摘 要

虚拟手术是虚拟现实技术在现代医学领域中的应用。基于虚拟现实技术的手术仿真系统具有很好的应用前景。视觉和触觉渲染是虚拟手术系统与用户交互的两种重要方式。逼真、实时的视觉和触觉渲染可以增强系统的真实性和沉浸感。

人体软组织形变在真实手术中十分常见：包括器械接触软组织引起的小范围的弹性形变，如皮肤的凹陷、凸起；以及由肢体运动产生的大范围的形变，如膝关节运动等。因此，能否有效的对人体器官组织的形变进行模拟，对于提高虚拟手术的真实感至关重要。由于虚拟手术系统具有实时交互特性，形变模拟不但要具有很好的逼真性，而且要具有足够的实时性。

本文研究了模拟虚拟手术中实时大范围形变的技术。与小范围弹性形变不同的是，大范围形变的形变程度大，形变往往要涉及到整个模型。对于模拟表面局部形变，我们常常采用计算量较小的线弹性形变模型。对于模拟肢体运动等等的大范围形变，为避免线弹性模型的失真，一般采用开销较大的非线性物理模型。并且，在大范围形变过程中，模型的大部分顶点都要参与计算，这更加大了系统的计算负荷，对系统的实时性存在负面影响。为此，必须研究能够同时满足逼真性和实时性要求的大范围形变方法。

基于几何的形变技术只要求在视觉上满足物体的物理特性，无需物理模型，因而能够快速实时的模拟大范围形变，并且一般没有数值稳定性的问题，同时在一定程度上满足视觉逼真性的要求。本文采用了计算机动画中基于几何的形变技术，提出了一种根据横截面构建骨架的方法和两种适合于模拟虚拟手术的大范围形变方法，基于柱坐标平滑变化的方法和改进的梯度域方法。

本文针对膝关节屈伸运动引起的形变，提出了一种基于柱坐标平滑变化的形变方法。该方法根据骨架与膝关节周围网格之间的对应关系，将骨架运动的变换平滑的映射到以柱坐标表示的网格顶点上，得到表面的形变效果。该方法能够较好地保持模型表面的整体形状和局部细节，并且计算开销很小。实验证明，此方法能够满足虚拟手术对逼真性和实时性的要求。

本文针对现有梯度域方法在某些情况下不能保持模型体积和难以反映模型表面材质特性的不足，提出了一种改进的梯度域方法。该方法通过修改能量函数中的权值，实现了表面材料相关的形变；通过在容易引起体积收缩的部位（如关节），加入虚拟节点构建体图，采用体微分坐标实现了保体积的形变。

本文最后介绍了上述形变技术在虚拟膝关节镜手术系统中的应用，进一步验证了本文算法的有效性。

关键字：虚拟手术，三角网格，形变模拟，基于几何的形变模拟，微分坐标

ABSTRACT

Virtual surgery is an application of Virtual Reality (VR) in medicine. VR-based surgical simulator has shown good prospect in application. Visual and haptic feedbacks are two important ways of interaction between visual surgery system and its user. Realistic and real-time visual and haptic feedbacks may enhance the immersion of a visual surgery system.

Deformation of human soft tissue is a common phenomenon in real surgery. Generally, there are two types of deformation in surgery. The one is small deformation caused by contact between medical instruments and human organs, such as sunk or bulging of human skin. The other one is large deformation induced by body moving like articulated deformation. Simulation to these deformation phenomena is essential to the visual realism of a simulator. Simulation of deformation in virtual surgery must be both realistic and real-time.

We in this thesis research the large deformation used in virtual surgery. Different from small one, large deformation performs large scale deformations and hence involve the entire mesh model. Physically-based models, especially the linear elastic models, have been widely employed in simulating local and small deformation on surface meshes due to their similarity. However, when used to simulate large deformations, nonlinear elastic models should be considered since linear models often present many artifacts. Nonlinear models are often too expensive to ensure real-time simulation. Moreover, the problem scale of large deformation is often quite large. All these facts prevent us from using physically-based models.

We find geometrically-based methods are suitable for real-time large deformation modeling due to their efficiency, robustness, and physically plausible feature, which can meet the requirements of virtual surgery. We use the geometrically-based methods to simulate large deformation in virtual surgery. Geometrically-based deformation methods have been extensively researched in the context of computer animation.

We propose a skeleton-driven deformation method based on smooth interpolation of transformation with cylindrical coordinates. We first extract the skeleton of a mesh. The deformation of the skin mesh is driven by the moving skeleton. The surface details can be preserved well due to our smooth interpolation scheme. Besides, our model is very efficient and can be performed in real-time.

Gradient domain method based on differential coordinates is another popular geometrically-based mesh deformation method. We augment the existed gradient domain method with two features and propose a improved gradient domain method. To make the deformation material-aware, we adjust the weights in the energy function. We add a virtual node near the model part that may introduce the most volume degrading during deformation, such as joint part in the articulated deformation. By constructing a

volume graph around this virtual node, our deformation is calculated using the volume differential coordinates, which can effectively preserve the volume of the joint part of a mesh.

We introduce the application of the proposed methods in our virtual surgery system. The results show the effectiveness and efficiency of our deformation method.

Key Words: Virtual Surgery, Triangular Mesh, Deformation Modeling, Geometrically-based Deformation Modeling, Differential Coordinates

表 目 录

图 目 录

独 创 性 声 明

本人声明所呈交的学位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，论文中不包含其他人已经发表和撰写过的研究成果，也不包含为获得国防科学技术大学或其它教育机构的学位或证书而使用过的材料。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谢意。

学位论文题目：虚拟膝关节手术中的大范围实时形变方法研究

学位论文作者签名：徐兴华　　日期：2007 年 12月27日

学位论文版权使用授权书

本人完全了解国防科学技术大学有关保留、使用学位论文的规定。本人授权国防科学技术大学可以保留并向国家有关部门或机构送交论文的复印件和电子文档，允许论文被查阅和借阅；可以将学位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、汇编学位论文。

（保密学位论文在解密后适用本授权书。）

学位论文题目：虚拟膝关节手术中的大范围实时形变方法研究

学位论文作者签名：徐兴华　　日期：2007 年 12月27日

作者指导教师签名：熊宗山　　日期：2007 年 12 月 27日

第一章 绪言

1.1 研究背景和意义

1.1.1 虚拟手术概述

虚拟手术又被称为手术仿真，它运用计算机技术（主要是计算机图形学与虚拟现实）将医学手术所包含的各个方面和过程尽可能地在计算机等设备上虚拟出来，来模拟和指导医生进行真实的医学手术。它所涉及的内容包括对医学数据进行采集、建模、可视化和操作，以及对虚拟人体器官在虚拟手术器械作用下的各种变化的模拟和对操作人员的各种感官反馈（如视觉和触觉渲染等）的模拟。虚拟手术涉及到诸多研究领域的知识，如医学、生物力学、机械学、材料学、计算机图形学、计算机视觉、数学分析、机械力学、材料学、机器人学等等。

虚拟手术让医务工作者沉浸于虚拟的手术环境中，体验与学习如何应付各种临床手术的实际问题。操作者可以通过视觉、触觉甚至听觉感知学习各种手术实际操作，并能够预演手术的整个过程。用它做科学分析时也可以用来验证一个假设的可行性。这样大大节约了培训医务人员的费用和时间，降低了非熟练人员实习手术的风险性，还可以利用专家的手术经验和手术实例对年轻医生进行培训，这对促进医学水平的提高有着重大意义。

虚拟手术系统的主要构成如图 1.1 所示，其中的关键技术包括人体器官模型的实时绘制与特效处理、形变建模和切割模拟、碰撞检测、特效处理以及触觉反馈实现等等，还可能包括对人体正常和病变组织的生理建模和病理建模。

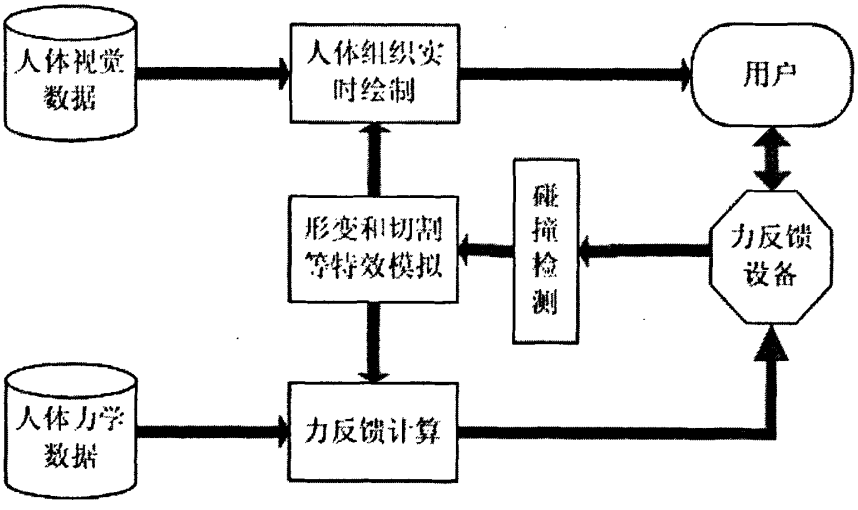

图 1.1 虚拟手术系统主要构成示意图

目前比较常见的虚拟手术系统按照通用性可以分为两类：第一种是使用通用的三维交互设备，可以应用于一般手术的，如美国 Mayo 医院附属的研究所设计的 VARSP[1]。第二种是针对具体手术设计的，比如 Juli Yamashita[2]设计的针对鼻内腔内窥手术的模拟导航系统，法国 INRIA[3]开发的针对肝脏的虚拟手术系统等，德国 Forschungszentrum Karlsruhe[4]开发的针对子宫和卵巢病变的虚拟腹腔镜手术系统，第一种方案通用性比较好，但它的真实感不够强。第二种方案是针对具体手术设计的，通用性不好，但它的真实感很强。这两种交互方案各有优劣，在实际的使用中应该针对具体应用做出具体选择。

本文中的方法就是针对于具体手术的，它的形变效果将应用于虚拟膝关节前交叉韧带重建手术。

1.1.2 膝关节前交叉韧带重建手术

膝关节前交叉韧带（ACL）的损伤是一种常见的膝关节疾患，尤以运动员高发，它严重影响患者特别是运动员的生理机能。前交叉韧带是膝关节最重要的静力稳定结构，一旦断裂可引起膝关节不同程度的不稳而影响运动功能，若治疗不当可发展为 ACL 缺失，从而导致膝关节进一步损害。可是膝关节 ACL 重建手术因为其复杂性，危险性，长期以来一直是运动医学界及骨科界极为重视而又未能很好解决的重要课题。

手术流程如下：

(1) 硬膜外麻醉；

(2) 平卧于手术台上，患侧肢体下垂；麻醉状态下做“抽屉试验”，确定 ACL 有无损伤；

(3) 手术入路：于两侧“象眼”（在膝关节间隙水平，髌韧带两侧）做约 0.5cm 长切口，左手侧进入关节镜观察，右手侧进入探钩、咬钳、刨削刀、磨钻、射频汽化刀等手术器械。

