是 $\mathbf{C}$ 的不变量的函数，即

$$W=W(\mathbf{C})=W(I_1,I_2,I_3) \tag{3.27}$$

其中 $\mathbf{C}$ 的不变量分别为

$$I_1=tr(\mathbf{C}),\quad I_2=\frac{1}{2}(tr(\mathbf{C^2})-tr(\mathbf{C})^2),\quad I_3=\det(\mathbf{C})=J^2 \tag{3.28}$$

那么第二 Piola-Kirchhoff 应力张量为

$$\begin{aligned}\mathbf{S}=2\frac{\partial W}{\partial \mathbf{C}} &= 2\frac{\partial W}{\partial I_1}\frac{\partial I_1}{\partial \mathbf{C}}+2\frac{\partial W}{\partial I_2}\frac{\partial I_2}{\partial \mathbf{C}}+2\frac{\partial W}{\partial I_3}\frac{\partial I_3}{\partial \mathbf{C}}\\ &= 2\frac{\partial W}{\partial I_1}\mathbf{I}+2\frac{\partial W}{\partial I_2}(I_1\mathbf{I}-\mathbf{C})+2\frac{\partial W}{\partial I_3}I_3\mathbf{C}^{-1}\end{aligned} \tag{3.29}$$

对于不可压缩的材料，由于体积保持不变， $I_3=1$ 。

对于大变形，线性模型不在适用，可以采用最简单的 St. Venant-Kirchhoff 非线性模型，其应变能函数为

$$W(\mathbf{E})=\frac{\lambda}{2}(tr(\mathbf{E}))^2+\mu\mathbf{E}^2 \tag{3.30}$$

第二 Piola-Kirchhoff 应力张量为

$$\mathbf{S}=\frac{\partial W}{\partial \mathbf{E}}=\lambda(tr(\mathbf{E}))\mathbf{I}+2\mu\mathbf{E} \tag{3.31}$$

Neo-Hookean 模型是从 St. Venant-Kirchhoff 发展来的，多用于模拟橡胶物质的可压缩模型，其应变能函数为 ($\bar{I}_1=J^{-2/3}I_1$)

$$W(\mathbf{C})=\frac{\mu}{2}(\bar{I}_1-3)+\lambda(J-1)^2 \tag{3.32}$$

第二 Piola-Kirchhoff 应力张量为

$$\mathbf{S}=\mu J^{-2/3}\mathbf{I}+\lambda J(J-1)\mathbf{C}^{-1} \tag{3.33}$$

Mooney-Rivlin 模型是一种更一般的 Neo-Hookean 模型，其应变能函数为

$$W(\mathbf{C})=\frac{\mu_1}{2}(I_1-3)-\frac{\mu_2}{2}(I_2-3) \tag{3.34}$$

### 3.2.5　应力和外力的关系：平衡方程

位移与应变、应变与应力的关系已经得出，最后需要建立动态平衡方程找出内力与外力之间的关系。变形过程用静力学方程来分析是不合适的，需要通过动力学运动方程来推导。而运动方程分又分为微分形式和积分形式。积分形式就是常说的虚功方程，这在有限元应用中具有十分重要的理论基础及地位。在非线性问题中（包括几何非线性和材料非线性），类似于线性问题，从平衡微分方程和应力边界方程出发，建立虚功方程，再此基础上推导出有限元离散方程用于数值计算。内部虚功与外部虚功的总和为零，故采用 Lagrange 描述的基于第二 Piola-Kirchhoff 应力张量 $\mathbf{S}$ 以及 Green 应变张量 $\mathbf{E}$ 的虚功方程为

$$\Pi = \underbrace{\int_{\Omega} \frac{1}{2}\mathbf{E}^T\mathbf{S}dV} - \underbrace{\int_{\Omega} \mathbf{u}^T\mathbf{f}_0 dV - \int_{\Gamma} \mathbf{u}^T\mathbf{t}_0 dA} \tag{3.35}$$

其中， $\mathbf{f}_0$ 为单位非变形体积所受的体力， $\mathbf{t}_0$ 为单位初始面积的牵引力向量， $\delta\mathbf{v}$ 是虚速度。由于采用的是 Lagrange 描述，所以积分的都是基于初始构型而言的。

## 3.3　有限元方程的求解

针对复杂的几何和物理结构，有限元模拟可以达到高精度的仿真。有限元解的精度依赖于两点：单元网格剖分精细程度和每个单元上线性独立的基底函数个数。所以提高 FEM 精度有三种方法，一种是加密有限元网格剖分以逼近正确解 (h-FEM)；另一种网格剖分不变，增加各单元基底多项式的阶数 (p-FEM)，还有一种是联合方法 (h-p-FEM)。 Tanveer 等[74]给出的混合 p-FEM 方法，用较少的几何单元和力场，以及高阶的基底多项式，对非线性超弹性模型进行分析，并与 ANSYS 软件的结果做比较，效率更高，结果更准确。要求解非线性方程组3.35，首先需要线性化该方程，也就是求解虚功方程的方向导数。此外，虚功方程是针对整个区域的积分，计算时需要将区域离散化成一个个有限单元的积分和。

对于四面体网格结构 $e$，网格中任意一个点 $\mathbf{u}_i$ 的坐标可以由节点坐标 $\mathbf{u}_{e_j}(j = 1,2,3,4)$ 线性表示，有 $\mathbf{u}_i = \mathbf{N}_e\mathbf{u}_{e_j}$，其中 $\mathbf{N}_e$ 表示该单元的形函数，用于坐标变换。在单元结构 $e$ 中，应变和位移的关系如下

$$\hat{\mathbf{E}}_e = \mathbf{P}_\mathbf{X}\mathbf{F}^T\mathbf{u} = \mathbf{P}_\mathbf{X}\mathbf{F}^T\mathbf{N}_e\mathbf{u}_e = \mathbf{B}\mathbf{u}_e = \mathbf{B}(\mathbf{u}_{e_1} : \mathbf{u}_{e_2} : \mathbf{u}_{e_3} : \mathbf{u}_{e_4})^T \tag{3.36}$$

其中， $\mathbf{B}$ 被称为四面体单元的应变-位移矩阵。

### 3.3.1　直接刚度矩阵法

先来考察线性物理方程对应的有限元问题，通过应变位移的关系以及和应力应变的关系，对于任意有限元单元 $e$，虚功方程 3.35 可写成

$$\Pi_e = \int_{\Omega_e} \frac{1}{2}(\mathbf{B}\mathbf{u}_e)^T\mathcal{C}(\mathbf{B}\mathbf{u}_e)dV_e - \int_{\Omega_e} (\mathbf{N}_e\mathbf{u}_e)^T\mathbf{f}_0 dV_e - \int_{\Gamma} (\mathbf{N}_e\mathbf{u}_e)^T\mathbf{t}_0 dA$$

$$= \frac{1}{2}\mathbf{u}_e^T \underbrace{(\int_{\Omega_e} \mathbf{B}^T \mathcal{C}\mathbf{B} dV_e)}_{\mathbf{K}_e} \mathbf{u}_e - \mathbf{u}_e^T \underbrace{(\int_{\Omega_e} \mathbf{N}_e^T \mathbf{f}_0 dV_e + \int_{\Gamma} \mathbf{N}_e^T \mathbf{t}_0 dA)}_{\mathbf{r}_e} \tag{3.37}$$

我们从中可以得到含有四个节点信息的 $12 \times 12$ 的单元刚度矩阵

$$\mathbf{K}_e = \int_e \mathbf{B}^T \mathcal{C}\mathbf{B} dV_e \tag{3.38}$$

其中 $\mathcal{C}$ 是弹性矩阵。通过变分原理，我们得到刚度矩阵和外力之间的关系

$$\frac{\partial \Pi_e}{\partial \mathbf{u}_e} = \mathbf{K}_e \mathbf{u}_e - \mathbf{r}_e = \mathbf{0} \tag{3.39}$$

分别求出每个单元的刚度矩阵以后，组装得到整体的刚度矩阵，有

$$\mathbf{K} = \sum_e \mathbf{K}_e \tag{3.40}$$

对于静态有限元分析，不需要考虑惯性因素，但是在动力学分析中，尤其是虚拟手中组织的变形，惯性力和阻尼力都是不可忽略的。因此该方程组不是简单的代数方程组，而是一个复杂的非线性微分方程组。

考虑到系统中的质量、阻尼、刚度矩阵所引起的三种力学因素，以及外部载荷的影响，系统离散后可以得到平衡方程组

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{D}\dot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{r} \tag{3.41}$$

其中 $\dot{\mathbf{u}}$ 是节点的速度向量， $\ddot{\mathbf{u}}$ 是加速度向量， $\mathbf{M}$ 是质量矩阵， $\mathbf{D}$ 是阻尼矩阵， $\mathbf{K}$ 是刚度矩阵， $\mathbf{r}$ 是外部载荷向量。

关于二阶常微分方程组的数值求解方法，一般有显示 Eular 法、隐式 Eular 法以及四阶龙格-库塔法。但是在该问题中，矩阵的阶数都比较高，通常的算法不切实际，一般使用直接积分法。直接积分法在时间域上进行离散，进行插值，化作差分格式，然后根据初始条件和边界条件，对离散后的线性代数方程组逐时刻求解位移。解决动态有限元问题，一般采用两种时间积分方式，一种是显示积分，另一种是隐式积分。隐式有限元一般具有较高的稳定性，比如 Newmark 方法[75]， Wilson $\theta$ 方法[76]。其中对于线性问题，时间步长可以任意选择而依旧保持形变稳定。 Joldes 等[77]给出了一种求解稳定时间步长的计算方法，该方法不仅适用于有限元模型，同时也适用于无网格的粒子模型。

这时问题转化成大矩阵的线性方程组求解问题，需要求解稀疏刚度矩阵的逆，是一个计算量十分庞大的工程。

### 3.3.2　间接刚度矩阵法

对于非线性物理方程，由于弹性矩阵 $\mathcal{C}$ 不是简单的线性矩阵，所以 $\mathbf{K}$ 是位移 $\mathbf{u}$ 的非线性函数，无法按照上节的方法直接求出单元刚度矩阵再进行叠加。平衡方程与线性的略有区别

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{D}\dot{\mathbf{u}} + \mathbf{K}(\mathbf{u})\mathbf{u} = \mathbf{r} \tag{3.42}$$

这时原先的线性方程组就变成了非线性方程组，$\mathbf{K}(\mathbf{u})\mathbf{u}$ 更无法离散化到每个位移点来求解。我们在这里通过线性方法的推导，对方程进行一些变化，以避开刚度矩阵的求解。

现在来考察平衡方程中的刚度矩阵项 $\mathbf{K}(\mathbf{u})\mathbf{u}$，结合 (3.36)，(3.25) 和 (3.38)，推导如下：

$$\begin{aligned}
\mathbf{Ku} &= (\sum_e \mathbf{K}_e)\mathbf{u} \\
&= (\sum_e \int_e \mathbf{B}^T \mathcal{C}\mathbf{B} dV_e)\mathbf{u} \\
&= \sum_e \int_e \mathbf{B}^T \mathcal{C}\mathbf{B}\mathbf{u}_e dV_e \\
&= \sum_e \int_e \mathbf{B}^T \mathcal{C}\hat{\mathbf{E}}_e dV_e \\
&= \sum_e \int_e \mathbf{B}^T \hat{\mathbf{S}} dV_e \\
&= \sum_e \mathbf{f}_e \\
&= \mathbf{f}(\mathbf{u})
\end{aligned} \tag{3.43}$$

可以看出，我们将非线性刚度矩阵项转化成了 $\mathbf{f}$ 的计算，$\mathbf{f}$ 称为内应力，也就是刚度矩阵所对应的刚度力。由每个单元节点的内力 $\mathbf{f}_e$ 组合而成，其中在四面体单元结构中有

$$\begin{aligned}
\mathbf{f}_e &= \int_e \mathbf{B}^T \hat{\mathbf{S}} dV_e \\
&= V_e \mathbf{B}^T \hat{\mathbf{S}}
\end{aligned} \tag{3.44}$$

其中 $\hat{\mathbf{S}}$ 是第二 Piola-Kirchhoff 应力的向量形式，$\hat{\mathbf{S}} = [S_{11}\ S_{22}\ S_{33}\ S_{12}\ S_{23}\ S_{13}]$。

经过如上变换，原方程 (3.42) 转化为

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{D}\dot{\mathbf{u}} + \mathbf{f} = \mathbf{r} \tag{3.45}$$

其中 $\mathbf{f}$ 是节点对应的内应力，$\mathbf{r}$ 是其对应的外部载荷。

我们考虑一致质量矩阵，也就是将物体的质量分散集中到每个节点上，只有节点存在重力作用。此时质量矩阵 $\mathbf{M}$ 化为对角矩阵，对角线元素表示每个节点的质量值。此外，阻尼矩阵通常与质量和刚度相关，常用 Rayleigh 阻尼，即 $\mathbf{D}=\alpha\mathbf{M}+\beta\mathbf{K}$ $(\alpha=const,\beta=0)$ 也是对角矩阵。内应力 $\mathbf{f}$ 和外部载荷 $\mathbf{r}$ 都可看做是节点独立的，也就是每个节点拥有自己独立的内力和外力分量。

对质量、阻尼和刚度矩阵离散化之后，平衡方程 (3.45) 可写做

$$\mathbf{M}(\ddot{\mathbf{u}}_n+\alpha\dot{\mathbf{u}}_n)=\mathbf{r}_n-\mathbf{f}_n \tag{3.46}$$

其中 $\mathbf{M}=diag(M_{11},M_{22},\cdots,M_{nn})$ ， $n=1,2,3,\cdots,3N$ ， $N$ 是节点数量。

### 3.3.3 中心差分的显式积分法

对于显式积分，积分时间步长受到限制，通常需要首先计算临界时间步长以保证变形过程不会因为步长过大而结果发散。这里采用中分差分的方法来进行处理，好处是可以转化为迭代过程，避免求解大规模非线性方程组。

现在问题转化成了微分方程的求解， $\dot{\mathbf{u}}$ 和 $\ddot{\mathbf{u}}$ 分别是位移对时间 $t$ 的一阶和二阶导数，通过中心差分的方法将微分形式转化为可以数值计算的差分格式

$${}^{t}\dot{\mathbf{u}}_n=\frac{1}{2\Delta t}({}^{t+\Delta t}\mathbf{u}_n-{}^{t-\Delta t}\mathbf{u}_n) \tag{3.47}$$

$${}^{t}\ddot{\mathbf{u}}_n=\frac{1}{\Delta t^2}({}^{t+\Delta t}\mathbf{u}_n-2\,{}^{t}\mathbf{u}_n+{}^{t-\Delta t}\mathbf{u}_n) \tag{3.48}$$

其中 $\Delta t$ 是单位时间步长， ${}^{t}\mathbf{u}_n$ 是当前时间点的位移， ${}^{t-\Delta t}\mathbf{u}_n$ 是一个单位时间步长之前的位移， ${}^{t+\Delta t}\mathbf{u}_n$ 则是之后的位移。把这两个公式应用到平衡方程 (3.46) 中并整理可得 $\mathbf{u}_n$ 所满足的递推公式

$${}^{t+\Delta t}\mathbf{u}_n=\gamma_1\mathbf{M}_{n\times n}^{-1}({}^{t}\mathbf{r}_n-{}^{t}\mathbf{f}_n)+\gamma_2^{t}\mathbf{u}_n+\gamma_3^{t-\Delta t}\mathbf{u}_n \tag{3.49}$$

其中 $\gamma_1=2\Delta t^2/(\alpha\Delta t+2)$ ， $\gamma_2=4/(\alpha\Delta t+2)$， $\gamma_3=1-\gamma_2$ 。

类似中心差分法这样的显示时间积分方法都是条件稳定的。时间步长 $\Delta t$ 应该被限制在如下范围内

$$\Delta t=\alpha\Delta t_{cr}=\alpha\frac{L_e}{c}(0<\alpha\leq 1) \tag{3.50}$$

其中 $L_e$ 是四面体单元中所有单元的最小长度， $c$ 是物质膨胀波的速度，有

$$c=\sqrt{\frac{E(1-\nu)}{\rho(1+\nu)(1-2\nu)}} \tag{3.51}$$

其中 $E$ 杨氏模量， $\nu$ 是泊松比， $\rho$ 是密度。这些都是物体本身的属性。不同个体的不同组织器官都有自身固有的属性。不同器官之间的差异较大，同种器官在不同个体中

的差异较小。在虚拟手术过程中，如果只是做手术训练，那么我们只需要一组通用属性参数就可以了。这些参数一般是通过医学测量得到的，对于一个标准医学模型，有如下结果

表 3.1: 不同组织器官的物理参数

| | 软组织 | 肿瘤 | 空腔 |
|---|---|---|---|
| 密度 $(g/mm^3)$ | 0.001 | 0.001 | 0.001 |
| 泊松比 (无量纲) | 0.49 | 0.49 | 0.3 |
| 杨氏模量 $(MPa)$ | 3000 | 6000 | 100 |

除了医学手术训练，在真实手术之前也可能需要进行虚拟的手术模拟，进行术前仿真。人体组织不是均匀的结构，我们很难把它等同于某种材质，上面所给出的物理参数只是一个参考值。这时我们就需要得到该病人组织的精确物理参数。通过与真实模型变形的配准比较，训练出更符合实际的参数用于真实情况的模拟。我们要想得到更准确的参数，需要对这些参数进行真实估计。用建立的模型与病人身上可采集的真实组织做形变的比较，得到最优解用于手术部位模型的参数估计。也可以加入一些其他控制参数，在交互过程中进行人为干预。

迭代时间步长的选取影响着最终的效果。时间步长 $\Delta t$ 首先有上限 $\Delta t_c r$ ，大于这个值会导致系统不稳定，出现错误的结果；步长太小会增加迭代次数，增加计算负担。过小的时间步长在理论上会提高模拟精度但同时会增加累计误差，所以步长的选取对结果至关重要。

## 3.4　本章小结

本章重点叙述了非线性有限元方程的构建，针对几何非线性和物理非线性带来的计算困难，运用一些技巧将其线性化，最终推导出可以用于计算机计算的迭代求解方法。本章涉及到的是各向同性的超弹性模型，并没有考虑各向异性、粘弹性以及弹塑性模型的构建与解决。下一章将详细阐述如何在 GPU 上实现本章的算法并提出一个碰撞模型。

# 第四章　非线性有限元方法的具体实现

## 4.1　OpenGL和CUDA简介

自 SGI (Silicon Graphics) 公司发布 OpenGL 1.0 版本以来，已经有 20 年的历史了。在 2012 年， Khronos 集团宣布推出 OpenGL 4.3 [78] 和面向移动平台的 OpenGL ES 3.0 。开放性图形库 (Open Graphics Library, OpenGL) ，是一个三维的计算机图形和模型库，最初是美国 SGI 公司为图形工作站开发的一种功能强大的三维图形机制（或者说是一种图形标准）。它源于 SGI 公司为其图形工作站开发的 IRIS GL ，在跨平台移植过程中发展成为 OpenGL 。作为一个性能优越的图形应用程序设计界面 (API) ，OpenGL 适用于广泛的计算机环境。从个人计算机到工作站和超级计算机， OpenGL 都能实现高性能的三维图形功能。 OpenGL 本身针对移动端和桌面端有多个版本，它是开发 iPhone、Android 以及很多控制器和桌面系统 3D 图形应用和游戏的重要标准之一。 OpenGL 被设计成独立于硬件，独立于窗口系统，在运行各种操作系统的各种计算机上都可用，并能在网络环境下以客户/服务器模式工作，是专业图形处理、科学计算等高端应用领域的标准图形库。主要竞争对手是微软的 DirectX ，该图形库是以 COM 接口形式提供的，所以极为复杂，稳定性差，另外微软公司拥有该库版权，目前只在 Windows 平台上可用。

图形处理器 (Graphics Processing Unit, GPU) 是特意为图形领域计算密集性、高并行性的运算而设计，因此 GPU 相比 CPU 而言，具有更多的用于数据运算的算术逻辑单元 (ALU) ，以及高内存带宽和多核心处理器。于是，人们将 GPU 用于非图形处理，即通用 GPU (General Purpose Graphics Processing Unit, GPGPU) 。早先的 GPGPU 开发，要求开发人员熟练掌握图形 API ，将待处理的问题映射到着色器上进行计算，将数据映射为纹理，用高级着色语言（如： Cg、GLSL、HLSL 等）编写 shader 程序，然后通过图形 API（如： OpenGL 和 DirectX ）执行。某种程度上，这种流程增加了开发难度，限制了 GPU 的应用。 2007 年， NVIDIA 公司推出的统一计算设备架构 (Compute Unified Device Architecture, CUDA) [79]，能够有效利用 GPU 的强劲运算能力和巨大的存储带宽进行通用计算，在图像视频处理、模式识别、石油勘探、流体力学、生物计算、分子动力学和金融分析领域，取得一到两个数量级的加速。 CUDA 架构有效的降低了 GPU 开发的门槛，正在广泛的应用于各个领域。而昂贵的 GPU 的储存单元一直是制约 GPU 发展的一个因素。

近年来，随着GPU硬件架构的不断更新，从 Tesla到 Fermi 再到 Kepler 。 GPU 的运算能力一直以超越摩尔定律的速度飞速发展，而 GPU 的浮点运算能力相比发展速度缓慢的 CPU 而言具有明显的优势。同时 GPU 又支持最新的 OpenGL 。随着 OpenGL 3.2 的推出，将集合着色器 (Geometry Shader, S) 加入到了图形流水线中，从此可以通

过硬件来处理庞大的顶点信息。最新的 OpenGL 4.3 版本标准，不仅带来了更好的图形表现，同时也增加了更多之前没有的效果，可以直接使用 OpenGL 技巧来使用运算着色器 (Compute Shader, CS) 释放 GPU 上的任务。这样可以在不耗用更多 CPU 功耗的情况下实现更好的物理和人工智能的模拟[80]。

因此我们希望通过 GPU 对着色器和 CUDA 的支持，设计符合 GPU 架构的高效并行算法及显示处理，充分利用 GPU 的强大处理能力和巨大存储带宽，针对不同时刻物体不同部位的形变大小，实现自适应的模型真实感绘制，从剖分到细分再到有限元计算，以更快的速度性能，得到更好的结果，使得虚拟手术的效果更加逼真。

## 4.2 有限元算法在 GPU 上的实现

CUDA 编程模型将 CPU 作为主机 (Host)， GPU 作为协处理器 (co-processor) 或者设备 (Device)。在一个系统中可以存在一个主机和多个设备。 CPU 主要负责进行逻辑性强的事物处理和串行计算， GPU 则专注于执行高度线程化的并行处理任务。 CPU 、 GPU 各自拥有相互独立的存储器地址空间： Host 端的内存和 Device 端的显存。

运行在 GPU 上的 CUDA 并行计算函数称为内核函数 (kernel)。一个 kernel 函数并不是一个完整的程序，而是整个 CUDA C 程序中的一个可以被并行执行的步骤。

CPU 串行代码完成的工作包括在 kernel 启动前进行数据准备和设备初始化的工作，以及在 kernel 之间进行一些串行计算。理想情况是， CPU 串行代码的作用应该只是清理上一个内核函数，并启动下一个内核函数。这样，就可以在设备上完成尽可能多的工作，减少主机与设备之间的数据传输。

### 4.2.1 显式迭代的并行化

用显式积分法求解有限元方程是非常适合 GPU 并行计算的。该方法主要由两大部分的循环组成：一个是对于每个有限元单元计算内应力 $\mathbf{f}_n$ ；另一个是针对每个节点计算下一时间步长所产生的位移 ${}^{t+\Delta t}\mathbf{u}_n$ ，后者的计算是依赖于前者计算的结果。在每个循环内，我们可以采用并行计算，因为对于有限元单元和节点来说，他们都是独立于周围的邻接单元和节点的。这种基于数据的并行计算，可以采用 GPU 的 CUDA 架构来实现并行处理单元的快速并行计算。

从上一章的推导来看，有一些数据在整个运算过程中保持不变，那么就可以对这些数据进行预计算。例如应变位移矩阵，中心差分方程的系数等可以提前计算好并储存起来。这些一次性计算的数据虽然可以是离线不计时间成本的，但我们依然传输到 GPU 端进行计算并保存在显存上，用于后面的迭代计算。因为 CPU 和 GPU 之间的通信是非常慢的，我们尽量一次性把数据放到显存，不增加传输负担。对于有限元变形配准问题来说，目标位移都是提前就算好的，变形过程中位移的更新可以一直在 GPU

中进行，几乎不需要通信。但对于实时交互的变形来说，每个时间步迭代之后，位移和力都有了一次更新，对于显示来说，可以把数据直接通过显存进行显示。但是对于力反馈来说，还需要将计算得到的数据传回 CPU 进行力的反馈，并更新外部载荷的大小。

我们使用线性四面体单元的方程，当用来仿真几乎不可压缩的材料时，这些单元结构会非常好的在 GPU 上生成所需要的结果。每个单元有四个顶点，每个顶点又有三个自由度。所有要存储的数据矩阵都是这些的倍数，比如应变位移矩阵是 $6 \times 12$ ，单元节点的内力向量是 $12 \times 1$ 等等。这样的维数非常适合转存到四通道 (RGBA) 的纹理内存，便于数据传输显示。我们使用图形 API (OpenGL) 与 GPU 交互，并行的 Kernel 函数用 CUDA 完成。

### 4.2.2 Kernel1：单元循环

这个Kernel函数的目的是计算上一章中介绍的单元节点的内力，也就是刚度力。在进行这步操作之前，所有预处理的变量都已经计算完毕储存在显卡的全局内存里面。假设该物体被剖分成 $N_e$ 个四面体单元 $T_i(i = 1, \cdots, N_e)$ ，那么将发起 $N_e$ 个独立线程，每个线程负责一个单元上四个节点的内力计算，而每个节点又有三个方向的内力。

对于每一个发起的线程，需要负责的计算包括：(1) 体积 $V_e$ 的计算，由于采用的是 Lagrange 方法，以初始构型为参考系，体积 $V_e$ 是初始状态下单元的体积，这形变过程中这个量不改变。所以可通过预处理得到。(2) 形变-位移矩阵 $\mathbf{B}$ ，需要先计算出位移梯度张量，也就是需要上一步的节点位移，可以通过全局节点位移得到；同时还需要形函数的梯度张量，预处理可得到。(3) 应力 $\hat{\mathbf{S}}$ ，需要通过位移梯度张量先算出右 Cauchy-Green 变形张量，再通过本构方程求出应力的分量值。

对于每个四面体单元，需要求解的是一个 $12(4 \times 3)$ 维向量。当所有单元节点的内力向量计算完成时，对于任一个节点，他在单个单元上拥有一组值，但在整个空间内却拥有多组数值，分别来自于每一个包含该节点的四面体单元。我们需要把这些相同节点的内力值叠加到一起，形成整体内力矩阵。一种方法是将这些节点数据传回内存，用 CPU 实现；另一种方法则是在下面的核函数中实现。