(4) 关节镜检查：清理手术视野进一步确认 ACL 有无损伤及损伤程度，确定实施 ACL 紧缩术还是 ACL 重建术。同时应注意是否伴发半月板、软骨的损伤。

(5) 肌腱切取：确定行 ACL 重建术后，在膝关节内下方做斜行切口，分离找到骨薄肌、半腱肌在胫骨上的止点并切断，将肌腱的一端穿入肌腱剥离器，左手拉紧肌腱末端，右手缓缓推入肌腱剥离器，在肌腱与肌腹交界处切断，取出 2 根肌腱。

(6) 肌腱处理：在全部处理过程中须保持肌腱湿润。肌腱的两端分别用不可吸收缝线编织缝合；在牵张器上“预张力”，防止术后移植肌腱发生松弛；肌腱中部打结并在肌腱结内嵌入骨棒，生理盐水浸泡备用。

(7) 髁间窝成形：清除损伤的 ACL、增生的滑膜，修整损伤的半月板及软骨。

(8) 定位：确定股骨及胫骨骨隧道的进针点。（a）ACL 的原来止点（髁间嵴前方）即为胫骨隧道的定位点。（b）外侧股骨髁的内侧面（髁间窝的外侧壁）前后方向分为 4 等分，后 1/4 交界处即为股骨隧道的进针点。

(9) 隧道制作：（a）先做胫骨隧道。胫骨隧道定位器的尖端经关节镜入路置于胫骨定位点，定位器的另一端置于胫骨上端内侧，插入导针，环钻沿导针方向钻入关节腔，将环钻内的骨头取出备用。（b）股骨定位器置于髁间窝外侧壁后 1/4 交界处，定位器内插入导针，用电钻将导针旋入，导针另一端从股骨下端外侧旋出，再用阶梯钻由外向关节内钻入，形成阶梯状隧道，最后用阶梯状锤骨棒夯实隧道。（c）在胫骨隧道外口的远侧约 1cm 处钻另一孔，此相邻二孔在骨髓腔内相通，备固定肌腱用。

(10) 穿入肌腱：将处理完毕的移植肌腱从股骨下端外侧的股骨隧道入口导入，进入关节腔，用钳子把移植肌腱的缝合线夹住，经由胫骨隧道拉出体外。

(11) 固定：右手拉紧肌腱缝线，左手屈伸膝关节 20 次，确保肌腱结及肌腱结内的骨棒与股骨阶梯状隧道内壁的远端紧密接触。肌腱远端分为两束，分别从胫骨上端的两孔内引出，打结，并用缝线将肌腱与周围软组织缝合，加强固定。最后把步骤 9 中取出的骨头塞入胫骨隧道内，使肌腱与隧道壁紧密镶嵌。

(12) 缝合切口，结束手术。

1.1.3 虚拟膝关节镜手术中人体模型的形变

在膝关节镜手术实施的过程中，患者的膝部在打孔的时候应该是向上屈起的；并且在新肌腱已植入并固定之后，要将患者的膝部大幅度地伸直与屈起多次。然而系统中实际采用的模型仅仅是一个已经弯曲到某个程度的静态模型，无法提供上述过程中的膝关节弯曲真实效果，并且在打孔操作中，如果膝关节弯曲程度不够会造成孔的位置的错误，导致打孔失败。因此，我们需要找到一种合适的膝关节模型的形变方法，这种形变方法必须具有足够的实时性，同时形变效果也要满足正确性，逼真性。

目前对于人体模型的形变大致有以下几种类型：

(1) 刚性形变。

对于人体表面模型的形变一开始采用的是刚性的形变方法，这种方法出现得很早，其中人体由彼此独立的皮肤片断组成，这些皮肤片断相对骨架静止，随着骨架一起运动。这种方法效率非常高，但效果很不逼真。目前这种方法仅应用在要求对系统资源占用极少，并且对形变效果要求比较低的场合。

(2) 利用局部形变算子。

Thalmann 等人[5]首先引入了依赖于关节的局部形变算子的概念到形变中，这种方法当时被用于控制手掌结构皮肤表面的形变，将连接处的点加权地连续映射到相关的各段骨架上，这种方法具有较快的速度，结果具有连续性，但灵活度低，形变效果一般。

(3) 骨架驱动形变。

骨架驱动形变（skeleton driven deformation, SDD）是一种经典的皮肤形变方法，又叫做骨架子空间形变（sub-space deformation, SSD），这是一种基于骨架的加权插值的方法[7][8][9]。其中的点的新位置可以表示为各关节变换的加权组合。如式（1.1）所示：

$$P_V = \sum_i \omega_i (M_i \times M_{i,Dress}^{-1} \times P_{Dress}) \tag{1.1}$$

其中，ω_i 是权值，$\sum_i \omega_i = 1$，对应于各关节对该点的影响程度，M_i 和 $M_{i,Dress}^{-1}$ 分别是第 i 个影响关节在新姿态和初始姿态下的变换矩阵，P_{Dress} 是点 p 在初始姿态下的坐标。根据第 i 个关节的运动（M_i 和 $M_{i,Dress}^{-1}$），可以计算得到点 p 的新位置 p_v。

这种方法速度快，效果较前两者好，但也会产生失真的情况。

(4) 基于实例的插值形变。

Mohr[10]给出的基于实例的插值形变方法是一种可以实时形变的方法。首先预定义一组关键形状，然后通过在关键形状间插值获得在其间演化形变的效果。这种方法适用于对于有同一个实体的多个时刻运动采样效果的模型的形变，并且具有很好的逼真度和实时性。不过该方法无法对单一姿态采样的模型进行形变。

我们主要研究基于单一姿态采样的人体表面网格模型的大范围形变技术，而且我们要对表面网格模型进行形变，因此我们的工作可以参考骨架驱动形变的这种模式，并在此基础上提出我们的基于骨架的表面网格模型形变方法。

1.2 本文的研究工作

针对虚拟手术系统，特别是课题中开发的“虚拟膝关节镜手术系统”中膝关节的形变特点，并且着重考虑系统实时性和视觉逼真性方面的要求，我们针对膝关节模型，首先提出了一个骨架获取方法：基于模型横截面的骨架获取方法，做法是将这组截面一系列中心点拟合为 3 维空间上两条相交的有向线段，作为模型的骨架。这个过程得到的骨架信息可以供后续的算法使用。

基于给定的骨架，可以给定骨架与模型的映射关系，通过改变骨架的位置信息对模型进行形变。在这个环节中，本文提出了两种实时形变方法：

（1） 利用柱坐标平滑变换的方法对膝关节表面模型进行形变。这种形变方法对顶点的柱坐标按照角度的连续性进行了平滑的变换，从而能够较好地保持模型的局部细节特征，并且能够保证模型不会发生体积和整体形状的失真，从而达到令人满意的视觉效果。这种方法的另一个优点是所有计算都是线性的，所以它同时具有很好的实时性。

（2） 本文利用改进的微分坐标形变方法对膝关节的表面模型进行形变。本文针对原微分坐标形变方法存在的不足做出了两个改进，即添加了表面材质约束和体积约束，克服了微分坐标形变方法的出现大范围形状的失真的问题。由于两种改进策略并没有增加太多计算的复杂度，也能使算法保持了它本身的实时性的优点，从而同时达到令人满意的形变效果与效率。

本文还介绍了算法在系统中的实现，并且通过实现结果表明了本文中所有方法的实时性和形变效果的逼真性。

1.3 本文的组织结构

本文分为 6 章，第 1 章简要介绍虚拟膝关节手术以及它与模型形变关系，并且概括了本文的研究工作。第 2 章对本文工作的基础——表面网格模型形变技术进行了说明，并且介绍了与之相关的一些研究工作。第 3 章介绍了基于模型横截面的骨架获取方法。第 4 章和第 5 章分别具体介绍了我们提出的两种形变方法，分别是柱坐标平滑变化的形变方法和改进的微分坐标形变方法。第 6 章大范围形变在虚拟膝关节手术中的实现以及其形变效果与性能测试。第 7 章在总结本文工作的基础上提出了对未来研究工作的展望。

第二章 表面网格模型形变技术概述

表面网格模型是一种常见的三维模型的表示形式，它由于数据量较体模型而言相对小，对物体表面的表现能力相对较强，而被广泛地运用在空间三维模型尤其是它们的表面模型的表示中。本文研究的问题就是表面模型的形变。因此，本章首先介绍表面模型有关知识，然后简单介绍研究者们在相关领域的研究工作，最后对形变方法做一个总结。

2.1 表面网格模型简介

表面网格模型将模型表示为空间上的一系列顶点所以及它们的连接关系所组成的集合。一旦得到了这些点的坐标，以及它们的连接关系，就能重现这个模型的表面特征。图 2.1 就是一个表面网格模型。

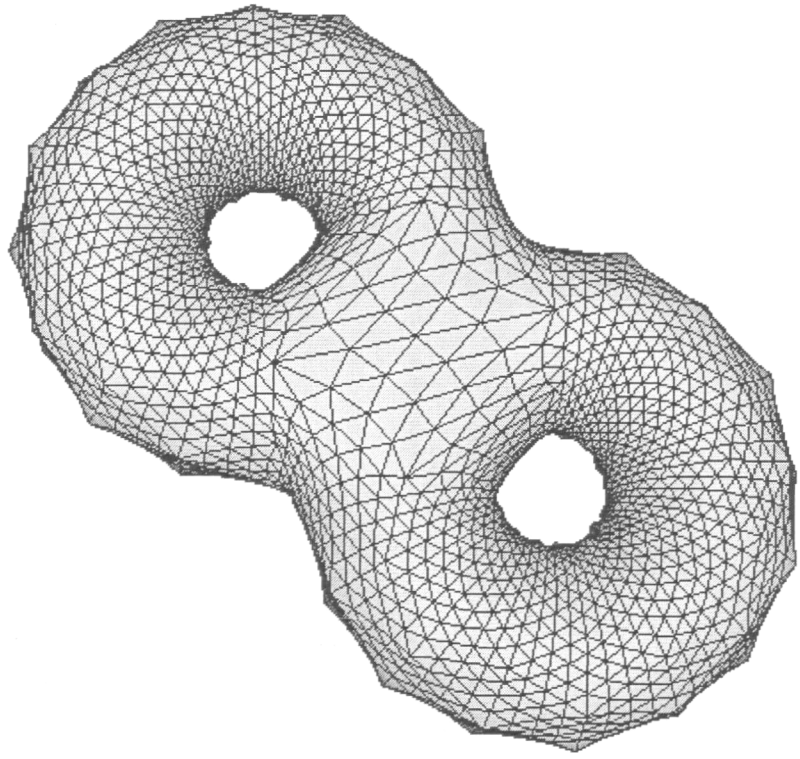

图 2.1 一个 3d 表面网格模型

由于表面网格模型的形状由各顶点位置和顶点之间的连接所确定，我们可以自然地想到，如果要使这个模型进行形变，可以通过更改其表面网格模型的顶点的坐标或者它们之间的连接关系来实现。在实际的操作中，往往只需改变模型中顶点的坐标，而不改变顶点之间的连接关系，就能达到所需的形变效果。本文所