### 4.2.3 Kernel2：节点循环

假设变形物体共有 $N_n$ 个节点 $V_i(i = 1, \cdots, N_n)$ ，发起 $N_n$ 个独立线程用来计算每个节点的位移向量，每个位移向量包括三个方向的分量。

首先先把上个核函数得到的单元内应力值进行合并，那么需要构建一个索引表，每个节点 $V_i$ 对应一个索引表 $Id_i[(s_1, t_1), (s_2, t_2), \cdots]$ ，索引表上每一对 $(s, t)$ 对应着该节点分别出现在单元 $T_s$ 的第 $t$ 个点( $t \in \{1, 2, 3, 4\}$ )。通过对这种动态索引表的循环，每个节点线程首先计算出该点的整体内力值 $\mathbf{f}_i$ 。

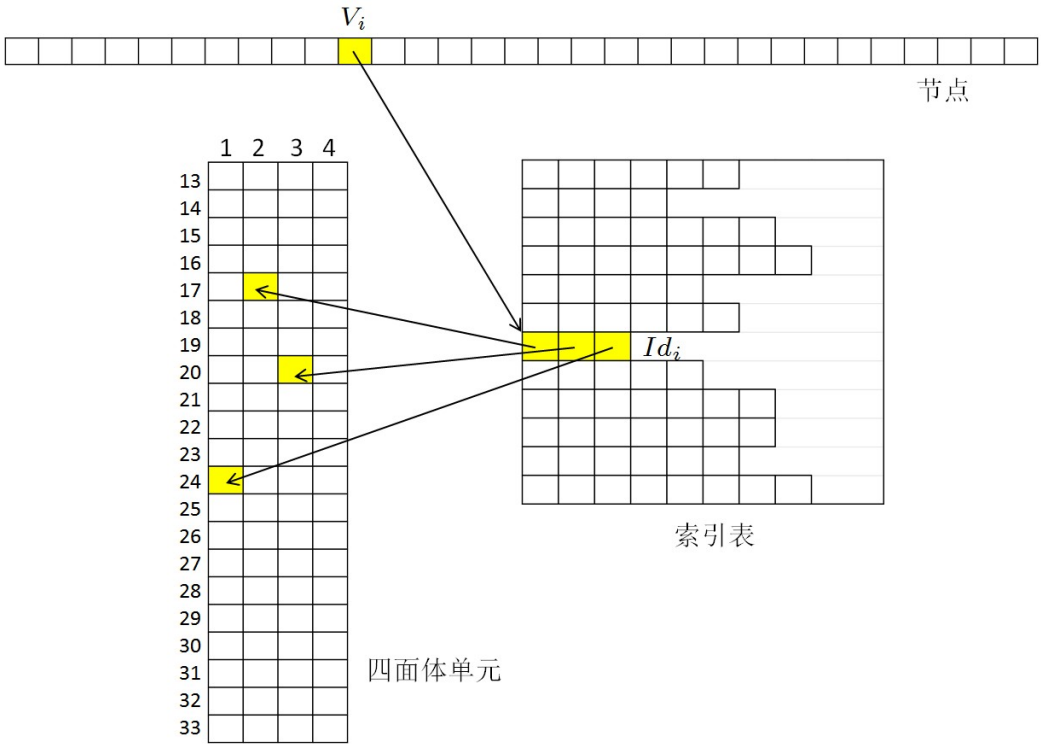
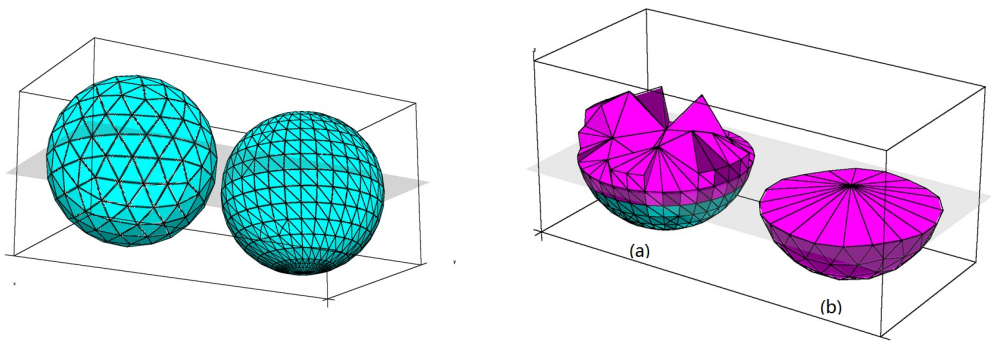



图 4.1: 整体内力矩阵的合成

如图 4.1 所示，节点 $V_i$ 对应索引表 $Id_i$，对应着三个单元节点，$(17,2),(20,3),(24,1)$，在不同四面体中的三个点实际指向同一个节点。

接着计算出外力 $\mathbf{r}_i$ 的值，包括重力，交互力等。不管是何种力，都要首先转化成针对节点的独立的力。当所以值都确定之后，利用差分公式，迭代出下一时刻的位移。这种更新位移的方法，很适合计算机求解。对于节点循环来说，所需要的总时间等于最慢的线程进行循环求内力所需要的时间。也就是说，如果一个剖分的几何体中，有某些节点同时共享了很多四面体的时候，计算效率就会下降，比如图 4.2 所对应剖分。




(a) 面网格　　(b) 体网格

图 4.2: 不同的四面体剖分结构

球体 (b) 所采用的剖分形式导致了球心的节点大量共享四面体，从计算的角度来看会大大降低运算效率，同时出现狭长的四面体不满足 Delaunay 四面体剖分的准则之一：最大化最小角，而且对变形本身也会产生很大的不稳定性。

## 4.3　快速碰撞模型

在交互式虚拟手术中，用户控制一个虚拟手术器械去自由碰撞虚拟器官。所以必须将碰撞节点以及产生的力的大小计算出来，传给 GPU 去计算位移，再将位移传回去计算反馈力的大小和方向。也就是说，在下一时间步计算之前，要完成上一步数据的传出以及反馈力的计算，并且在这一时间步内运行 Kernel2 之前把这次的外力数据传入。在不影响实时性的前提下完成这一系列操作也是目前需要解决的问题之一。

物体发生形变的外力主要来自于碰撞，与刚体碰撞检测相比较，可变形体的碰撞检测更为复杂。不光需要考虑物体间的碰撞，还要考虑到大变形下的自碰撞[81]。在刚体碰撞检测中，通过使用一些空间数据结构可以是算法的效率得到很大提高，这些数据是建立在预处理阶段完成的。而可变性物体的碰撞检测中，这些数据需要实时更新，从而使性能大幅度下降，因此选择数据需要十分小心。正确的碰撞处理需要获取大量的碰撞信息，包括接触点穿透深度等，针对可变形物体的特性，需要有合适的方法来快速获取这些信息。

可变形物体碰撞检测主要有这几种算法：层次包围体[82]，空间剖分[83, 84]，随机方法[85, 86]以及距离场[87, 88]等方法。 Levin 等[89]模拟了 Eular 构型下大规模物体的碰撞， Allard 等[84]模拟了任意分辨率下的大规模物体碰撞， Kaufman 等[90]模拟了带有摩擦力的大规模物体碰撞。

在变形过程中，物体的刚度是约束物体形变的重要原因之一。此外就是周围物体对其碰撞以及刚体边界带来的全局约束。物体发生碰撞的过程可以假设成求解局部几何碰撞，包括点/点，点/面的优化问题。目前已有一些方法来求解这种优化问题，比如连续碰撞检测 (Continuous Collision Detection, CCD) 以及线性互补问题 (Linear Complementarity Problem, LCP) 的 Gauss-Seidel 的迭代求解方法[90]。

我们假设软组织碰撞刚体边界的过程，简化之后就是点与面发生的碰撞。为了找到一种局部最优解，我们提出了基于投影方式碰撞检测，描述不同摩擦情况下的碰撞效果[91]。这种简易碰撞模型不但判断了碰撞的发生以及碰撞程度，还加入了摩擦效果的模拟。

本文的主要目的是有限元过程的处理，故没有使用过于复杂的碰撞检测方法。但是为了能够发生形变，这里给出了一种与刚体平面碰撞的简单模型。我们将碰撞模型简化成位移的改变只发生在碰撞那一 $\Delta t$ 时间内，要求的只是碰撞发生的真正位置，不考虑碰撞中发生的物理过程。这种模型不需要计算碰撞产生的力以及速度，也就是不需要对外力 $\mathbf{r}$ 进行修改。只需要对每个时间步迭代生成的位移 $\mathbf{u}$ 进行判断处理。此外，

使用该方法也减轻了数据传输的压力，因为位移的更新只需要在 GPU 上面完成，不需要将外力传输到 GPU 中进行计算。

我们假设 ${}^{t}\mathbf{x}$ 是物体上的一点，将要在时刻 $t$ 到 $t+\Delta t$ 内与局部碰撞表面 (Local Contact Surface, LCS) 发生碰撞。 LCS 碰撞接触面是一个局部平面，所以也可以是曲面上的一段平面，但是这样局部接触面为了便于分析，首先把整个模型旋转，使碰撞面的法向与 $y$ 轴正向保持一致。

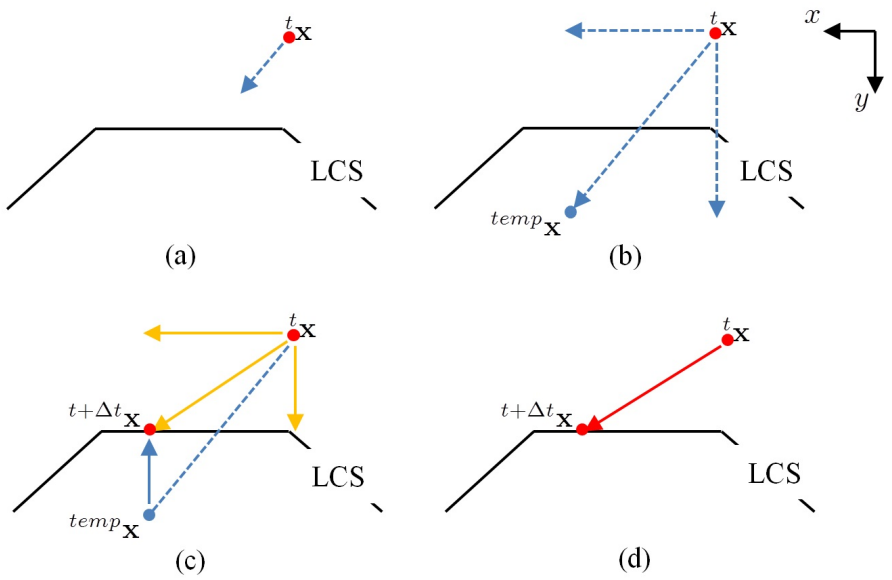



图 4.3: 内投影碰撞模型， ${}^{t+\Delta t}\mathbf{x}$ 是节点的最终碰撞位置

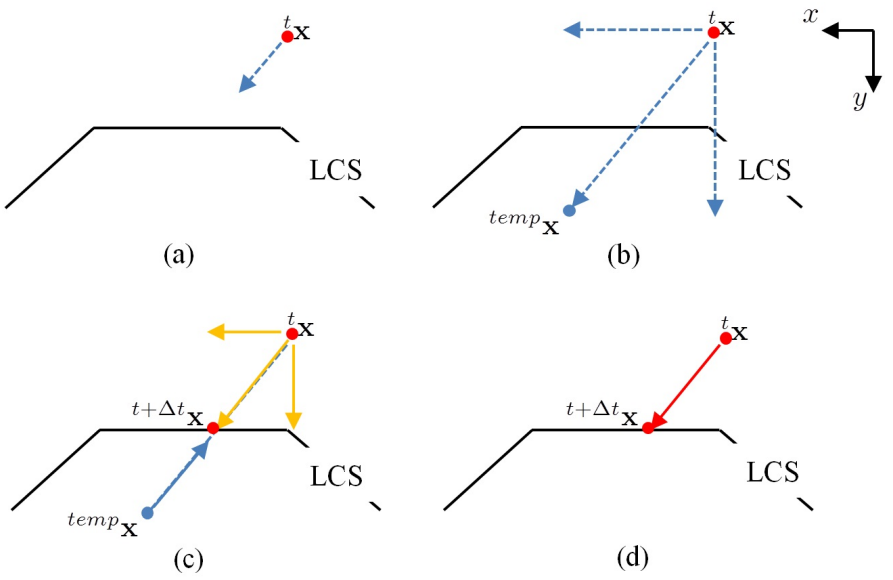



图 4.4: 外投影碰撞模型， ${}^{t+\Delta t}\mathbf{x}$ 是节点的最终碰撞位置

在图 4.3 和图 4.4 中，我们阐述了二维中的两种碰撞模型。由于变形的效果，节点

${}^{t}\mathbf{x}$ 在下一时刻本应该处于 ${}^{temp}\mathbf{x}$ 的位置。然而新的位置却在边界也就是碰撞面之外，我们需要将其投影到碰撞面 LCS 上。

在内投影模型中，节点 ${}^{t}\mathbf{x}$ 在 $x$ 方向的位移不需要改变，只需要缩短 $y$ 方向的位移。这意味着在接触面的 $x$ 方向上没有摩擦力，节点“滑”向下一时刻的位置。但是在外投影模型中， $x$ 和 $y$ 方向的位移都相对缩短了。也就是碰撞面是粗糙有摩擦的，在静磨擦的作用下，节点停止了横向运动。这两种投影模型是两种特殊的极限情况，更多时候是介于这两种情况之间，物体受到滑动摩擦的影响。

现在考虑三维的情况，碰撞点 ${}^{t}\mathbf{x}$ 的位移 ${}^{t+\Delta t}\mathbf{u}$ 将会进行一次更新

$$ {}^{t+\Delta t}\mathbf{u} = {}^{t}\mathbf{u} + ({}^{t+\Delta t}\mathbf{x} - {}^{t}\mathbf{x}) = {}^{t}\mathbf{u} + [u_x, u_y, u_z]^T \tag{4.1} $$

$u_x$ 表示滑动摩擦的范围从静摩擦到无摩擦， $u_y$ 在各种摩擦中都等比例减少， $u_z$ 和 $u_x$ 的情况相同。我们定义 ${}^{temp}\mathbf{x} - {}^{t}\boldsymbol{x} = [\hat{u}_x, \hat{u}_y, \hat{u}_z]^T$ ，则更新的位移是

$$ \begin{aligned} (u_y/\hat{u}_y)\hat{u}_x = u_{xout} \leqslant \quad & u_x \quad \leqslant u_{xin} = \hat{u}_x \\ & u_y \quad = |{}^{t}\mathbf{x} - LCS| \end{aligned} \tag{4.2} $$

$$ (u_y/\hat{u}_y)\hat{u}_z = u_{zout} \leqslant \quad u_z \quad \leqslant u_{zin} = \hat{u}_z \tag{4.3} $$

该算法不需要求解优化问题，避免了复杂的计算，只需要对一个方向的预测位移进行判断，并同比例更新其他两个方向的位移。该方法的计算量相对于有限元的计算可以忽略不计，到达变形效果的同时又保证了不增加计算的复杂性。但是其局限性也是相当明显的，适用范围仅限于点与规则平面的碰撞，也就是一种边界约束，无法实现两物体碰撞以及物体自身的碰撞效果。

## 4.4　CUDA-OpenGL 互操作

CUDA 和 OpenGL 互操作的基本方式是使用CUDA生成位移数据，然后使用OpenGL在屏幕上绘制出数据所表示的图形。两者的结合可以通过两种方式来实现：(1) 使用 OpenGL 的像素缓冲区对象 (PBO) 。在该方式下， CUDA 直接生成像素数据， OpenGL 显示这些像素；(2) 使用 OpenGL 的顶点缓冲区对象 (VBO) 。在该方式下， CUDA 生成顶点网格数据， OpenGL 可以根据需要绘制出平滑的表面图或线框图或一系列顶点，也可以通过各种shader对图形就行后续处理，例如实现表面的各种精细显示。

这两种方式的核心都是利用 cudaGLMapBufferObject 函数将 OpenGL 的缓冲区映射到 CUDA 的内存空间上，这样就可以充分利用 CUDA 的优点写出性能高的程序在该内存空间上生成数据，这样如果只有一个GPU，就不需要数据传递， OpenGL 可以直

接使用。如果不使用 CUDA ，这些数据需要由 CPU 来计算产生。一方面， CPU 的计算速度通常比 GPU 慢；另一方面，这些数据需要传送到 GPU 上以供 OpenGL 显示使用。鉴于此，当数据量很大时， CUDA 和 OpenGL 的混合使用会非常方便和快捷。

## 4.5 算法流程

基于第三章的算法原理以及 GPU 上的具体实现细节，仿真软组织变形就是要实时求出模型上所有点的位移情况。针对方程 (3.49) 中 $\mathbf{u}$ 的求解，有限元变形算法设计如下

表 4.1: 有限元变形算法步骤

| 有限元变形算法步骤（含运算位置） |
| --- |
| ▶ 开始 |
| ▶ 原始模型表面网格处理 |
| ▶ 进行四面体剖分得到体网格 |
| ▶ 读取模型的点、面、体数据并复制到显存 |
| ▶ 预计算后面需要用到的常量以及边界设置[CPU+GPU] |
| ▶ 实时处理循环开始{ |
| ▶ 从外部设备输入交互力[CPU] |
| ▶ 计算质量矩阵 $\mathbf{M}$ [GPU] |
| ▶ 计算整体外力向量 $\mathbf{r}$ [GPU] |
| ▶ 计算内力向量 $\mathbf{f}$ (Kernel 1) [GPU] |
| ▶ 计算位移向量 $\mathbf{u}$ (Kernel 2) [GPU] |
| ▶ 通过碰撞模型更新位移向量 $\bar{\mathbf{u}}$ [GPU] |
| ▶ 计算交互力传回设备[CPU] |
| ▶ 新节点坐标 = 旧节点坐标 + $\bar{\mathbf{u}}$ [GPU] |
| ▶ 绑定新节点坐标到纹理内存[GPU] |
| ▶ 模型渲染显示[GPU+OpenGL] |
| ▶ }单次变形循环结束 |
| ▶ 清除显存数据，结束 |

## 4.6 本章小结

使用 CUDA API 和 OpenGL API ，我们基本实现了该算法的 GPU 版本。表 4.1 所示的是一个完整的变形仿真系统的计算流程。其中质量矩阵的计算通常会放到预处理的步骤中，因为在变形过程中，每个节点的质量改变很小，对整个系统影响不大。但

在这里，为了模型更准确，我们在每一个时间步之内都要重新计算一次质量矩阵。下一章我们将对具体的实验结果做一些分析。

# 第五章　实验结果分析

在非线性有限元的计算过程中， GPU 计算的高效性一般是毋容置疑的。但是其准确性和稳定性却像一个黑盒子一样，需要我们用数据去说话，是我们进行分析的重要组成部分。人体组织器官虽然结构复杂，但是都可以转化为立方体、圆柱体、椭球体的组合。这些标准的测试物体通常采用六面体网格就可以得到很好的效果。但是对于拓扑结构更为复杂的物体，我们常采用四面体网格变形的准确性除了与硬件设备有关系外，还和剖分单元数量以及算法的鲁棒程度有关。

如图 2.11 所示，我们采用直径 $10mm$ 的圆环体，节点数 2145 ，剖分后的四面体单元数 7456 。距离圆环 $2mm$ 的地方我们建立一个刚体碰撞面，在只考虑重力的情况下，观察碰撞效果并分析结果。仿真平台是 Visual Studio 2008 ，编程语言是 C++、CUDA 4.2 以及 OpenGL 4.3 ，在这个平台上搭建了碰撞变形的有限元程序。硬件平台有 $2.4GHz$ 的 CPU、$4GB$ 的内存以及包含 $4GB$ 显存和 1536 个 CUDA 内核的 GTX680 显卡。

软组织的材料属性影响着碰撞结果的准确性和稳定性。我们在这里采用近似人体组织器官的材料属性，包括了密度 $0.001g/mm^3$ ，阻尼系数 $\alpha = 10$，非线性物理模型采用 Neo-Hookean 本构方程，杨氏模量为 $3000MPa$ ，泊松比为 0.49 属于近似不可压缩的材质。在符合要求的时间步长范围内，我们选择的步长 $\Delta t = 0.000025s$，从碰撞过程初始到完全平静下来需要较长的时间，我们只考察碰撞模拟的前$0.2s$。之所以没有采用组织器官模型作为分析对象，是为了与工程标准模型做对比，测试该算法的准确性。我们通过和商业软件 ABAQUS 的结果做比较，来判断我们的整个系统的稳健性。

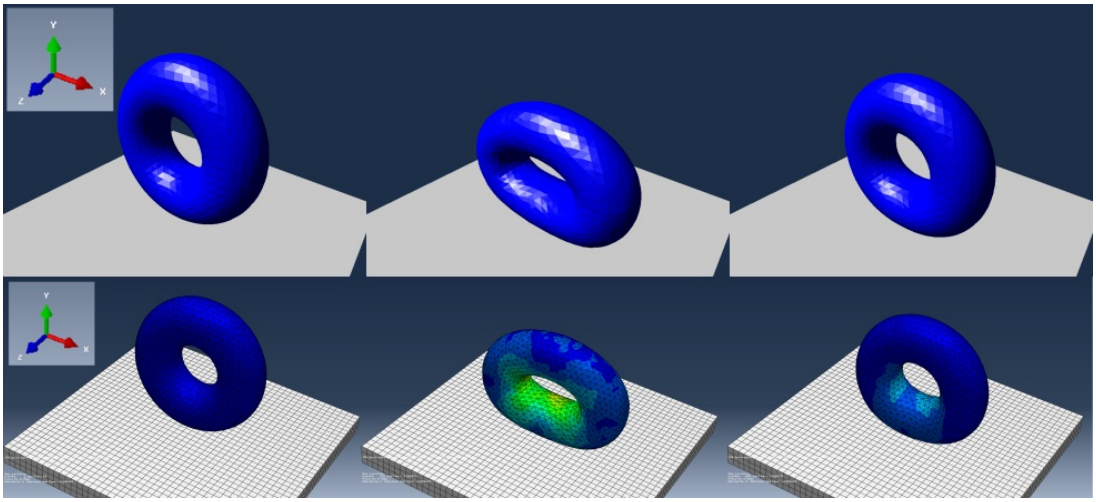


图 5.1: 圆环体变形比较

图 5.1 中所示就是我们的算法与 ABAQUS 算法的显示结果。上图纯色的圆环体是本文的结果，下图带有应力分布颜色的是 ABAQUS 的结果。直观来看结果是很接近的。

## 5.1 变形过程的模拟

图 5.2 所示的是圆环体在下落碰撞时的变形过程。大致可以分为四个阶段，从位移曲线中可以很容易判断出每个阶段的变化。

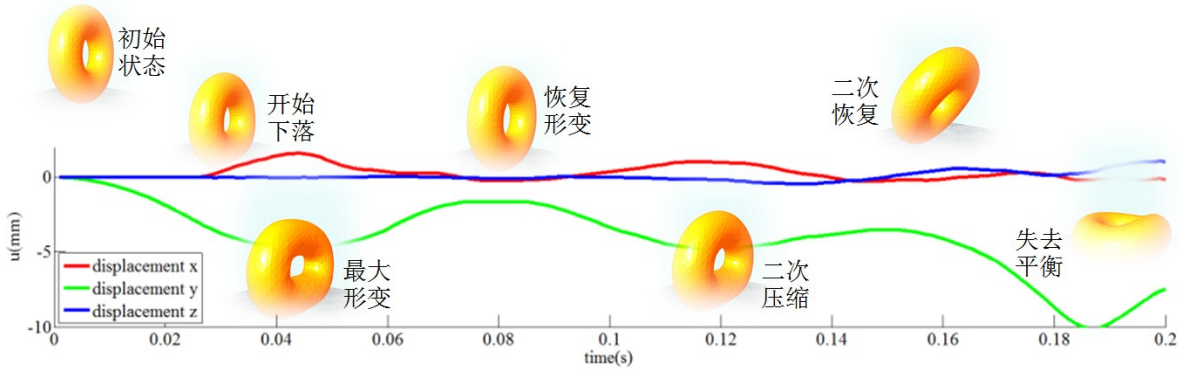



图 5.2: 圆环下落碰撞的变形模拟

1) $0s \sim 0.03s$：**下落阶段**

在前 $0.03s$，圆环在只有重力的作用下加速下落，可以看到 $x$ 和 $z$ 方向的位移几乎没变，只有 $y$ 方向的位移在不断减小。

2) $0.03s \sim 0.07s$：**首次碰撞压缩变形**

在 $0.03s$ 的时候，圆环首次发生碰撞，变形开始。由于碰到了刚体，圆环开始在 $y$ 方向上进行压缩变形。从图中最上面的红线可以看出，在 $y$ 方向进行压缩的时候，$x$ 方向开始膨胀拉伸。在 $0.45s$ 的时候，$x$ 和 $y$ 方向同时达到了形变最大，此时速度为零。此后一直到 $0.07s$，由于刚度的作用，圆环恢复形变。

3) $0.07 \sim 0.15$：**能量衰减**

在 $0.07s$ 的时候，圆环几乎恢复到了初始形状，从图中可以看到三条位移曲线几乎都回到了最小位移的地方。在 $0.08s$ 左右，圆环在惯性作用下轻轻弹起到最高处，图中有段 $x$ 对应的曲线稍稍低于坐标轴，那是由于弹到最高处之后，$x$ 方向开始收缩。之后，圆环下落开始第二次碰撞变形。由于阻尼作用的影响，变形过程中能量在一直减小，所以第二次的变形程度比第一次相差甚远。如果采用的是一个剖分均匀对称，无阻尼的物体，这种弹起落下弹起再落下的过程会一直进行下去。

4) $0.15 \sim 0.2$：**失去平衡** 经过多次弹起落下的过程，圆环最终在不平衡结构的作用下，跌倒在平面上，达到最终的平稳状态。

## 5.2　准确性评估

判断一个有限元形变的方法是否正确，在工程上经常会做一些真实的物理实验，比如对某种金属材料的拉伸和压缩，通过应力应变的变化来与仿真结果做比较。然而在生物力学领域，这种方法实现难度很大，因为大部分软组织物质几何不规则，材质不均匀，边界不固定等。真实实验做出的结果往往和理论有一定偏差，尤其我们在模拟变形的时候常常把模型假设成各向同性均匀的物质。所以我们无法和真实的实验做比较，既然无法准确模拟生物力学，那就单纯只比较有限元方法的处理过程是否正确。接下来，我们就要选择一款仿真软件，作为我们的参考标准。

真实世界的仿真是非线性的， ABAQUS 软件被广泛的认为是功能最强的有限元软件。大量的复杂问题可以通过选项块的不同组合很容易的模拟出来。在大部分模拟中，甚至高度非线性问题，用户只需提供一些工程数据，像结构的几何形状、材料性质、边界条件及载荷工况。在一个非线性分析中， ABAQUS 能自动选择相应载荷增量和收敛限度。他不仅能够选择合适参数，而且能连续调节参数以保证在分析过程中有效地得到精确解。用户通过准确的定义参数就能很好的控制数值计算结果。所以在这里我们将对本文的结果与 ABAQUS 的结果做比较以保证有限元方法的准确性。