考虑的情况主要是根据原模型的特点，按照一定的方法，得到每个顶点的新坐标，从而达到形变效果。

2.2 表面网格模型形变的评价标准

而网格形变的过程中，采用不同的形变算法，运算效率和得到的模型形变效果不尽相同。对于虚拟手术系统中的形变而言，我们关心的主要是形变的逼真度和形变的实时性，因此，很有必要找到一种有足够逼真度和实时性的方法。

2.2.1 形变的逼真度

形变的逼真度与人的视觉关系比较大，形变后的模型应该具有人的视觉所认同的正确性和真实感，尤其是虚拟现实中的物体，要具有它在现实世界里的原型的主要特点。虚拟现实中的形变，其效果要和已经存在于现实中的各种物理规律相一致。

要评价形变效果的逼真度，我们主要从以下几个方面去衡量：

(1) 形变后所保持的整体形状。

在虚拟现实系统中，形变后，应该保持整体的形状不发生大范围的失真。而对于表面网格模型的形变，容易出现对薄壳模型形变的效果，通常是体积的失真。如果形变物体是实心的体而不是薄壳模型，对其表面模型进行的形变有可能仅仅符合薄壳模型的形变规律，而不符合实心物体的形变规律（如图 2.2）。为解决这一个问题，形变的算法应该有一个使全局形状不失真的约束。

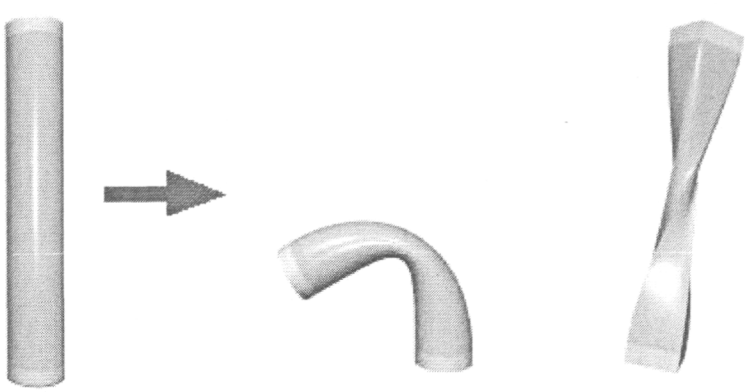

图 2.2 整体形状的失真（体积失真）[11]

(2) 形变后所保持的局部细节。

在虚拟现实系统中，形变后的物体应该具有其形变前的表面局部细节特征。物体表面的细节是我们的视觉能够辨认物体的重要依据，有时候我们能够很快地辨认形变后的物体，一定程度上是因为我们已经熟悉了形变前物体的表面细节特征。表面细节特征变化很不合理的时候我们往往会感觉到失真。从物理规律的角度，

当一个物体在大范围形变的时候，表面细节在很大程度上是跟随物体整体形状进行旋转和平移的，往往只有相对较小程度的扭曲，如果处理不当，会产生失真（如图 2.3）。为解决这一问题，我们可以把表面细节从模型中提取出来，根据它们得到出更加逼真的变形结果。

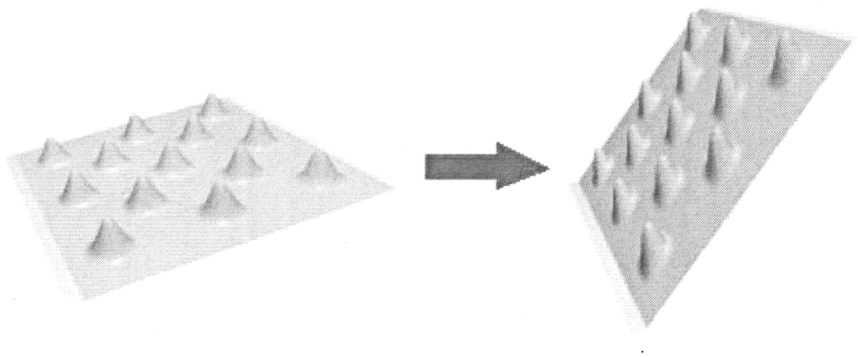

图 2.3 表面细节的失真[11]

(3) 形变后所满足的拓扑性。

在虚拟现实系统中，虚拟物体应该具有和原型物体一样的拓扑性。物体拓扑性的改变会很大程度上影响物体的逼真度。在形变中，如果处理不当，经常会出现模型拓扑性改变的情况，比如模型的自相交等等（如图 2.4）。为避免改变拓扑性，在形变的时候应该对物体的拓扑信息做相应的记录和处理。

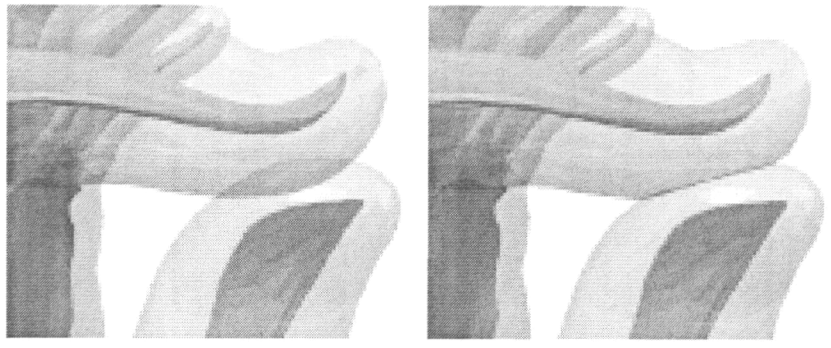

图 2.4 模型的自相交[12]

（左图是自相交的现象，右图是正确的效果）

为了能够很好地保证形变效果的逼真性，许多形变方法利用了表面网格点与点之间的联通特性，将网格中的边的信息和点的信息综合起来进行处理，得到形变的约束条件，在求解的时候加入这些约束条件就可以得到更加逼真的形变。

当然也有的形变方法没有利用网格中边的信息，仅仅通过考虑点的分布，处理每个点的位置信息，在求解的时候也可以通过点所在空间的不可重叠性作为约束，达到逼真的形变效果。

2.2.2 形变的实时性

形变的实时性是虚拟手术的重要需求之一，实时性是使用户能够达到足够的沉浸感，并且能与系统很好地进行互动的一个先决条件。人眼对连续的感觉阀值大约是 25 FPS（帧/秒），如果形变过程中的刷新率能够达到 25FPS 以上，就能产生较好的视觉上的实时性。

形变的实时性与形变算法的运算机理有关。从这个角度，可以将目前的算法分为线形和非线性两类，线形算法的运算效率较高，一般都有很好的实时性，可是有时形变效果却相对不够逼真[11]。非线性算法恰恰相反，其形变效果很好，但往往实时性不高。因此，虚拟手术中形变技术的研究中，我们更倾向于采用线性算法进行形变。

2.3 相关工作

下面主要介绍一些常见的用于表面模型形变的方法。

2.3.1 自由形式形变

自由形式形变（Freeform Deformation）在 1986 年由 Sederberg 和 Parry[13]提出，随后又很多人进行了改进[14][15][16]。这种方法基本思路是首先将模型的组成元素（一般是点）嵌入空间中的一系列占有一定体积的相互关联的格子中，根据每个点的笛卡尔坐标，确定该点所在的格子，以及在这个格子内部的相对坐标。

这些格子按照其顶点位置的改变而形变，格子中的每个点都有一个在格子内部的相对坐标(s,t,u)，当格子顶点的位置发生变化之后，格子内的部分也相应地发生形变，模型中的点就会随着这些格子的形变而移动，以达到形变效果。其过程可以用公式表示如下：

$$F(X)=U$$

$$\overline{F}(U)=X$$

$$F(x)=\overline{F}\ (F_{-1}(X)=\overline{F}(U)=\overline{X} \tag{2.1}$$

由此得到点的新坐标。

这种方法的形变效率较高，可是有可能会造成失真。

2.3.2 能量最小化模型

能量最小化（Energy minimization）由 Welch 和 Witkin[17]在 1992 年提出，之后的十多年来经常被用于对形变进行优化。他们对系统的张量和扁率的变化进行了综合考虑，即：

$$Q(w)=\iint_w \|G\|_\alpha^2+\|B\|_\beta^2 \tag{2.2}$$

通过求解变化达到最小值时的点的坐标，即求解一个能量最小化方程组：

$$\begin{vmatrix} H & A^T \\ A & 0 \end{vmatrix} \begin{vmatrix} p \\ y \end{vmatrix} = \begin{vmatrix} f \\ b \end{vmatrix} \tag{2.3}$$

其中，H 是计算形状特征的算子，A 是线性约束，b 是实际控制点的的位置，p 是方程组的解（向量）

直到 2004 年，Boier-Martin[18]对它做了改进，使它能够保护模型表明的细节在形变时不会发生失真。其做法是摒弃了以往的位移场的能量最小化，改为矢量场的能量最小化，这就带来了保细节的效果。它的能量最小化系统是：

$$\left|\begin{array}{c|c} \overline{\Delta}^k & \\ \hline 0 & I_{F+H} \end{array}\right| \begin{vmatrix} p \\ f \\ h \end{vmatrix} = \begin{vmatrix} 0 \\ f \\ h \end{vmatrix} \tag{2.4}$$

其中 $\overline{\Delta}^k$ 是局部细节的算子，p 是形变区域内的自由顶点，f 是形变区域外的固定顶点，h 是操纵手柄区域内的变化顶点。

这种方法对于保持局部细节是很有效的，其利用能量最小化模型来保持细节和添加与控制点的关联的这种思路，已经广泛地被别的算法吸收和采纳。可是有可能会产生较大范围上整体形状的失真[19]。

2.3.3 基于梯度域的网格形变

基于梯度域的网格形变有多种形式，以 Alexa[20]在 2003 年提出的方法，Yu[21]在 2004 年提出的方法，Sorkine[22]在 2004 年提出的方法，Lipman[23][24]在 2005 和 2006 年提出的方法等等为代表。其基本思路是将形变考虑为一个表面微分坐标的形变能量最小化的过程 ，能量函数和位置约束综合在一起计算，将形变引起的模型的偏移和旋转相对均匀地平滑地分布到整个网格中每个点的局部位置。

其中对表面微分坐标的计算通过微分算子 L_s 计算：

$$(L_s)_{ij}=\begin{cases} d_i & i=j \\ -1 & (i,j)\in E \\ 0 & \text{其它情况} \end{cases}, \tag{2.5}$$

其中 d_i 是每个顶点的邻接点的个数，E 是模型的边集。

其能量最小化的过程是

$$\min_x\left(\|Lx-\delta(x)\|^2+\sum_{j\in C}\varpi^2\left|x_j-c_j\right|^2\right) \tag{2.6}$$

最终转化为解方程组

$$\begin{pmatrix} L \\ \varpi \cdot I_{m\times m} \mid 0 \end{pmatrix} x = \begin{pmatrix} \delta(x) \\ \varpi \cdot c_{1:m} \end{pmatrix} \tag{2.7}$$

为了提高计算效率，可以把解方程的计算量分解，分为承担大部分计算复杂度的预处理部分，和反复循环计算的部分。

这种形变方法所提供的用户与模型之间交互的自由度很大，但是这种方法维持了一部分形变前的全局坐标系下的梯度，因此有时候形变效果不理想[19]。

2.3.4 多控制的网格编辑

基于分级 B-样条编辑技术（hierarchical B-Spline editing），多控制的网格编辑（multiresolution mesh editing techniques）在 1997 年被 Zorin[25]首先提出，随后还有许多人提出类似的方法[26][27]。它的基本思想是通过网格的细分将网格划分为一系列的频率波段，首先通过控制低频部分的信息生成一个粗略的形变后的平滑网格，然后再向其上添加高频部分的信息生成表面局部细节。

2007 年，Kun Zhou[19]提出了一个直接操纵的细分表面模型，并利用 GPU 加速运算，得到了很高形变的效率。它的基本思想是，通过梯度域方法将操纵点的信息与整个模型结合起来，得到能量最小化模型：

$$\min_{V_d}\left(\left\|Lf(V_c)-\hat{\delta}\left(f(V_c)\right)\right\|^2+\left\|Cf(V_c)-U\right\|^2\right) \tag{2.8}$$

L 是微分算子，V_c 是细分前的平滑网格，$f(V_c)$ 是细分后的网格，$\hat{\delta}\left(f(V_c)\right)$ 是微分坐标，V_d 是细分后的网格，C 是控制点，U 是控制点的当前新位置。

Kun Zhou 提出了一种壳模型算法以减小形变过程中保持细节而产生的非线性所带来的开销，并且保证了生成结果的迭代过程的收敛性。

Marinov[28]同年提出细分表面模型，将一个双波段的多控制的形变框架应用在 GPU 上，也有很高的效率。

多控制的网格编辑方法形变效果比较好，而且在 GPU 上运行时形变效率比较高，可是由于它有一个网格细分的过程，这种细分有可能会造成最后的形变效果失真。

2.3.5 线性旋转不变方法

Lipman[23]在 2005 年引入了一种刚性的运动不变的网格表示法，这种方法基于网格中离散的方向信息和距离信息的片段，进行网格形状的重构，重构的网格形状可以通过严格的变换而唯一确定。在局部结构和顶点上加上用户定义的约束，

以此来定义表面的编辑操作。这些约束被合并起来，形成两个线性重构系统，对它们依次进行求解可以得到形变后的形状，这样得到的形状可以通过使用最小二乘法尽量保持局部细节。

这两个系统是由分段的网格得到的，它们包含了网格形变所需要的全部形状信息：第一个系统定义了局部片断之间的联系，用下面两个公式所代表的方程组进行计算：

$$\left\langle \hat{x}_m^{n_k^i}, \hat{N}^{n_k^i} \right\rangle = \left\langle \tilde{x}_{k+1}^i - \tilde{x}_k^i, N^{n_k^i} \right\rangle + \left(\tilde{L}_{k+1}^i - \tilde{L}_k^i\right)\left\langle N^i, N^{n_k^i} \right\rangle$$

$$\left\langle \hat{x}_{m+1}^{n_k^i}, \hat{N}^{n_k^i} \right\rangle = -\left\langle \tilde{x}_k^i, N^{n_k^i} \right\rangle + \tilde{L}_k^i \left\langle N^i, N^{n_k^i} \right\rangle \quad (2.9)$$

其中，n_k^i代表全局排序的第i个顶点的一环邻域内的局部排序第k个顶点的全局序号，$\hat{x}_k^i$是全局排序的第i个顶点指向它的一环邻域内局部排序第k个顶点的矢量，$\tilde{x}_k^i$是$\hat{x}_k^i$在第i个顶点所决定的局部标系下的坐标表示，N^i和$N^{n_k^i}$分别是点i处和点n_k^i处的单位法向量，$\tilde{L}_k^i$是第i个顶点的一环邻域内第k个点到第i个顶点切平面的距离。

第一个系统描述了点与点之间的方向关系。

第二个系统记录了片段内部各顶点的局部坐标。通过解如下的方程组可以计算出最终结果：

$$\hat{x}^j - \hat{x}^i = \hat{x}_k^i + \tilde{L}_k^i N^i = \left\langle \hat{x}_k^i, b_1^i \right\rangle b_1^i + \left\langle \hat{x}_k^i, b_2^i \right\rangle b_2^i + \tilde{L}_k N^i \quad \forall (i,j) \in E \quad (2.10)$$

第二个系统描述了点与点之间的距离关系。

两个系统相结合，点与点之间的方向和距离的关系相结合，就能表现出整个网格的结构。这种方法对（这些局部）形状进行的线性结合，使线性形状修改成为可能，当然这种修改要正确地进行旋转操纵。

这种方法对于保持模型中旋转部分的细节能够获得很好的效果。可是这种形变在模型旋转较小但位移较大的情况下会产生失真[19]。

2.3.6 各种非线性方法

前面介绍的五种方法都是线性的方法，当然人们还提出了许多非线性方法，比如 Botsch 错误！未找到引用源。[31]在 2006 年发表的的关联棱柱的方法和在 2007 年提出的基于刚性单元的空间形变，Oscar Kin-Chung Au[32]在 2007 年提出的能识别柄的等值线方法，Sumner[33]在 2007 年提出的嵌入形状的形变，等等。

这些方法虽然思路和过程大相径庭，可是它们都利用了比较复杂的表示模型，并进行了大量的图形优化工作，因此它们的形变效果很好。不过，大量的非线性

运算也导致了它们的运行效率普遍较低，而且它们的实现复杂度较高。

2.4 小结

为了使形变算法能够更逼真地显示和具有更快的速度，人们已经进行了大量的工作，提出了许多种算法。当前热点主要集中在形变的可测量性、可扩展性、几何表示和复杂度的无关性以及精简模型的使用等方面。目前所研究的算法中有的能够很好地保持逼真的效果，可是往往由于计算开销大而无法保证其实时性；而实时性好的算法，也牺牲了一些逼真度。实时性和逼真度都是衡量一个算法成功与否的标准。一种成功的算法应该是能够针对当前的问题特点而提出的，能够充分利用好现有的技术，并能圆满解决相关问题的方法。同时它也应该是在形变的逼真度与实时性之间的一个恰当的折衷。

第三章 基于模型横截面的骨架获取方法

由于在现实中，膝关节的表面皮肤是随着骨架的运动而运动的，因此我们的形变方法都是以骨架为基础的。所以我们在对模型的形变之前，首先要得到模型的骨架。而膝关节模型的横截面是已知的，我们可以通过模型的横截面构建模型的骨架。本章我们将对这种方法进行介绍。

3.1 算法思想

膝关节模型的截面各点的坐标已知，我们可以将这组截面中心的轨迹拟合为两条相交的有向线段，作为模型的骨架轴线。这里要做的工作是找到各截面中心点的合理划分和根据这种划分拟合出的骨架轴线，如图 3.1。骨架轴线拟合的结果由式（1）表示：

$$\vec{a}=\frac{1}{m-1}\sum_{i=1}^{m-1}\vec{a}_i\ ,\quad \vec{b}=\frac{1}{n-m-1}\sum_{i=m}^{n-2}\vec{b}_i \tag{3.1}$$

其中 $\vec{a}$ 和 $\vec{b}$ 是两个与骨架平行的矢量，n 是点的个数，通过划分，得到两个点集 $A=\{P_i|i\in[0,m-1]\}$，$B=\{P_i|i\in[m,n-1]\}$ 分别拟合两个矢量，$\vec{a}_i$ 和 $\vec{b}_i$ 是点集 A 和 B 拟合过程中用到的局部分量。

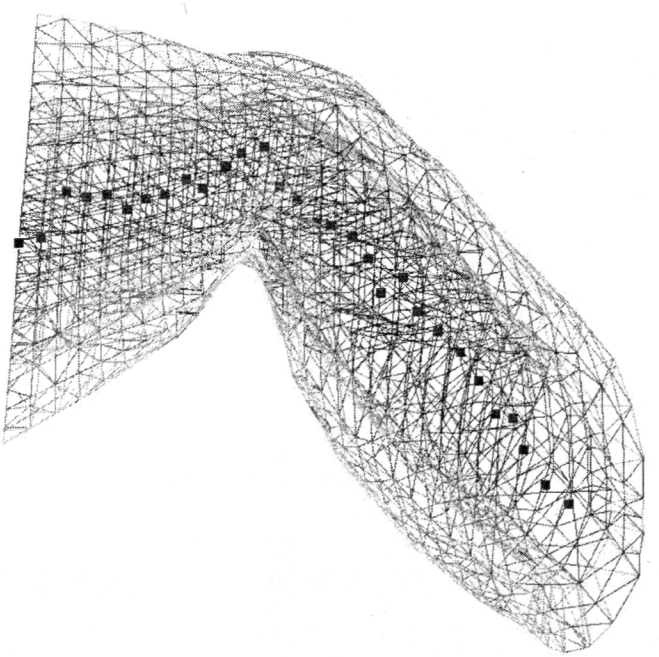

图 3.1 双层的膝关节 3D 网格模型
其中黑色点为横截面的中心点

当骨架方向确定之后，以过点P_1平行于$\vec{a}$的直线不一定与过点P_n平行于$\vec{b}$的直线相交，即有可能骨架在连接点处出现不连续的情况。我们通过求过两条直线的最短线段并取其中点O，作为最终拟合得到的骨架连接处，最后获得的骨架就由线段P_1O，OP_n组成。

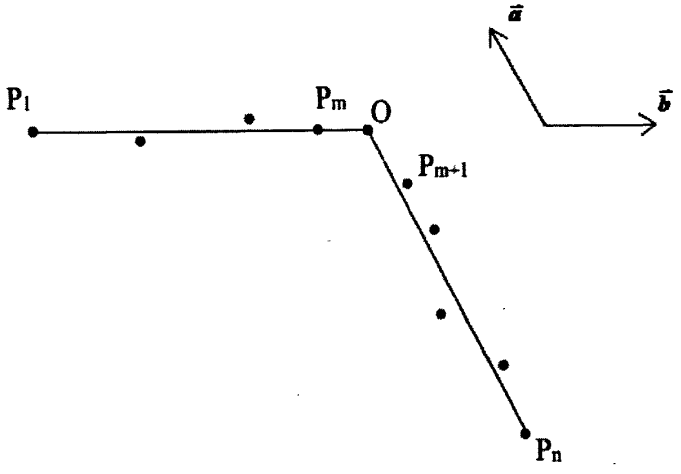

图 3.2 获取轴线的示意图，
其中P_1到P_n是横截面的中心点

3.2 算法过程

如图 3.2，首先将所有横截面中心点沿着从大腿到小腿的方向依次排序，分别命名为P_i，其中$i\in[0,n-1]$，n为中心点的总数。设两轴交点为O。设$\vec{a}=\frac{\overrightarrow{P_0O}}{|P_0O|}$是要拟合出来的上半部分的轴的单位向量，单位向量$\vec{a}_0$是不断逼近$\vec{a}$的中间结果；$\vec{b}=\frac{\overrightarrow{P_{n-1}O}}{|P_{n-1}O|}$是要拟合出来的下半部分的轴的单位向量，单位向量$\vec{b}_0$是不断逼近$\vec{b}$的中间结果。单位向量$\vec{a}_i=\frac{\overrightarrow{P_0P_i}}{|P_0P_i|}$；单位向量$\vec{b}_i=\frac{\overrightarrow{P_{n-1}P_i}}{|P_{n-1}P_i|}$。将所有中心点划分到$A$、$B$两个点集中，使$A=\{P_i|i\in[0,m-1]\}$，$B=\{P_i|i\in[m,n-1]\}$，可见要求出$A$、$B$，就要先确定$m$。