在变形结果中，我们随机选取了一些点，图 5.3 5.4 5.5 5.6 分别是这些随机点的在 $x$，$y$，$z$ 方向上的位移变化及总位移变化。带 ∘ 的蓝线表示 ABAQUS 的结果，带 + 的红线是本文方法得出的结果，坐标轴附近的绿色小短线代表两者之间的偏差。很清楚可以看到，两条线几乎是重合的，也就是说本文方法的结果是比较准确的。

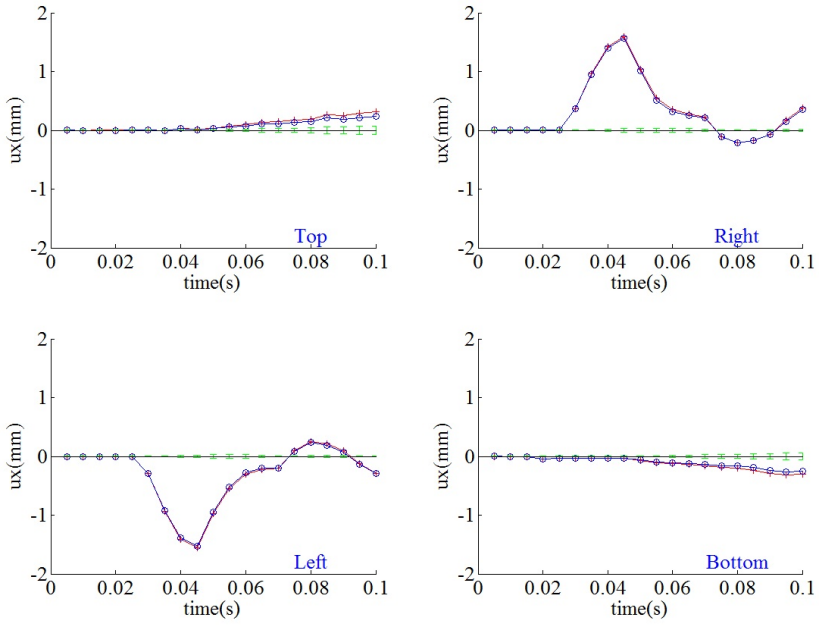



图 5.3:　$x$ 方向的位移

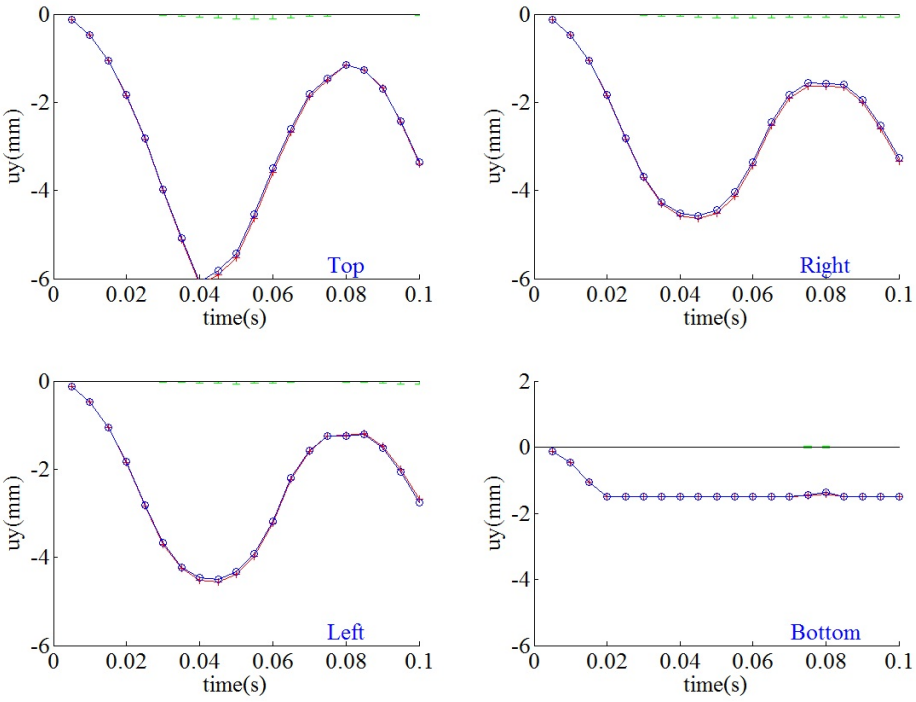



图 5.4: $y$ 方向的位移

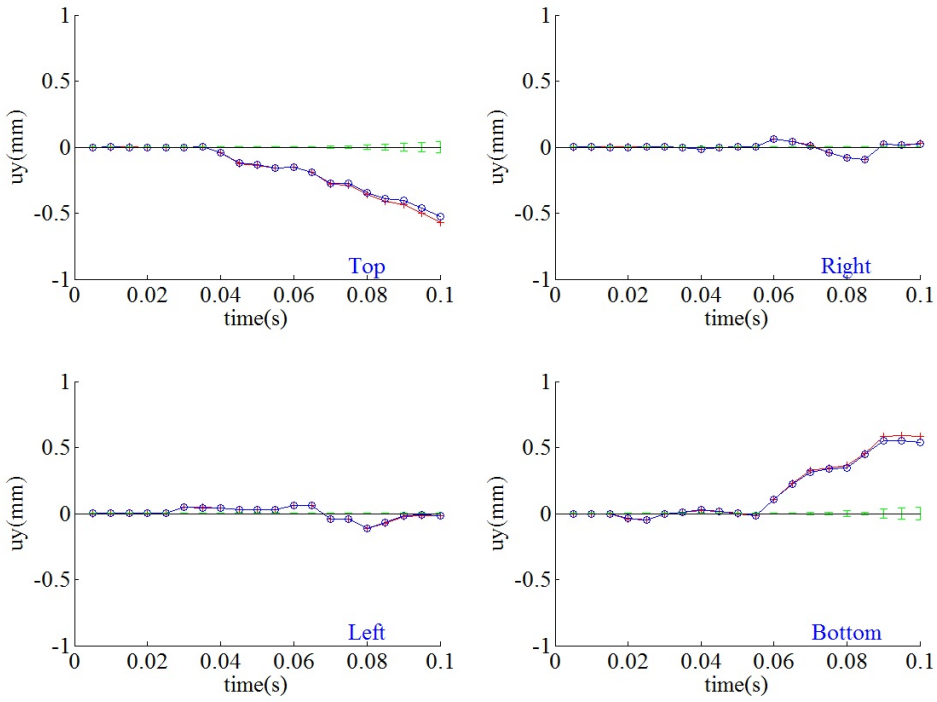



图 5.5: $z$ 方向的位移

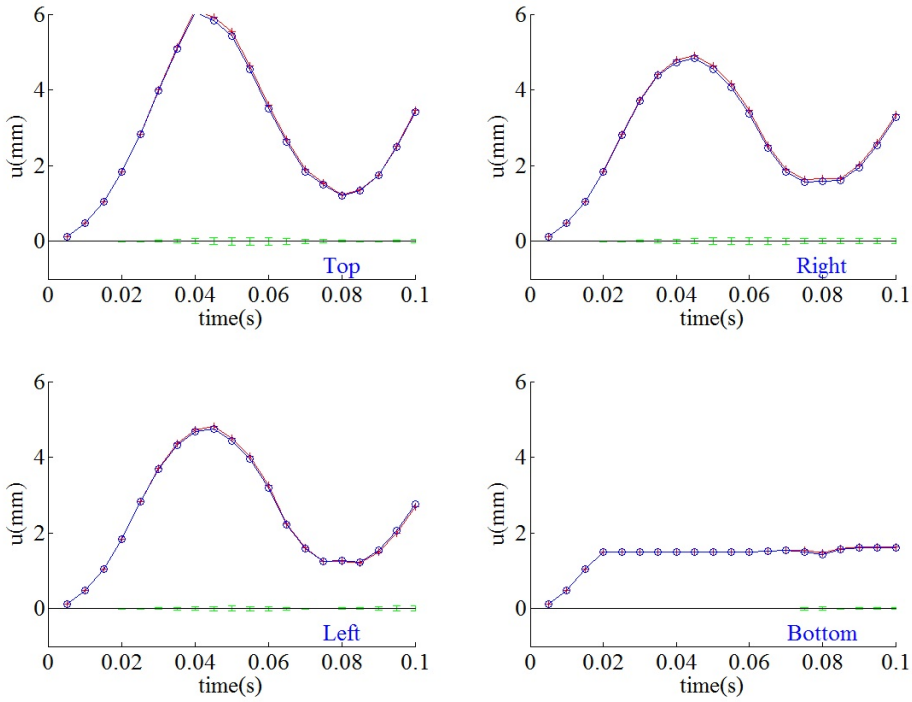



图 5.6: 整体位移

图 5.7 表示的是本文方法与 ABAQUS 计算得出结果的平均偏差。在前 $0.1s$，从 2145 个节点位移平均得到偏差在 $0.1mm$ 左右，而整个模型的大小是在 $20mm$ 左右。也就是说本文基于 GPU 的方法与 ABAQUS 方法之间的偏差小于 0.5% 。

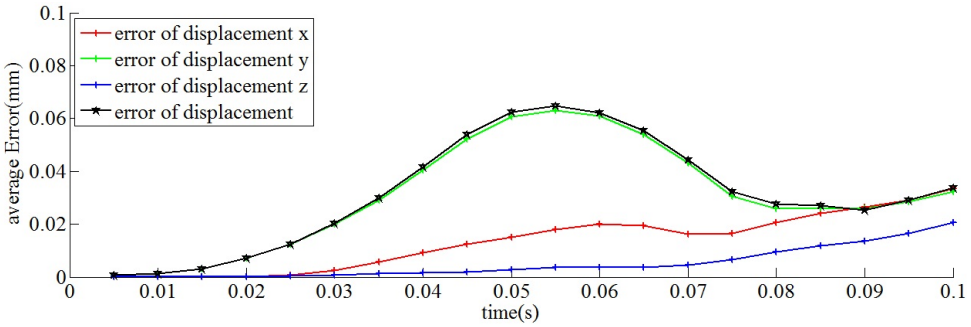



图 5.7: 本文方法与 ABAQUS 所得结果的位移偏差

## 5.3　不同材质的模拟分析

图 5.8 所示三种不同材质的圆环所产生的最大形变。其中 $A$ 的杨氏模量是 $30GPa$，$B$ 的杨氏模量是 $3GPa$，$C$ 的杨氏模量是 $0.3GPa$。三种材质的变形结果说明，本文的 GPU 方法对大形变的模拟确实有效可行，鲁棒性很强。

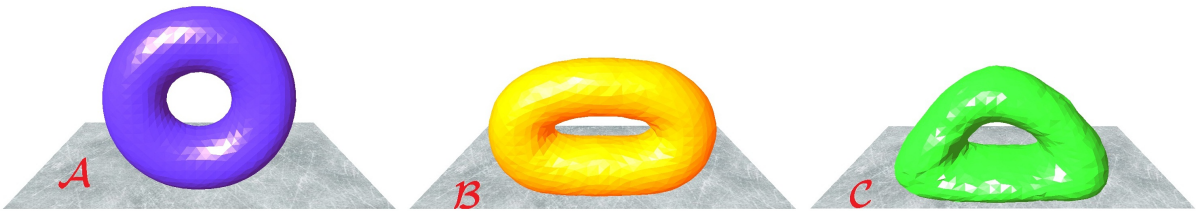



图 5.8: 不同杨氏模量的圆环产生的最大变形

杨氏模量表征了一种弹性材料的刚度特性，而泊松比则反映了其压缩的特性。当泊松比为 0.5 的时候，材料是完全不可压缩的，然而这种物质是不存在的，只有无限接近完全不可压缩。一般我们认为大于 0.49 的材料就是近似不可压缩，泊松比越小，体积越容易被压缩。当泊松比越接近 0.5，物体越难以被压缩变形，如图 5.9 所示，不仅位移会变小，变形周期也在缩短。当泊松比达到 0.4999 时，计算结果已经出错，这是由于此时的迭代时间步长几乎趋近于零。微小的时间步长会导致很大的迭代次数，累计误差已经远远的占据了计算的主要部分，导致结果崩溃。图 5.10 中涵盖了这个世界上几乎所有的材料。泊松比是影响变形速度和稳定性的主要因素。近不可压缩的材质通常要比可压缩材质更难求解，因为它约束了网格的体积，网格只能变形，不能减小体积。此外，在泊松比一定的情况下，小的杨氏模量允许更快的迭代时间步长。