两条轴线的构建过程如下：

① 将$\vec{a}_1$赋给$\vec{a}_0$，将$\vec{b}_{n-2}$赋给$\vec{b}_0$，A ={ P_0 ， $P1$}，B={ P_{n-2} ， P_{n-1} }。

② 将 i 依次取 2 至 n-3，对于每个 i，执行步骤 3。

③ 若$\vec{a}_i\vec{a}_0 > \vec{b}_i\vec{b}_0$则使$A = A \cup \{P_i\}$，执行步骤④；若$\vec{a}_i\vec{a}_0 < \vec{b}_i\vec{b}_0$则使$B = B \cup \{P_i\}$，执行步骤⑤。

④ 修改$\vec{a}_0$的值为$\frac{1}{i}\sum_{j=1}^{i}\vec{a}_j$，返回步骤②。

⑤ 令两个部分的分界值$m = i$。

⑥ 得到拟合的中间结果$\vec{a}_0 = \frac{1}{m-1}\sum_{i=1}^{m-1}\vec{a}_i$，$\vec{b}_0 = \frac{1}{n-m-1}\sum_{i=m}^{n-2}\vec{b}_i$。

⑦ 上半轴的长度$l_1 = \left|\overrightarrow{P_nP_1} \times \vec{b}_0\right|$，下半轴的长度$l_2 = \left|\overrightarrow{P_nP_1} \times \vec{a}_0\right|$

⑧ 两轴交点位置$O = \frac{(P_1 + \vec{a}_0 * l_1) * l_1 + (P_n + \vec{b}_0 * l_2) * l_2}{l_1 + l_2}$

第 8 步意为求出与两个轴都相交的公垂线，以两轴上的骨架长度为权重，在两交点之间的线段处取一点作为轴心O。最后结果所得轴线$\vec{a}$起点是P_0，终点是O；$\vec{b}$起点是O，终点是P_n。

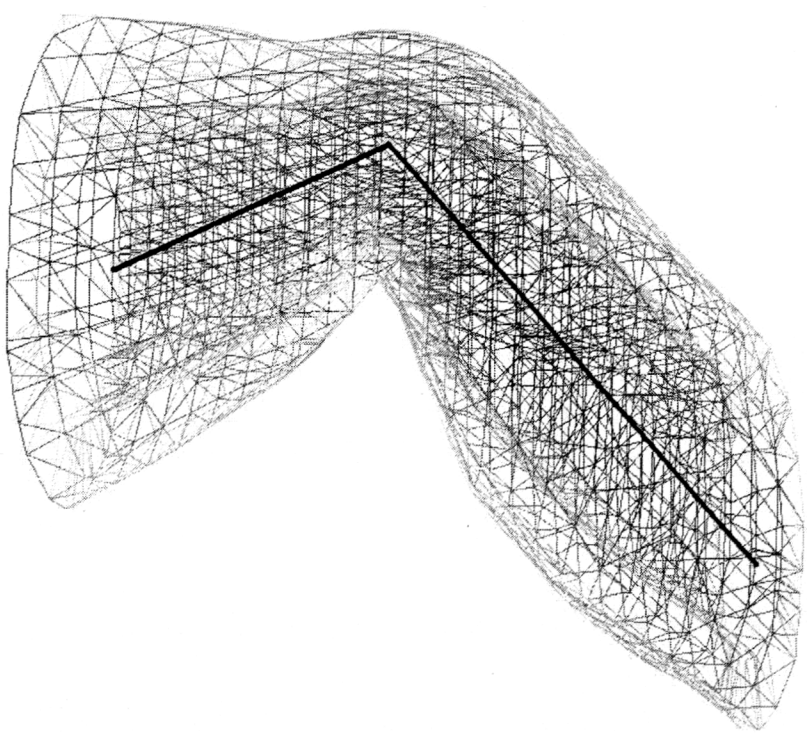

图 3.3 膝关节双层 3D 模型的轴线示意图

3.2 算法评价

由于这个过程属于预处理环节，仅在形变之前执行一次，并且算法的复杂度

为 max(O(m*n)，$O(n^2)$)（m 是每个横截面点的个数，n 是横截面的个数），而实际问题中每个横截面的点的数量，横截面的个数一般都不会很大，所以这个算法的计算开销基本可以忽略。

这个方法能够为后续的形变正确地找到网格的骨架模型，并且有很高的效率，较容易通过编程实现。计算的结果如图 3.3 所示。

第四章 柱坐标平滑变化的形变方法

前面介绍了我们获取骨架的方法，接下来要根据骨架对 3D 表面网格进行形变，为此我们提出了两个方法用来解决这个形变问题。本章要介绍的是第一个方法——柱坐标平滑变化形变方法。

4.1 算法思想

膝关节模型的弯曲可以近似地看作小腿部分与大腿部分的夹角变化的过程，这个过程在骨架上的反映是骨架的夹角的变化，而在表面模型上的反映是一系列顶点的运动，这些顶点运动的趋势是围绕膝关节旋转轴线的旋转。当然，这种旋转如果处理不好的话会带来很多问题，比如细节的丢失，整体形状的非正常扭曲，甚至还会出现对拓扑结构的破坏。为防止这些失真的现象的发生，我们把这个问题放到柱坐标系下考虑，并将每个点的旋转自然地对应于柱坐标角度分量的连续变化。

设 P 为模型上任意一点，设点 P 的柱坐标是(r,ρ,y)，由于旋转角所在的平面与在骨架所在平面是平行的，所以在形变过程中，我们把每个点的运动近似地看作是仅仅绕旋转轴的旋转运动。因此可以在柱坐标系下，仅仅对每个点的角度分量ρ进行处理即可。

距离骨架交点的那些点由于受到附近骨架的牵连较多，受远处骨架的牵连较少，因此可以近似地认为是同附近骨架的运动是相一致的，可以简单地看作是连同骨架一起按给定的角度旋转。于是，我们可以根据骨架把模型分为两部分：关节附近的部分与远离关节的部分，而远离关节的部分又可以依附近骨架的不同而又分成多个部分。

每个点的角度分量的变化可以用$\rho=\rho_0+\Delta\rho$来计算，为了使相邻顶点的角度分量有一个一致的均匀的变化，相邻顶点处的$\Delta\rho$必须是平滑的，接下来的工作就是要保证$\Delta\rho$的平滑变化。

对于处于关节附近的每个点，构造过这个点的垂直于旋转轴的模型的截面，如图 4.1 所示，当点落在膝关节外侧的时候，θ_0是膝关节外侧的那个大于 180 度的角；当点在膝关节内侧的时候，θ_0使膝关节内侧那个小于 180 度的角。ρ的变化是按照在其对应夹角θ在θ_0中的比例来变化，当θ/θ_0=1 的时候变化量就是骨架夹角的变化$\Delta\varpi$，于是由公式（4.1）能得到一个$\Delta\rho$的平滑变化。

$$\rho'=\rho+\Delta\rho=\rho+\Delta\varpi*\theta/\theta_0 \tag{4.1}$$

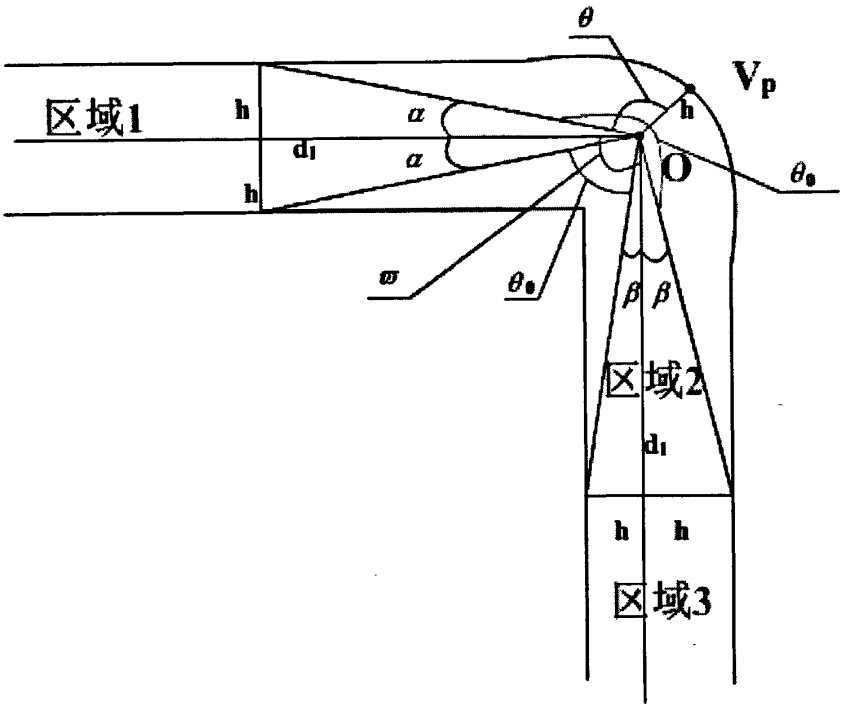

图 4.1 形变参数示意图

4.2 算法过程

设$\vec{a}$与$\vec{b}$的夹角为ϖ，设过交点的两轴的公共法线为$\vec{\eta}$，形变后夹角变为ϖ'，即$\vec{b}$绕$\vec{\eta}$旋转的角度为$\Delta\varpi=\varpi'-\varpi$。区域 1 中的点保持不动；区域 2 中的点旋转的角度与它们的位置有关；区域 3 中的点相对$\vec{b}$静止，即它们跟随$\vec{b}$绕$\vec{\eta}$旋转$\Delta\varpi$角度。以下是处理的步骤：

第 1 步，坐标变换。在笛卡尔坐标系下，通过坐标平移和旋转变换，两轴交点为O'，上半轴变为$\vec{a}'$，$\vec{a}'$与 x 轴重合，下半轴变为$\vec{b}'$，过交点的两轴的公共法线$\vec{\eta}$与 y 轴重合。这样，形变就可简化为$\vec{a}'$与 x 轴重合不动，$\vec{b}'$在XOZ平面内绕坐标原点旋转的问题。以O'为原点，以$\vec{a}'$为极轴，求每个点的坐标(x,y,z)对应的柱坐标(r,ρ,y)。

第 2 步，区域划分。根据点在轴上的投影到轴心O'距离的远近，把网格划分为三个区域，如图 4.2。区域 1 位于上半部分，随着上半轴作简单旋转；区域 3 位于下半部分，随着下半轴作简单旋转；区域 2 位于区域 1、3 之间，随着两轴的旋转角度的变化而形变。区域 1、3 的每个点在上半轴上的投影到两轴交点的距离大于某一临界值 d_1、d_3，要注意的是 d_1、d_3 的选取应该大于一定的值，以避免自相

交的发生。实际的划分结果如图 4.3。

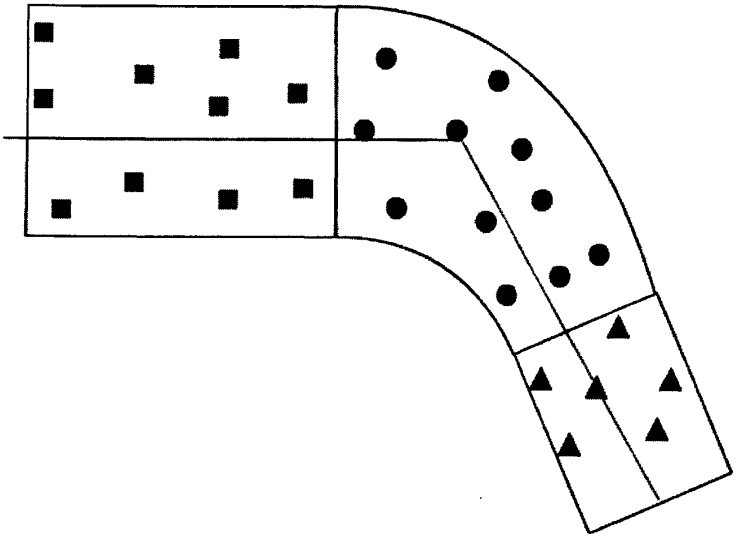

图 4.2 膝关节双层 3D 模型的划分示意图

四边形的点在区域 1 中，为上半段相对固定的点；

圆形的点在区域 2 中，为形变情况与位置有关的点；

三角形的点在区域 3 中，为与下半段的相对静止的点。

第 3 步，对于不同区域的点，求出其绕 y 轴旋转的角度，对它们进行的坐标旋转变换。区域 1 的点旋转角度为 0；区域 3 中的点旋转角度 $\Delta\varpi$；对于区域 2 中的每个点 V，其绕 y 轴旋转的角度因它的位置而不同，设 V 形变后为 $V'(r',\rho',y')$，具体步骤如下：

① 求 V 在 XOZ 平面上的投影 $V_p\ (r,\rho,0)$；

② 求 V_p 到两条轴线段的距离 h；

③ 如图 4.2 在 XOZ 平面内，求高为 h 的区域 2 在上半部分边界上的点 E_1，E_1 和 O' 的连线与 $\vec{a}'$ 的夹角 $\alpha=\arctan(h/d_1)$；求高为 h 的区域 2 在下半部分边界上的点 E_1，E_2 和 O' 的连线与 $\vec{\beta}'$ 的夹角 $\beta=\arctan(h/d_2)$；

④ 设 $\angle V_pO'E_1$ 为 θ，当 $|P_1O|+|P_nO|>|P_1V_p|+|P_nV_p|$，即 V_p 处于关节内侧时，$\theta=\rho-\pi-\alpha$。当 $\theta>0$ 时可保证形变结果的正确性。由图 4.1 知 $\rho>\pi+\alpha$，由此可保证 $\theta>0$；当 $|P_1O|+|P_nO|<|P_1V_p|+|P_nV_p|$，即 V_p 处于关节外侧时，有

$$\theta=\begin{cases}\pi-\rho-\alpha(0<\rho<\pi)\\3\pi-\rho-\alpha(2\pi<\rho<3\pi)\end{cases}\tag{4.2}$$

因此可保证 $\theta>0$，这能够保证不会发生网格的重叠或者翻转的情况。

⑤ 求得 $\angle E_1O'E_2$ 的值，当 $|P_1O|+|P_nO|>|P_1V_p|+|P_nV_p|$，即 V_p 处于关节内侧时，

$\theta_0 = \varpi - \alpha - \beta$，由图 4.2 知 $\theta_0 > 0$；当 $|P_1O| + |P_nO| < |P_1V_p| + |P_nV_p|$，即 V_p 处于关节外侧时，$\theta_0 = 2\pi - \varpi - \alpha - \beta$，由图 4.1 知 $\theta_0 > 0$。

⑥ 在过该点的平行于面 XOZ 的平面内，点 V 绕 y 轴旋转的角度 $\Delta\rho$ 为 $\Delta\varpi * \theta/\theta_0$，得到变换后的柱坐标的分量 $\rho' = \rho + \Delta\varpi * \theta/\theta_0$。

⑦ 根据变换后 V' 的柱坐标 (r', ρ', y')，求得它的直角坐标。

值得注意的时，形变后每个点的柱坐标分量 y 不变。

第 4 步，通过坐标的逆变换，将网格恢复到初始的坐标系下。

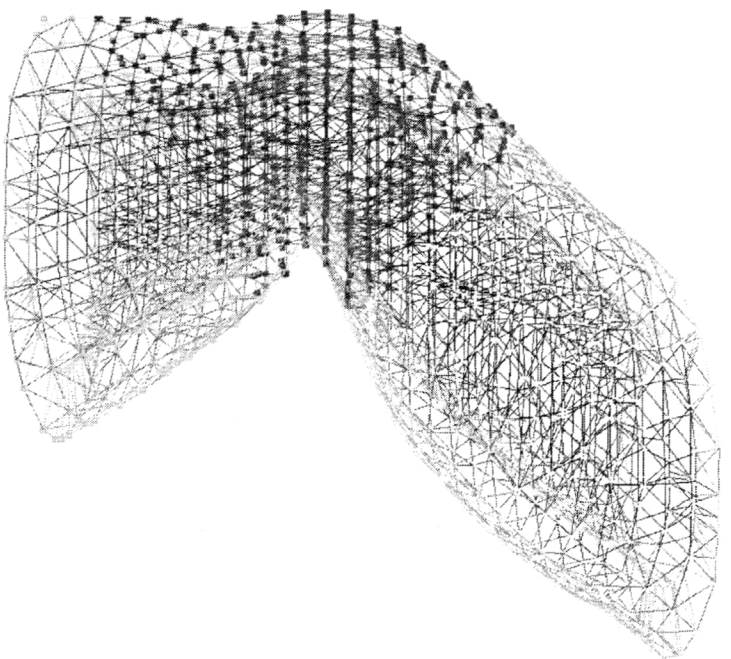

图 4.3 膝关节双层 3D 模型的划分

中间加粗的点代表区域 2，左右两边分别是区域 1 和 3 的相对固定的点

4.3 算法评价与测试

首先在模型上将各区域划分开来，划分结果如图 4.3 所示。接下来的形变结果如图 4.4 所示。通过对膝关节双层模型的形变结果的观察和分析，我们可以知道该算法有如下特点：

保持局部细节。这个算法在以旋转轴为原点的柱坐标系下，将每个点的角度分量进行了平滑的变换，因此相邻顶点 P_i 和 P_{i+1} 旋转的角度 $\Delta\rho_i$ 和 $\Delta\rho_{i+1}$ 差别不大，而柱坐标的其他分量又都不变，因此变型前后模型的细节能够很好地保持。

保持整体形状。在这个算法的过程中，由于柱坐标的半径分量 r 在形变过程中不变，因此整体形状上不会出现薄壳形变中出现的体积失真的情况。

保持其拓扑性。由公式 4.2 可知，在形变之前任意两点柱坐标角度分量相比

较大的点，变型后仍然相对较大，即不会被之前角度相对较小的点所超过，因为形变过程是一个保持其原先角度分量大小顺序的角度缩放的过程。

算法的实时性：由于这个方法是一个线性的方法，所以具有较好的实时性，对于含有 1966 个节点的膝关节网格模型，在 CPU 为 Intel 1.6GHz，内存为 512MB 的计算机上达到约 40fps，满足了实时性和真实感的要求。

总体看来这个方法的形变结果具有较好的逼真度，而且具有良好的实时性，能够满足虚拟手术中模型形变的逼真度和实时性要求。

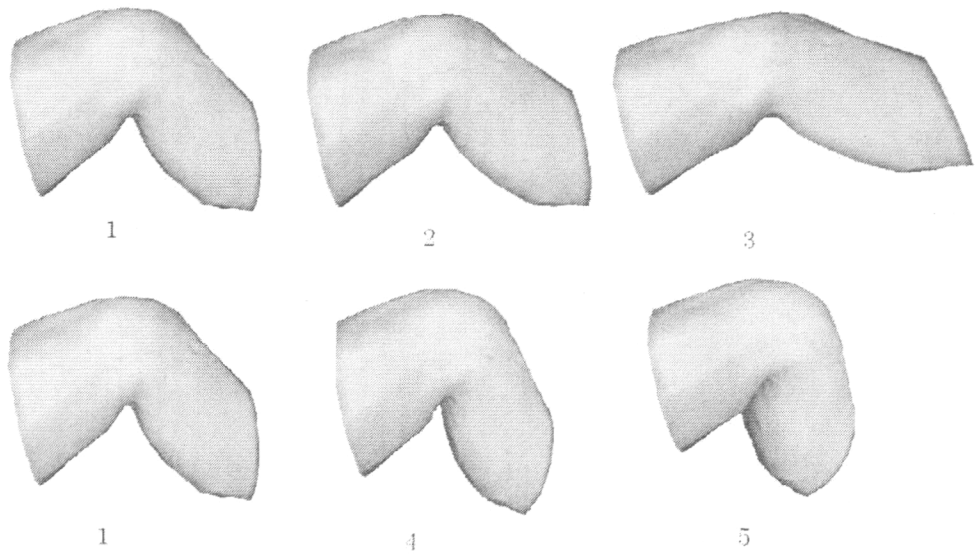

图 4.4 膝关节双层 3D 模型的形变

（1）原始模型；（2）（3）夹角变大的过程；（4）（5）夹角变小的过程。

第五章 改进的线性微分坐标形变方法

本章我们在现有的线性微分坐标形变方法的基础上作了改进，使其更加适合应用于我们的工作。

5.1 线性微分坐标形变方法介绍

本方法所用到的模型是三角网格模型。三角网格模型是一种比较常见的表面网格模型，它的特征是网格中的所有面片都是三角形。它包含足够多的模型表面信息，表示方式不会带来二义性，其形变有相对均匀和稳定的效果，因此三角网格常见于对模型的表示中。

对于 3D 表面三角网格模型的形变，Lipman[35]提出了一种利用拉普拉斯算子（Laplacian operator）的线性微分坐标方法，这种形变方法能够很好地保持原先模型的细节，可是当进行大范围形变的时候会产生失真，因此我们将对这个算法进行改进。下面首先介绍一下原来的方法。

5.1.1 表面微分坐标

首先设给定的三角网格为 $M=(V,E,F)$，点的个数为 n，V 是顶点集，E 是边集，F 是面集。给定的点用笛卡尔坐标表示 $v_i=(x_i,y_i,z_i)$。