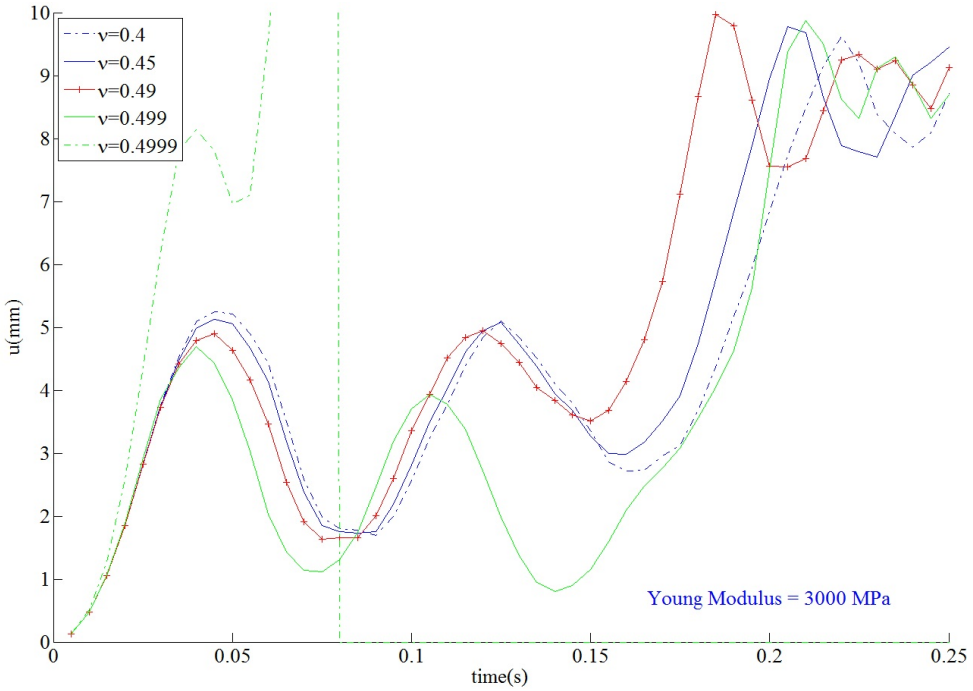



图 5.9: 泊松比对变形结果的影响

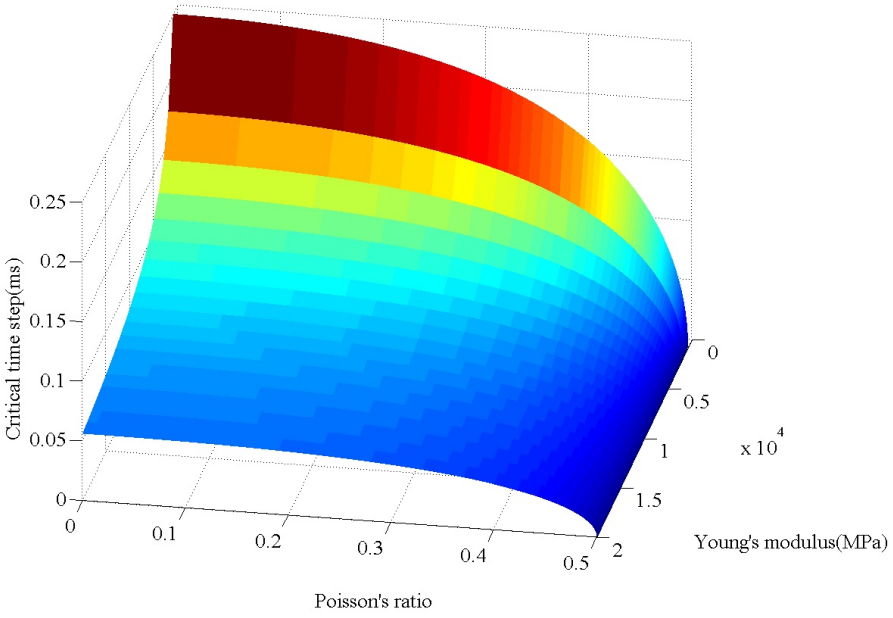



图 5.10: 时间步长与泊松比和杨氏模量的关系

## 5.4　不同碰撞模型的模拟分析

网格剖分之后的圆环不是轴对称的几何体，在经过几次碰撞之后，重心偏离中心线。在 $z$ 轴方向有力矩产生，使得圆环最终倒下。在接触面产生的相对运动由于不同的模型导致不同的摩擦结果。

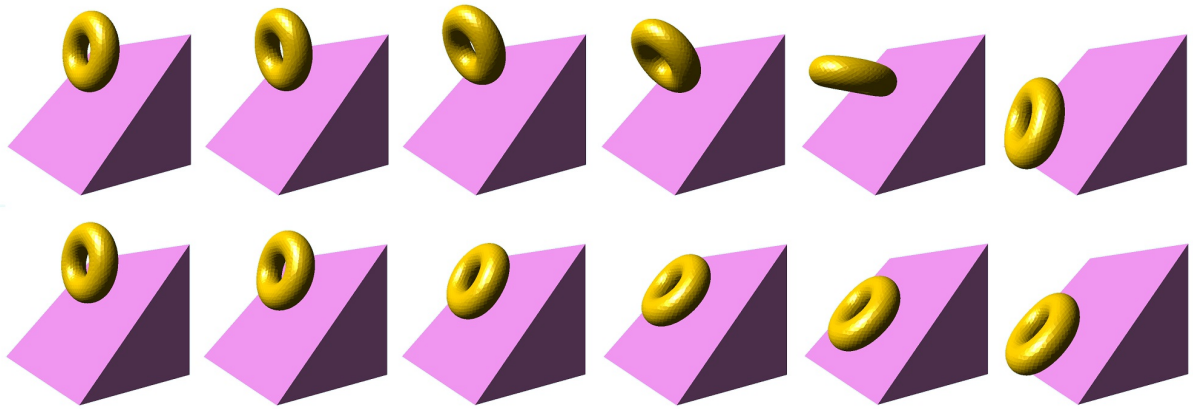


图 5.11: 两种不同的摩擦效果

图 5.11 中所示的是圆环落向一个斜面发生的弹性碰撞。上图中的摩擦属于静态摩擦，圆环接触斜面之后向着斜面的法向弹起，然后翻滚着从斜面落下。这是图 4.4 中所

述模型的结果。而下图中的摩擦属于滑动摩擦，说明斜面的光滑程度很高，圆环碰到斜面之后，迅速沿着斜面开始下滑。也就是图 4.3 中所示的模型。这个例子很好的说明了，本文设计的简单碰撞模型，不仅起到了边界约束的作用，也能简单的表征碰撞摩擦的关系。

## 5.5 网格剖分程度对变形的影响

我们用两种不同剖分程度的模型进行有限元分析比较。第一种模型是直接网格剖分得到的，第二种模型是先对表面网格进行细分然后再进行体网格剖分得到的。第二种模型的表面光滑程度与内部网格稠密度都比第一种要高很多。节点数从 2145 增加到 8395，四面体网格数从 7456 增加到 27918，几乎是原来的四倍。

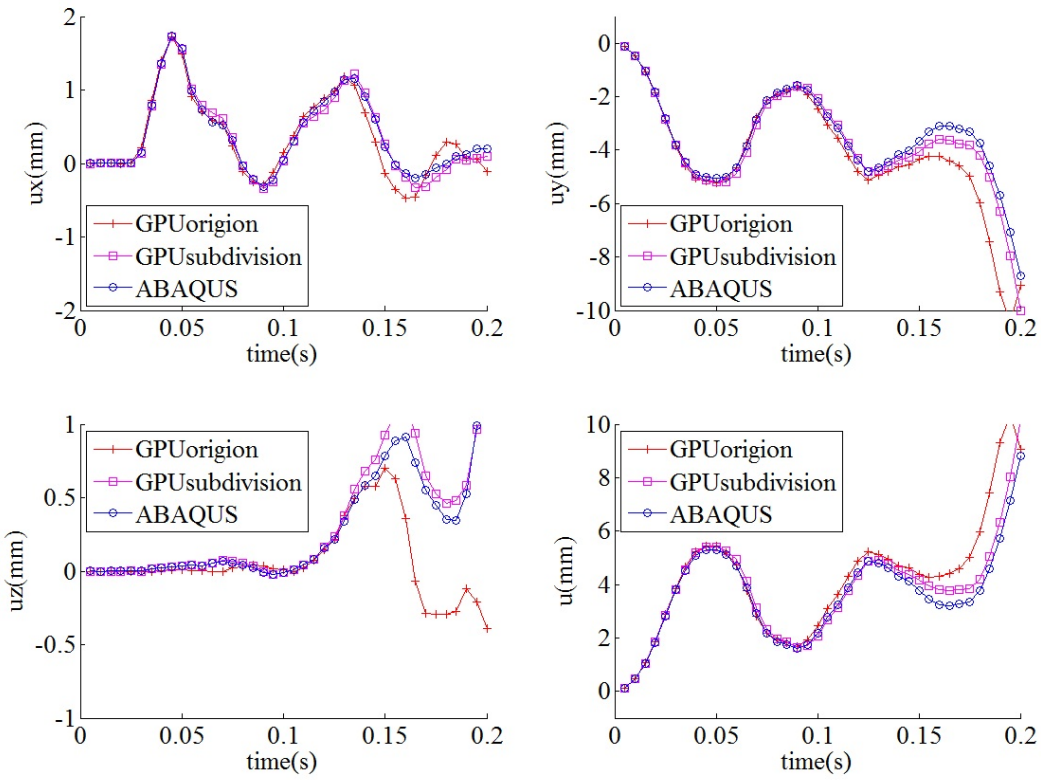



图 5.12: 网格剖分程度不同的模型仿真结果与 ABAQUS 结果比较

从图 5.12 中可以看出，网格加细之后的模型与 ABAQUS 的结果更为接近。

理论上讲，网格剖分越细，有限元的解就越精确。但剖分过细之后带来的问题是网格过小，在压缩变形过程中很容易得到不稳定的结果。同时由于网格体积变小，随之而来的是迭代时间步长的减小。为了达到实时性，每步迭代所消耗的计算时间不能

长于迭代时间步长。也就是说，网格越小，迭代时间步长就越小，单位时间步长内完成计算的可能性就会下降。所以适度划分网格也是有限元变形的重要基础。

## 5.6　计算效率

我们对 GPU 上的算法实现进行了评估，并和 CPU 的程序做了对比。在 CPU 上，计算时间基本和模型网格数成线性关系。而在 GPU 上，性能随着网格数目的增多而提高。这是由于使用 GPU 时可以划分成两部分：一部分时间属于 CPU 占用时间，包括了程序的控制， Kernel 的初始化，以及数据的传输；另一部分是真正 GPU 并行运算的时间。所以当模型规模较小的时候，第一部分时间占用的比例相对较大，程序的高效率不是很明显。当数据量增大时，才能看到明显的效果。表 5.1 所得的时间是 8000 次迭代的平均值。

表 5.1: GPU 算法单次迭代中各项数据计算时间

| 网格 | GPU 时间 $(ms)$ | | | | | CPU |
|---|---|---|---|---|---|---|
| 数目 | 求**M** | 求**f** | 求**u** | 更新**u** | GPU总时间 | 总时间$(ms)$ |
| 7456 | 50.0 | 2280.0 | 54.8 | 500.0 | 2884.8 | 298288 |
| 12584 | 102.8 | 4024.0 | 101.2 | 980.0 | 5208.0 | 666624 |
| 27918 | 190.0 | 6604.0 | 160.4 | 1788.0 | 8742.4 | 1186344 |

## 5.7　本章小结

本章通过比较实验结果中的位移，来证明本文方法的正确性及稳定性。同时对变形过程中的细节问题通过实验结果来加以说明。最后我们给出一些变形结果。

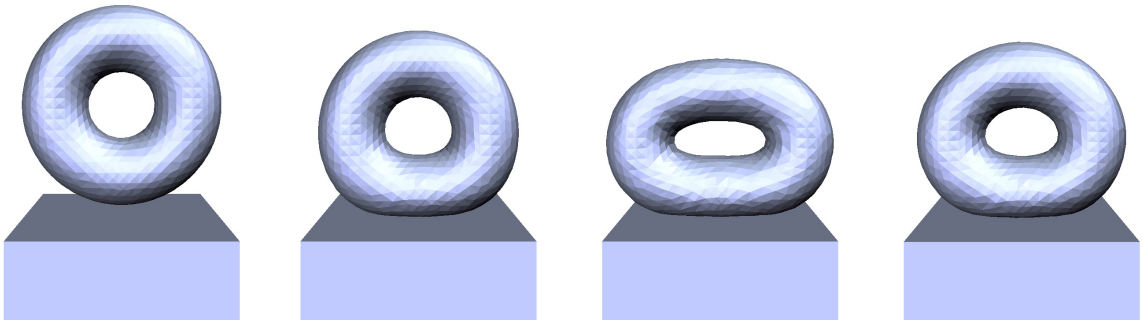


图 5.13: 圆环变形过程

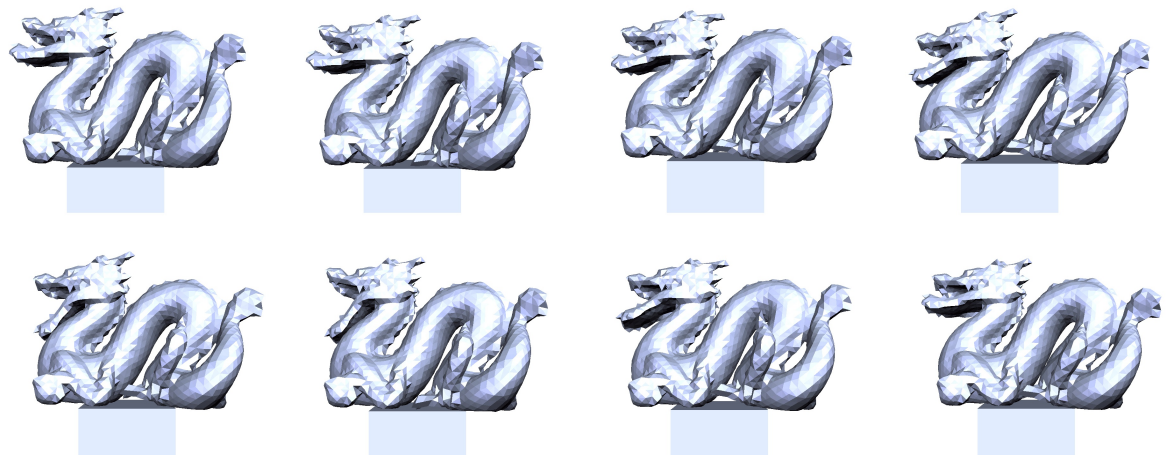


图 5.14: 龙模型：杨氏模量30000 $MPa$ ， 泊松比 0.4

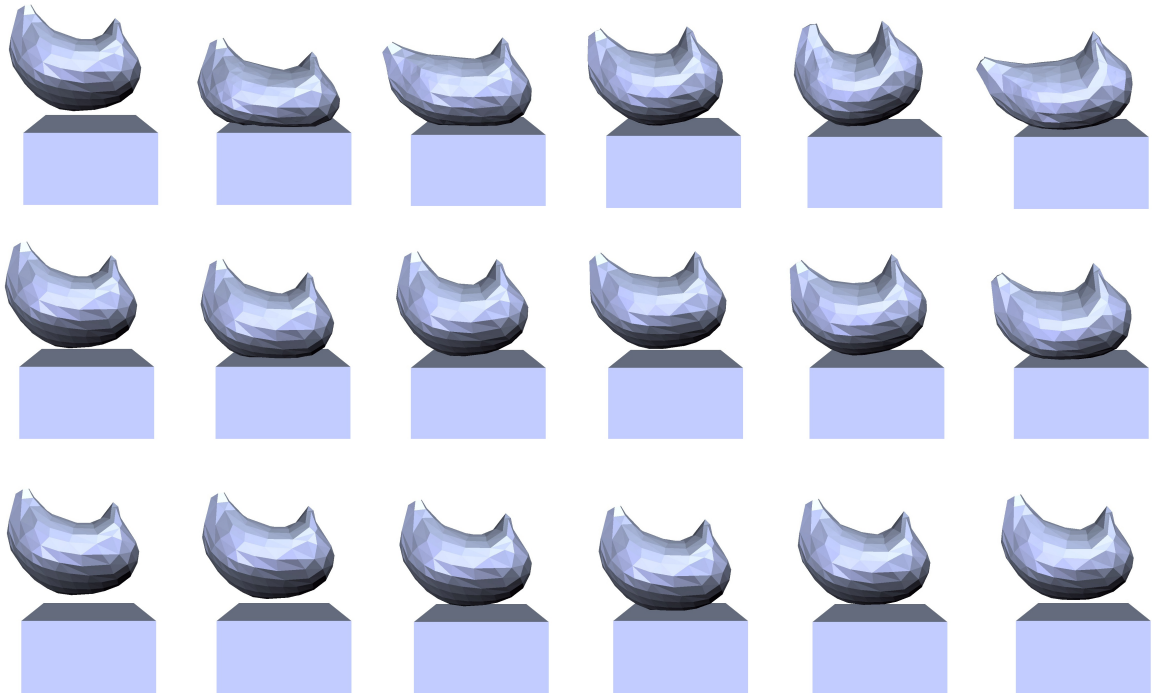


图 5.15: 胃模型1：杨氏模量3000 $MPa$ ， 泊松比 0.49
胃模型2：杨氏模量30000 $MPa$ ， 泊松比 0.2
胃模型3：杨氏模量300000 $MPa$ ， 泊松比 0.02

# 第六章　总结与展望

## 6.1　本文工作总结

本文在网格简化、曲面细分、以及网格剖分的基础上，提出了一种基于 GPU 的非线性有限元处理方法，用于解决软组织自由形变的问题，同时给出了一种快速碰撞的高效模型。本文主要研究内容如下：

1. 全文首先描述了虚拟手术过程中的关键技术，对物体形变部分做了详细的调研工作，分别介绍了以质点-弹簧模型和有限元模型为主的变形方法，并比较其优劣性。

2. 详细介绍了做有限元计算之前对模型网格结构的处理方法，包括基于 Garland 算法的网格简化方法、基于 Loop 细分曲面的网格细分方法以及基于 Delaunay 准则的网格剖分方法，并对其一一进行实现。

3. 针对非线性有限元中的几何非线性和物理非线性，分别提出一种计算处理方法，将非线性的模型线性化，适合于计算机以及 GPU 来计算。此外对有限元的建立以及求解做了较为详细的说明。

4. 将非线性有限元算法在 GPU 上进行实现，并且提出一套碰撞方案，适合 GPU 的快速计算。

5. 通过仿真实验模拟，并与 ABAQUS 软件结果做比较，验证了本文方法的可行性与稳定性，对有限元方法中的一些参数和模型做了实验数据分析。

## 6.2　今后研究展望

虚拟手术仿真系统将会有一个长远的发展，其中形变模拟需要解决的问题还有：

1. 生理功能仿真。现有的形变模型最多只能提供力反馈功能。然而，人体软组织具有特殊的生理功能，在手术进行过程中，手术交互的各种动作对血流速度、压力、心跳和体温等生理特征会产生较大影响，如何能把这些生理功能集成到组织器官模型中是个严峻的考验。我们的物理模型比较简单，没有实现多组织的模拟，针对混合在一起的组织，单一的弹性模型已经不再适用，需要去探索更多的方法。

2. 手术器械与组织器官的复杂交互模拟。现有的模型对手术器械采取了简化处理，实际情况下这种交互的过程是非常复杂的，复杂的器械会带来复杂的形变效果，准确地模拟手术操作过程将极大的提高仿真效果；

3. 实时图形渲染技术。为了实现更真实的变形效果，现有的基于多边形图形渲染的技术会随着几何对象复杂度的增加而提高。高效的渲染组织器官模型也将是人们关注的一个热点问题。目前的想法是，通过有限元模拟出组织变形情况之后，运用

Tessellation 和 Displacement Mapping 技术[92]，将复杂的表面细节加入到简单的变形模型中，但是又不会增加有限元计算网格，只是在结果中直接利用图形着色器进行渲染，达到真正逼真的效果显示。

虚拟手术涉猎知识非常广泛，可研究的关键技术非常多，仅形变这一内容，就因为组织器官的不同而产生了不同的方法，但目标都是一样的：更快速更准确。随着计算机与医学技术的不断进步，这一多学科交叉技术必将在临床上发挥极其强大作用。

# 参 考 文 献


[1] Ian Stavness, John E. Lloyd, Yohan Payan, and Sidney Fels. Coupled hard-soft tissue simulation with contact and constraints applied to jaw-tongue-hyoid dynamics. *International Journal for Numerical Methods in Biomedical Engineering*, 27(3):367–390, March 2011.

[2] S. F. Johnsen, Z. A. Taylor, M. Clarkson, S. Thompson, M. Hu, K. Gurusamy, B. Davidson, D. J. Hawkes, and S. Ourselin. Explicit contact modeling for surgical computer guidance and simulation. In David R. Holmes III and Kenneth H. Wong, editors, *Proceedings of SPIE*, volume 8316, pages 831623–831623–9, February 2012.

[3] G. Hirota, S. Fisher, A. State, C. Lee, and H. Fuchs. An implicit finite element method for elastic solids in contact. In *Proceedings Computer Animation 2001. Fourteenth Conference on Computer Animation (Cat. No.01TH8596)*, pages 136–254. IEEE Comput. Soc, 2001.

[4] Qiong Wang, Hui Chen, Wen Wu, Hai Yang Jin, and Pheng Ann Heng. A virtual surgical simulator for mandibular angle reduction based on patient specific data. In *2012 IEEE Virtual Reality (VR)*, pages 85–86. IEEE, March 2012.

[5] Qiong Wang, Hui Chen, Wen Wu, Jing Qin, and Pheng Ann Heng. Impulse-Based Rendering Methods for Haptic Simulation of Bone-Burring. *IEEE Transactions on Haptics*, 5(4):344–355, January 2012.

[6] Q Wang, H Chen, W Wu, H Jin, and P Heng. Real-time Mandibular Angle Reduction Surgical Simulation with Haptic Rendering. *IEEE transactions on information technology in biomedicine : a publication of the IEEE Engineering in Medicine and Biology Society*, 16(6):1105–1114, September 2012.

[7] Guillem Pratx and Lei Xing. GPU computing in medical physics: A review. *Medical Physics*, 38(5):2685, 2011.

[8] Jianying Li, Yu Guo, Heye Zhang, and Yongming Xie. A master-slave robotic simulator based on GPUDirect. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5203–5207. IEEE, October 2012.

[9] Theodore Kim and Doug L James. Physics-based character skinning using multidomain subspace deformations. *IEEE transactions on visualization and computer graphics*, 18(8):1228–40, August 2012.

[10] D. Kravtsov, O. Fryazinov, V. Adzhiev, A. Pasko, and P. Comninos. Embedded Implicit Stand-Ins for Animated Meshes: A Case of Hybrid Modelling. *Computer Graphics Forum*, 29(1):128–140, March 2010.

[11] Sarah F. Gibson. 3D chainmail. In *Proceedings of the 1997 symposium on Interactive 3D graphics - SI3D '97*, pages 149–ff., New York, New York, USA, 1997. ACM Press.

[12] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. *ACM SIGGRAPH Computer Graphics*, 21(4):205–214, August 1987.

[13] Gaizka San-Vicente, Iker Aguinaga, and Juan Tomás Celigüeta. Cubical Mass-Spring Model design based on a tensile deformation test and nonlinear material model. *IEEE transactions on visualization and computer graphics*, 18(2):228–41, February 2012.

[14] CE Etheredge. A Parallel Mass-Spring Model for Soft Tissue Simulation with Haptic Rendering in CUDA. 2011.

[15] Peter E Hammer, Michael S Sacks, Pedro J del Nido, and Robert D Howe. Mass-spring model for simulation of heart valve tissue mechanical behavior. *Annals of biomedical engineering*, 39(6):1668–79, June 2011.

[16] Jing Qin, Wai-Man Pang, Yim-Pan Chui, Tien-Tsin Wong, and Pheng-Ann Heng. A Novel Modeling Framework for Multilayered Soft Tissue Deformation in Virtual Orthopedic Surgery. *Journal of Medical Systems*, 34(3):261–271, December 2010.

[17] Panagiotis A Foteinos and Nikos P Chrisochoides. High-quality multi-tissue mesh generation for finite element analysis. pages 159–169, 2013.

[18] Karol Miller, Grand Joldes, Dane Lance, and Adam Wittek. Total Lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation. *Communications in Numerical Methods in Engineering*, 23(2):121–134, August 2006.

[19] Z a Taylor, M Cheng, and S Ourselin. High-speed nonlinear finite element analysis for surgical simulation using graphics processing units. *IEEE transactions on medical imaging*, 27(5):650–63, May 2008.

[20] Zeike A Taylor, Stuart Crozier, and Sebastien Ourselin. A reduced order explicit dynamic finite element algorithm for surgical simulation. *Medical Imaging, IEEE Transactions on*, 30(9):1713–1721, 2011.

[21] GR Joldes, A Wittek, and Karol Miller. Real-Time Nonlinear Finite Element Computations on GPU: Handling of Different Element Types. *Computational Biomechanics for Medicine*, pages 73–80, 2011.

[22] K Noe and T Sø rensen. *Solid Mesh Registration for Radiotherapy Treatment Planning*, volume 5958 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[23] S Niroomandi, I Alfaro, E Cueto, and F Chinesta. Accounting for large deformations in real-time simulations of soft tissues based on reduced-order models. *Computer methods and programs in biomedicine*, 105(1):1–12, January 2012.

[24] P. Wang, a.a. Becker, I.a. Jones, a.T. Glover, S.D. Benford, C.M. Greenhalgh, and M. Vloeberghs. Virtual reality simulation of surgery with haptic feedback based on the boundary element method. *Computers & Structures*, 85(7-8):331–339, April 2007.

[25] Peter Xiaoping Liu. A non-linear viscoelastic tensor-mass model for surgery simulation. In *2009 IEEE International Workshop on Haptic Audio visual Environments and Games*, pages 48–53. IEEE, November 2009.

[26] F. S. Sin, D Schroeder, and J. Barbič. Vega: Non-Linear FEM Deformable Object Simulator. *Computer Graphics Forum*, 32(1):36–48, February 2013.

[27] Matthias Müller and Nuttapong Chentanez. Solid simulation with oriented particles. *ACM Transactions on Graphics*, 30(4):1, July 2011.

[28] Jing Qian and Jia Lu. Point-cloud method for image-based biomechanical stress analysis. *International Journal for Numerical Methods in Biomedical Engineering*, 27(10):1493–1506, October 2011.

[29] S. De and K. J. Bathe. The method of finite spheres. *Computational Mechanics*, 25(4):329–345, April 2000.

[30] S. De, J.-W. Hong, and K. J. Bathe. On the method of finite spheres in applications: towards the use with ADINA and in a surgical simulator. *Computational Mechanics*, 31(1-2):27–37, May 2003.

[31] Suvranu De, Jung Kim, Yi-Je Lim, and Mandayam a. Srinivasan. The point collocation-based method of finite spheres (PCMFS) for real time surgery simulation. *Computers & Structures*, 83(17-18):1515–1525, June 2005.

[32] Barbara Solenthaler and Markus Gross. Two-scale particle simulation. *ACM Transactions on Graphics*, 30(4):1, July 2011.

[33] Nadir Akinci, Markus Ihmsen, Gizem Akinci, Barbara Solenthaler, and Matthias Teschner. Versatile rigid-fluid coupling for incompressible SPH. *ACM Transactions on Graphics*, 31(4):1–8, July 2012.

[34] Xia Jin, Grand Roman Joldes, Karol Miller, King H Yang, and Adam Wittek. Meshless algorithm for soft tissue cutting in surgical simulation. *Computer methods in biomechanics and biomedical engineering*, (January 2013):37–41, September 2012.