定义点 i 的微分坐标为点的坐标与周围一环邻域内点的平均坐标的差，如图 5.1。点 i 的微分坐标可以表示如下：

$$\delta_i=(\delta_i^{(x)},\delta_i^{(y)},\delta_i^{(z)})=v_i-\frac{1}{d_i}\sum_{j\in N(i)}v_j \tag{5.1}$$

其中 $N(i)=\{j\,|\,(i,j)\in E\}$，$d_i=|N(i)|$ 是点 i 的一环邻域内的点的个数。

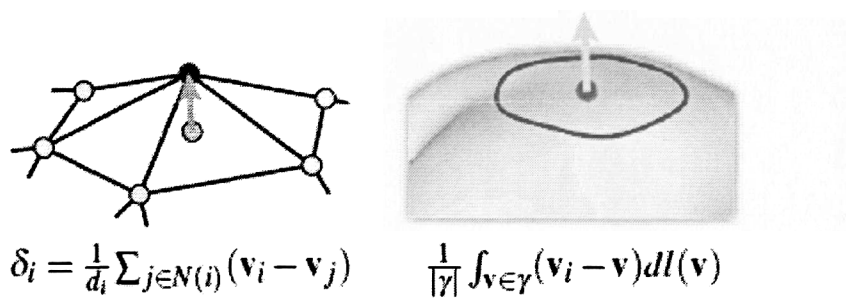

图 5.1 微分坐标示意图[29]

5.1.2 表面微分坐标的计算

由笛卡尔坐标到微分坐标的变换可以用矩阵的形式表示，设它为 L。为了得到矩阵 L，可以先设 A 为存储网格边的信息的矩阵：

$$A_{ij}=\begin{cases}1 & (i,j)\in E\\ 0 & \text{其它情况}\end{cases}$$

设D为对角矩阵，$D_{ii}=d_i$。于是可以得到坐标变换矩阵L：

$L=I-D^{-1}A$，将 L 进行标准化得到$L_s=DL=D-A$，即：

$$(L_s)_{ij}=\begin{cases}d_i & i=j\\ -1 & (i,j)\in E\\ 0 & \text{其它情况}\end{cases}$$

这个算子可以对模型上所有的点在 x，y，z 三个方向上的分量进行处理。

通过使用拉普拉斯算子对模型网格顶点的坐标进行运算，从而将模型的局部细节用微分坐标表示出来[34][36]：

$$\delta_i=\Delta s(p_i)=-H_i n_i \tag{5.2}$$

其中，Δs是拉普拉斯算子，p_i是点 i 的位置信息，H_i是点 i 处的局部平均曲率，n_i是点 i 处的表面法向量。直接用这种坐标表示就可以将表面模型的高频信息提取出来，带来保持细节的操作。

当然，只记录了表面模型高频信息的微分坐标并不足以得到最终结果，因为矩阵 L_s 是奇异的，因此，必须对形变后的模型提供位置约束，即：给定网格中某些点在形变后应该在的位置，这些位置约束和前面求出的所有点的微分坐标共同作用于形变系统中。通过解超定方程组能够得到形变结果：

$$\begin{pmatrix}\dfrac{L}{\varpi\cdot I_{m\times m}\,|\,0}\end{pmatrix}x=\begin{pmatrix}\delta'(x)\\ \varpi\cdot c_{1:m}\end{pmatrix} \tag{5.3}$$

这一个环节对应到实际的形变中就是用户在模型上选取的一些操纵点，用户通过赋值、拖拽等手段将这些点的位置信息改变，将这些信息加到求解的方程中去，以此得到受这些点的变化所影响的形变结果。

5.1.3 表面微分坐标的变换

如果直接采用原模型中的微分坐标生成新模型，而不考虑微分坐标跟随表面的旋转，会造成模型的失真，图 5.2 是一个表面细节旋转不变的效果示意图。图 5.2(a)是原模型，在平直的底面上有一些复杂的局部细节，图 5.2(b)是预期的正确形变结果，图 5.2(c)是微分坐标（表面细节）未经过旋转的形变结果。明显可见，未经过旋转的微分坐标

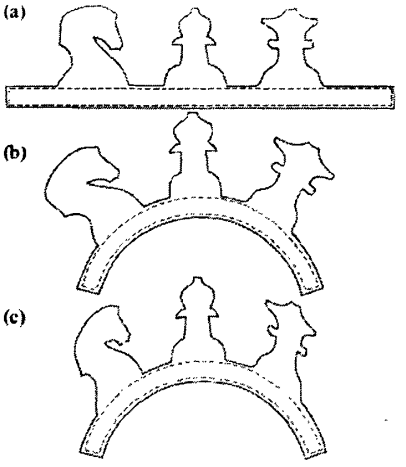

图 5.2 旋转不变效果示意图[29]

为此，应该对表面微分坐标进行相应的旋转。Zayer 提出的方法中使用了调和场的方法对微分坐标进行旋转[46]。对于每一点 V_i，对应的微分坐标的旋转矩阵可以用下式来计算：

$$T_i = slerp(R, I, 1-s_i)\cdot((1-s_i)S + s_i I) \tag{5.4}$$

其中 $slerp()$ 是四元数的插值函数，R 是控制点的旋转矩阵，S 是控制点的缩放矩阵，I 是单位矩阵，s_i 是介于 0 和 1 之间的权值，它决定了该点处的微分坐标的旋转和缩放的程度，s_i 由下式计算获得：

$$\begin{pmatrix} L \\ \hline \varpi \cdot I_{m\times m} \mid 0 \end{pmatrix} s_i = \begin{pmatrix} 0 \\ \varpi \cdot r_{1:m} \end{pmatrix} \tag{5.5}$$

由此可求出对每一点的微分坐标进行相应的旋转，令 $T = \{T_1, T_2, \ldots, T_n\}$，方程组 5.3 可变化为如下形式：

$$\begin{pmatrix} L \\ \hline \varpi \cdot I_{m\times m} \mid 0 \end{pmatrix} x = \begin{pmatrix} T \cdot I_{n\times n} \cdot \delta(x) \\ \varpi \cdot c_{1:m} \end{pmatrix} \tag{5.6}$$

5.1.4 计算过程的线性化

如果每次都直接解方程组（5.3），那么无法保证算法实时性，此算法根据方程组系数矩阵的特点，将解方程分解为预处理和回代结果的两个过程，每次求解只需做一次回代即可，大大减少了计算量。具体方法如下：

设方程（5.3）的系数矩阵为 $\tilde{L}$，由于此方，程为超定方程，它的解应该是一个能量最小化过程的解，设它为 $\tilde{x}$，则有：

$$\tilde{x} = \arg\min_x \left(\left\| Lx - \delta^{(x)} \right\|^2 + \sum_{j\in C} \varpi^2 \left| x_j - c_j \right|^2 \right) \tag{5.7}$$

可以根据超定方程（5.3）得到它的正规方程：

$$(\tilde{L}^T\tilde{L})\tilde{x} = \tilde{L}^T\delta \tag{5.8}$$

正规方程（5.5）的系数矩阵 $M = (\tilde{L}^T\tilde{L})$ 是对称正定的，因此可以在程序预处理的时候只对它做一遍 Cholesky 分解，把它分解为一个上三角矩阵 R 的转置与它自身相乘，即：

$$M = R^T R \tag{5.9}$$

从而将解方程分为两个步骤：

$$R^T\xi = \tilde{L}^T\delta^{(x)} \tag{5.10}$$

$$Rx = \xi \tag{5.11}$$

这两个方程是每次形变所作的计算，由于它们的系数矩阵都是三角矩阵，所以通过快速地回代就能得到结果，从而具有了较好的实时性。

5.1.5 算法评价

这个算法能够较好地保持模型的细节，而且允许用户通过操纵模型上的某些控制点而获得形变结果，并具有相对较小的计算开销，因此具有较好的用户交互性。但形变结果有可能出现在大范围上的失真。如图 5.3，折线 1 表示原始模型中的骨架，折线 2 表示形变后骨架的预期位置，折线 3 表示形变后从结果模型还原出来的骨架位置。线段 1 表示形变前模型上某两点之间的连线，线段 2 表示形变后这两点之间的连线。从图中可以看出，形变使新模型尤其是关节部分明显地偏离了正确的骨架的位置，未能符合预期的效果；模型中选取的局部两点的距离发生了较大变化，这表明这部分的体积未能得到很好地保持。

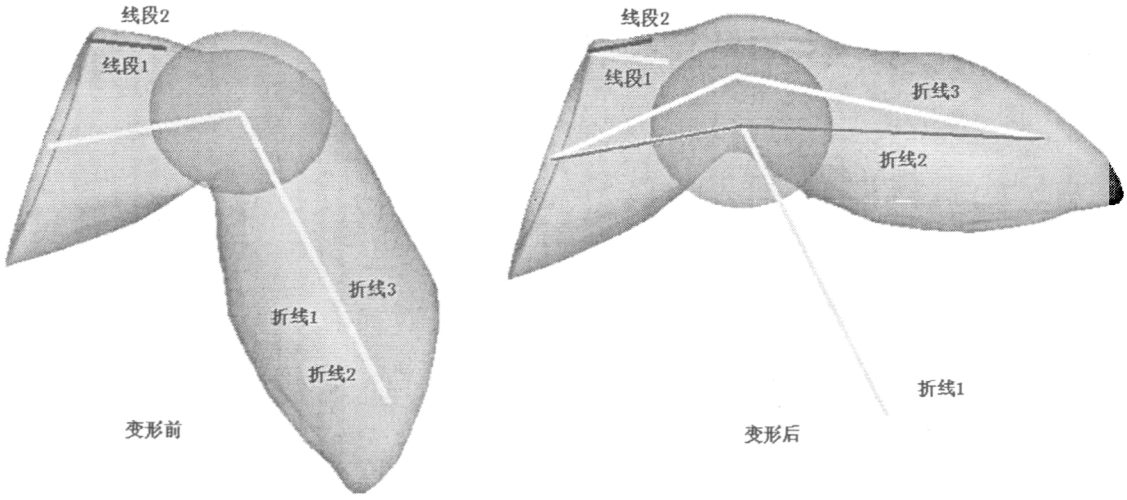

图 5.3 未经过改进的微分坐标形变方法

5.2 改进策略与改进效果

这个形变方法产生失真的原因是由于整个网格的形状被控制点和表面网格中点的连接方式所决定，尤其是在膝关节弯曲的部位，距离控制点较远，未能很好地受到骨架位置的约束，形变后的模型具有薄壳模型所具有的特点，在弯曲部位容易出现网格大范围偏离骨架的情况，即形变后的体积不满足真实情况而导致失真。为了解决这两个问题，我们将原算法作了如下改进：

5.2.1 基于骨架的形状修正

为解决形变后模型膝关节模型与骨架偏移过大的问题，我们采用了添加骨架连接处的控制点的方法，具体思想如下：选取骨架的交点，即旋转的中心点为圆心O，以某个合适的长度为半径作球体，取所有球上或者球内的点，设集合P是这些点的集合。用一种含有中心点位置信息的微分坐标来描述它的局部几何形状。