[35] Ashley Horton, Adam Wittek, Grand Roman Joldes, and Karol Miller. A meshless Total Lagrangian explicit dynamics algorithm for surgical simulation. *International Journal for Numerical Methods in Biomedical Engineering*, 26(8):977–998, March 2010.

[36] Benjamin Gilles, Guillaume Bousquet, Francois Faure, and Dinesh K. Pai. Frame-based elastic models. *ACM Transactions on Graphics*, 30(2):1–12, April 2011.

[37] François Faure, Benjamin Gilles, Guillaume Bousquet, and Dinesh K. Pai. Sparse meshless models of complex deformable solids. *ACM Transactions on Graphics*, 30(4):1, July 2011.

[38] Hugues Hoppe. Progressive meshes. pages 99–108, 1996.

[39] A.D. Kalvin and R.H. Taylor. Superfaces: polygonal mesh simplification with bounded error. *IEEE Computer Graphics and Applications*, 16(3):64–77, May 1996.

[40] J Rossignac and P Borrel. *Multi-resolution 3D approximations for rendering complex scenes*. 1993.

[41] David Luebke and Carl Erikson. View-dependent simplification of arbitrary polygonal environments. pages 199–208, 1997.

[42] FJM Schmitt, BA Barsky, and WH Du. An adaptive subdivision method for surface-fitting from sampled data. *ACM SIGGRAPH Computer Graphics*, 1986.

[43] Paul S Heckbert and Michael Garland. Survey of polygonal surface simplification algorithms. Technical report, DTIC Document, 1997.

[44] William J Schroeder, Jonathan A Zarge, and William E Lorensen. Decimation of triangle meshes. 26(2):65–70, 1992.

[45] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. pages 19–26, 1993.

[46] B Hamann. A data reduction scheme for triangulated surfaces. *Computer Aided Geometric Design*, 1994.

[47] A.D. Kalvin and R.H. Taylor. Superfaces: polygonal mesh simplification with bounded error. *IEEE Computer Graphics and Applications*, 16(3):64–77, May 1996.

[48] Julie C Xia and Amitabh Varshney. Dynamic view-dependent simplification for polygonal models. pages 327–334, 1996.

[49] Hugues Hoppe. Smooth view-dependent level-of-detail control and its application to terrain rendering. pages 35–42, 1998.

[50] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. *Proceedings of the 24th annual conference on Computer graphics and interactive techniques - SIGGRAPH '97*, pages 209–216, 1997.

[51] Edwin Catmull and James Clark. Recursively generated b-spline surfaces on arbitrary topological meshes. *Computer-aided design*, 10(6):350–355, 1978.

[52] D Doo and M Sabin. Behaviour of recursive division surfaces near extraordinary points. *Computer-Aided Design*, 10(6):356–360, November 1978.

[53] C Loop. Smooth subdivision surfaces based on triangles. 1987.

[54] Nira Dyn, David Levine, and John A. Gregory. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Transactions on Graphics*, 9(2):160–169, April 1990.

[55] Jeff Bolz and P Schröder. Evaluation of subdivision surfaces on programmable graphics hardware. *submitted for publication*, 2003.

[56] Kun Zhou, X Huang, W Xu, B Guo, and H Shum. Direct manipulation of subdivision surfaces on GPUs. *ACM Transactions on Graphics*, 26(3), 2007.

[57] Minho Kim and Jörg Peters. Realtime loop subdivision on the GPU. In *ACM SIGGRAPH 2005 Posters on - SIGGRAPH '05*, page 123, New York, New York, USA, 2005. ACM Press.

[58] Le-jeng Shiue, Ian Jones, and Jörg Peters. A realtime GPU subdivision kernel. *ACM Transactions on Graphics*, 24(3):1010, July 2005.

[59] Matthias Nieß ner, Charles Loop, Mark Meyer, and Tony Derose. Feature-adaptive GPU rendering of Catmull-Clark subdivision surfaces. *ACM Transactions on Graphics*, 31(1):1–11, January 2012.

[60] Jonathan Richard Shewchuk. Tetrahedral mesh generation by delaunay refinement. pages 86–95, 1998.

[61] Jonathan Richard Shewchuk. General-Dimensional Constrained Delaunay and Constrained Regular Triangulations, I: Combinatorial Properties. *Discrete & Computational Geometry*, 39(1-3):580–637, March 2008.

[62] Mark A. Yerry and Mark S. Shephard. Automatic three-dimensional mesh generation by the modified-octree technique. *International Journal for Numerical Methods in Engineering*, 20(11):1965–1990, November 1984.

[63] Charlie C.L. Wang. Direct extraction of surface meshes from implicitly represented heterogeneous volumes. *Computer-Aided Design*, 39(1):35–50, January 2007.

[64] SH Lo. Volume discretization into tetrahedra-ii. 3d triangulation by advancing front approach. *Computers & Structures*, 39(5):501–511, 1991.

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

[66] Olivier Comas and Zeike A Taylor. Efficient nonlinear FEM for soft tissue modelling and its GPU implementation within the open source framework SOFA. pages 1–12, 2009.

[67] Grand Roman Joldes, Adam Wittek, and Karol Miller. An adaptive Dynamic Relaxation method for solving nonlinear finite element problems. Application to brain shift estimation. *International journal for numerical methods in biomedical engineering*, 27(2):173–185, February 2011.

[68] Apichart Boonma. Analytical Modeling and Evaluation of Microneedles Apparatus with Deformable Soft Tissues for Biomedical Applications. *Computer-Aided Design and Applications*, 10(1):139–157, August 2013.

[69] Jing Qin, Jing Zhang, Chee-Kong Chui, Wei-Min Huang, Tao Yang, Wai-Man Pang, Venkatesh Sudhakar, and Stephen Chang. A simulation framework for estimating wall stress distribution of abdominal aortic aneurysm. *IEEE Engineering in Medicine and Biology Society*, 2011:900–3, January 2011.

[70] Shun Li, Jing Qin, Jixiang Guo, Yim-Pan Chui, and Pheng-Ann Heng. A Novel FEM-Based Numerical Solver for Interactive Catheter Simulation in Virtual Catheterization. *International journal of biomedical imaging*, 2011:815246, January 2011.

[71] Christian Dick, Joachim Georgii, and Rüdiger Westermann. A real-time multigrid finite hexahedra method for elasticity simulation using CUDA. *Simulation Modelling Practice and Theory*, 19(2):801–816, February 2011.

[72] LH He. Stress and deformation in soft elastic bodies due to intermolecular forces. *Journal of the Mechanics and Physics of Solids*, 2013.

[73] Seyed Reza Mousavi, Iman Khalaji, Ali Sadeghi Naini, Kaamran Raahemifar, and Abbas Samani. Statistical finite element method for real-time tissue mechanics analysis. *Computer methods in biomechanics and biomedical engineering*, 15(6):595–608, January 2012.

[74] Muhammad Tanveer and Jean W. Zu. Non-linear vibration of hyperelastic axisymmetric solids by a mixed p-type method. *International Journal of Non-Linear Mechanics*, 47(4):30–41, May 2012.

[75] KJ Bathe. *Finite element procedures*. 1996.

[76] Joo Kuan-Jung and Edward L. Wilson. An adaptive finite element technique for structural dynamic analysis. *Computers & Structures*, 30(6):1319–1339, January 1988.

[77] Grand Roman Joldes, Adam Wittek, and Karol Miller. Stable time step estimates for mesh-free particle methods. *International Journal for Numerical Methods in Engineering*, 91(4):450–456, July 2012.

[78] Khronos Group. *OpenGL 4.3 API Reference Card*. 2012.

[79] *http://cudazone.nvidia.cn/what-cuda/.*

[80] Dave Shreiner, Graham Sellers, John Kessenich, and Bill Licea-Kane. *The OpenGL Programming Guide 8th Edition.* 2012.

[81] Geoffrey Irving, Craig Schroeder, and Ronald Fedkiw. Volume conserving finite element simulations of deformable models. *ACM Transactions on Graphics*, 26(3):13, July 2007.

[82] G Zachmann and E Langetepe. Geometric data structures for computer graphics. *Tutorial at ACM SIGGRAPH*, 2003.

[83] Fabio Ganovelli, John Dingliana, and Carol O'Sullivan. Buckettree: Improving collision detection between deformable objects. 11, 2000.

[84] Jérémie Allard, François Faure, Hadrien Courtecuisse, Florent Falipou, Christian Duriez, and Paul G. Kry. Volume contact constraints at arbitrary resolution. *ACM Transactions on Graphics*, 29(4):1, July 2010.

[85] Jan Klein and Gabriel Zachmann. Adb-trees: Controlling the error of time-critical collision detection. pages 19–21, 2003.

[86] Laks Raghupathi, Laurent Grisoni, François Faure, Damien Marchal, M-P Cani, and Christophe Chaillou. An intestinal surgery simulator: Real-time collision processing and visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 10(6):708–718, 2004.

[87] Sarah F Frisken, Ronald N Perry, Alyn P Rockwood, and Thouis R Jones. Adaptively sampled distance fields: a general representation of shape for computer graphics. pages 249–254, 2000.

[88] A Fuhrmann, G Sobotka, and C Groß. Distance fields for rapid collision detection in physically based modeling. *Proceedings of GraphiCon 2003*, 2003.

[89] David I. W. Levin, Joshua Litven, Garrett L. Jones, Shinjiro Sueda, and Dinesh K. Pai. Eulerian solid simulation with contact. *ACM Transactions on Graphics*, 30(4):1, July 2011.

[90] Danny M. Kaufman, Shinjiro Sueda, Doug L. James, and Dinesh K. Pai. Staggered projections for frictional contact in multibody systems. *ACM Transactions on Graphics*, 27(5):1, December 2008.

[91] Yu Guo. Fall over or sliding down? In *SIGGRAPH Asia 2012 Posters on - SA '12*, volume 5, page 1, New York, New York, USA, 2012. ACM Press.

[92] Charles Loop, Scott Schaefer, Tianyun Ni, and Ignacio Castaño. Approximating subdivision surfaces with Gregory patches for hardware tessellation. *ACM Transactions on Graphics*, 28(5):1, December 2009.

# 致　　谢

值此论文完成之际，谨在此向多年来给予我关心和帮助的老师、同学、朋友和家人表示衷心的感谢!

首先感谢我的导师王平安教授，给我一个登上计算机图形学的舞台，这是我学生时代的结束，也是我新的生活的起点。王老师既是中科院深圳先进技术研究院的老师，又是香港中文大学的老师。正是借助王老师的帮助，让我结识了不同社会制度下的学生，感受到了不同环境中的学习生活状态，给我的未来之路提供了一个新的方向。

感谢秦博，在我毕业的最后几个月中耐心指导，从研究方法到论文写作，让我受益匪浅，希望毕业之后还能继续合作写文章。

感谢永明，从我在北京上课开始，对我的不断鼓励。总是在我绝望的时候给我带来动力，希望你的公司越办越好。感谢张博及张博的两名客座学生李伟方和刘占鑫，在有限元方法上给了我很大的帮助，能一起讨论问题感觉非常好。

感谢实验室的建英，帮助我完成网格模型的建立，我还学会了 Maya 的基本操作。感谢平姐，在英文写作上面的帮忙，没的说，写东西你真的很有水平的。感谢港中大的伟明和明强，和你们讨论新的想法，“经营”港深两地的“货物”运输。感谢Cool，让我深刻认识到 $\alpha$ 波的神奇。感谢 JY、ZQ WANG、SS，学习之余一起探讨各种问题、一起打球、一起打牌、一起打饭，非常感谢。感谢室友，卧聊到深夜。还有院里其他的同学们，一起走过这段人生。

感谢谢科、亮亮，在我第一次出国的时候对我的各种关照和帮助，通过你们我认识了很多学术圈内的同行和大牛。感谢 Siggraph Asia，通过你，我对自己选择的这个领域愈加的喜欢，我以后一定会经常光顾的。

谢谢爸爸妈妈以及关心我的亲人们，对不起让你们每天担心着我，对不起我都这么大了还不能让你们好好歇歇，对不起接下来的日子里还是不能陪伴在你们的身边，你们是我坚强的后盾。

谨以本文献给关心着我的人!

# 作 者 简 历

## 基本情况

**姓名：**郭煜　　**性别：**男　　**出生日期：**1987年10月　　**籍贯：**山西

## 教育经历

2010年9月— 2013年5月，中国科学院深圳先进技术研究院，硕士

2006年9月— 2010年7月，中南大学，本科

## 【攻读硕士学位期间发表的论文】

[ 1 ] Jianying Li, **YuGuo**, Heye Zhang, and Yongming Xie. A master-slave robotic simulator based on GPUDirect. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'12)*, pages 5203 - 5207. IEEE, October 2012. [EI] doi:10.1109/IROS.2012.6385808

[ 2 ] **Yu Guo**. Fall over or sliding down? *SIGGRAPH Asia 2012 Posters (SA'12)*, volume 5, page 1, New York, New York, USA, 2012. ACM Press. [EI] doi:10.1145/2407156.2407162

[ 3 ] **郭煜**, 秦璟. 虚拟手术中软组织形变的综述. 集成技术, 第二卷, 第二期, 2013年3月. 科学出版社. [EI]

[ 4 ] **Yu Guo**, Heng Pheng-Ann. Surface Contact Deformation of Soft Body with Nonlinear Finite Element Method. *The 19th ACM Symposium on Virtual Reality Software and Technology (VRST'13).* [EI] (To be submitted)

## 【攻读硕士学位期间参加的科研项目】

[ 1 ] 虚拟骨科手术机器人模拟训练系统，深圳市配套项目