这种微分坐标是在传统微分坐标的基础上添加了与骨架连接点（旋转中心点）的关联而得到的。其做法是取圆心为控制点，将其加入网格模型中的点集V中，即 $V'=\{v \mid v \in V \text{或} v=O\}$，向边集E中添加新的边，使得 $E'=\{e \mid e \in E \text{或} e=(O,v) \text{或} e=(v,O),(v \in V)\}$。这个做法的意义是在获得网格表面微分坐标的时候将$O$点也算作$P$里面每个点的邻接点，这样求得的微分坐标不仅含有局部细节信息，而且含有整体的形状信息，其微分坐标如图 5.4 所示。在图 5.4 中，加粗的线条表示点 i 的微分坐标向量。

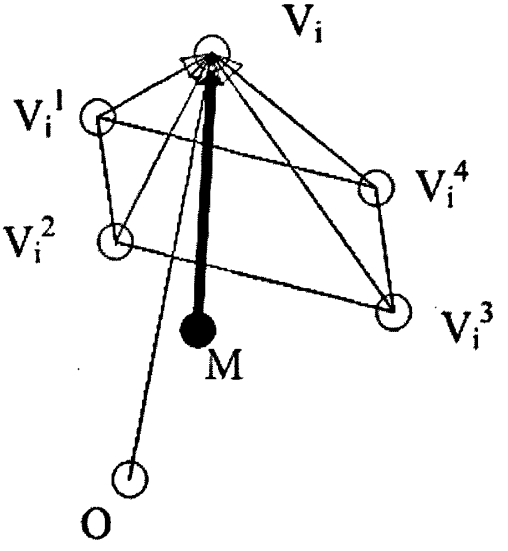

图 5.4 改进的微分坐标示意图

由此，点 i 改进后的微分坐标可以表示如下：

$$\delta_i = (\delta_i^{(x)}, \delta_i^{(y)}, \delta_i^{(z)}) = v_i - \frac{1}{d_i + 1}(\sum_{j \in N(i)} v_j + v_o) \tag{5.12}$$

我们将点 O 作为形变控制点之一，这个控制点的位置由形变后骨架的交点来确定，在超定方程组中添加一条关于点 O 的方程，把它的坐标赋值为初始坐标，并给这个方程的系数和等式右边同时乘以较大的权值。

通过添加骨架连接处的控制点，能够很好地保持形变后的骨架的位置，如图 5.5(c)，可是图中也可以明显看出，由于在采用新微分坐标的点与采用原微分坐标的点之间没有很好的衔接，在这两部分点连接的部分出现了较大的失真。

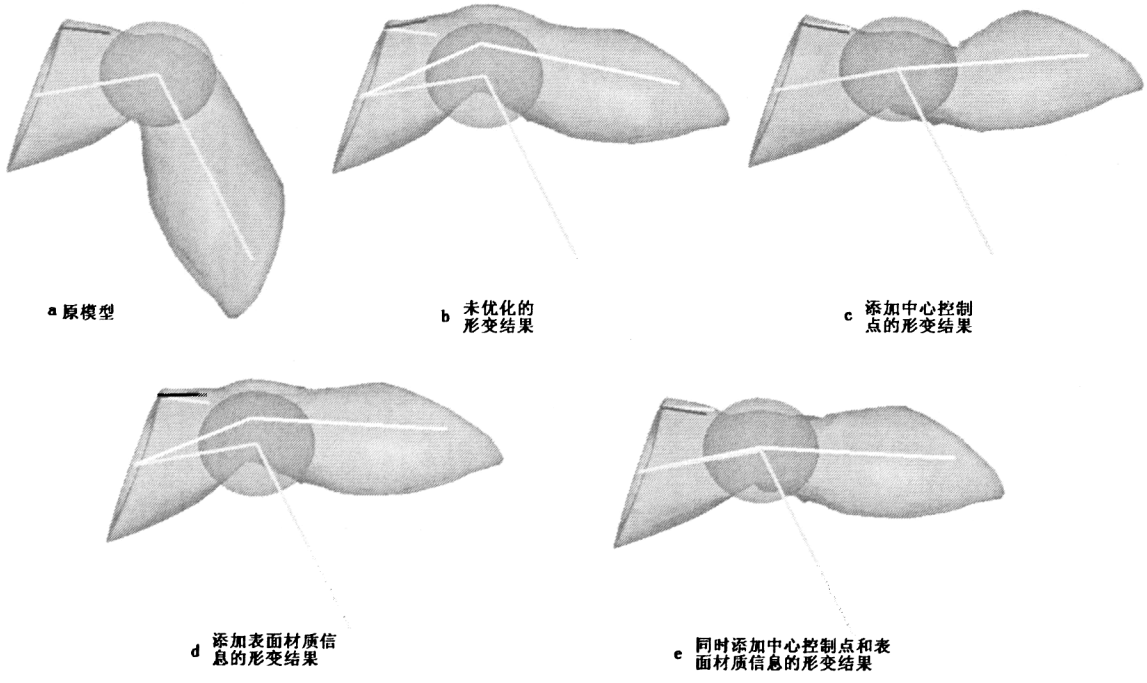

图 5.5 改进的线性微分坐标方法的形变效果

5.2.2 基于表面材质的形状修正

为解决表面微分坐标无法正确还原形状的问题，我们对每个点微分坐标所构成的方程的系数和方程右边同时进行了加权，权重较大的方程对应点的微分坐标的变化较小。

$$\hat{L} = LW, \quad \hat{I}_{m \times m} = [I_{m \times m} \mid 0]W \tag{5.13}$$

其中，$W = \begin{bmatrix} \varpi_1 & & & 0 \\ & \varpi_2 & & \\ & & \cdots\cdots & \\ 0 & & & \varpi_n \end{bmatrix}$，是记录每个点的权重的对角矩阵。

我们可以得到形变方程：

$$\begin{pmatrix} \hat{L} \\ \varpi \cdot I_{m\times m} \mid 0 \end{pmatrix} x = \begin{pmatrix} T \cdot \hat{I}_{n\times n} \cdot \delta(x) \\ \varpi \cdot c_{1:m} \end{pmatrix} \tag{5.14}$$

在离关节较远处，点的坐标更多地受骨架运动的影响，更多地体现出刚性的特征，可以把这部分的点所在的方程系数与比较大的权重相乘。

在离关节较近处，点的坐标应该体现出关节部位形变的特点，更多地体现出柔性的特征，它们所在方程的系数权重应该比较小。

而判断一个点是否离关节较近，我们在骨架的交点即旋转的中心位置为圆心，取某个合适的半径（使这个球体能够包含完整的关节），判断该点是否落在球体中，以此决定权重ϖ，如果点在球体中或球体上，则权重ϖ取较小值，如果点在球体外，则权重ϖ取较大值。

对形变过程同时进行添加骨架连接处控制点和添加表面材质信息，将得到较好的形变效果，如图 5.5(e)。但如果仅仅添加表面材质信息，无法保证形变结果符合骨架的正确位置，如图 5.5(d)。

5.2.3 改进策略小结

第一个改进的意义是将形变最明显也最容易失真的位置上的每个点都添加了一个全局的约束，减小这些重要部位形变后位置的偏移。

第二个改进的意义在于能够表现出不同类型表面的形变特征，使算法能够根据模型的骨架或者不同部位的材质，表现出逼真的形变效果。

这两个方面的改进互为补充，只有同时使用才能获得好的形变结果。

5.3 算法的过程

第 1 步，读取网格上的所有点、边和面的信息。控制点选取在模型的两端，即远离关节的位置，把它们在线性表中的序号记录下来。

第 2 步，根据点与关节中心点的距离建立关节附近点的集合 P，对网格区域进行划分，向其中加入旋转的中心点O，以及O点与P中所有点所连接的边，建立标准化的拉普拉斯算子L。

第 3 步，根据前一步的划分，求得每个点所在方程的权重ϖ。在P中的点，其权重较小，P以外的点，其权重较大。从而得到表示每个点所在方程权重的矩阵：

$$W = \begin{bmatrix} \varpi_1 & & & 0 \\ & \varpi_2 & & \\ & & \cdots\cdots & \\ 0 & & & \varpi_n \end{bmatrix} \tag{5.15}$$

第 4 步，对原始网格，利用改进的拉普拉斯算子求其表面顶点处的微分坐标 $\delta(x)$，即 $\delta(x)=\hat{L}x$，其中 $\hat{L}=LW$。

第 5 步，对求解超定方程 $\begin{pmatrix}\hat{L}\\ \hat{I}_{m\times m}\,|\,0\end{pmatrix}x=\begin{pmatrix}\delta(x)\\ \varpi_{1:m}\cdot c_{1:m}\end{pmatrix}$ 做预处理：设方程系数矩阵为 $\tilde{L}$，将方程两边同时左乘以 $\tilde{L}^T$，从而将方程化为正规方程：

$$(\tilde{L}^T\tilde{L})\tilde{x}=\tilde{L}^T\begin{pmatrix}\delta(x)\\ \varpi_{1:m}\cdot c_{1:m}\end{pmatrix} \tag{5.16}$$

设方程（5.11）的系数矩阵是 $M=\tilde{L}^T\tilde{L}$，由于 M 具有正定对称性，因此对 M 进行 Cholesky 分解，得到上三角矩阵 R，满足 $M=R^TR$。

第 6 步，根据骨架的运动状态得到控制点 $c_{1:m}$ 的新位置的坐标，根据骨架的位置，选取模型中远离关节处的点，重新获得它们的位置，其中包括确定中心点 O 的位置，通过调和场方法确定所有微分坐标的变换 T，并且确定正规方程的右边的新值 $\tilde{L}^T\begin{pmatrix}T\cdot\hat{I}_{n\times n}\cdot\delta(x)\\ \varpi_{1:m}\cdot c_{1:m}\end{pmatrix}$。

第 7 步，解下三角矩阵方程 $R^T\xi=\tilde{L}^T\begin{pmatrix}T\cdot\hat{I}_{n\times n}\cdot\delta(x)\\ \varpi_{1:m}\cdot c_{1:m}\end{pmatrix}$，求得中间结果 ξ；解上三角矩阵方程 $Rx=\xi$，得到最终结果 x 即为网格上所有点的新坐标。返回第 6 步，继续更新控制点的信息。

在这七个步骤中，前五步是预处理过程中的处理流程，第 6 步和第 7 步是算法的主循环，是能够进行实时交互的部分。

5.4 算法评价与测试

该算法在微分坐标形变方法的基础上进行了两处改进，分别是添加了超定方程组中方程的权重和添加了形变的中心约束。该改进算法在保持原算法保持细节和实时性等特点的同时，能够更好地按照骨架进行形变和保持体积，极大地提高了形变的逼真程度。从图 5.5 中我们可以清楚地看到，未改进的微分坐标方法在形变的过程中会出现未按照骨架进行形变、体积压缩的情况，而改进的微分坐标方法在形变过程中不会出现这么明显的失真。

实时性方面，这个方法的预处理环节的计算开销较大，不过每次循环的时候求解方程组就转化为求解两个三角矩阵的方程组的问题，而解三角矩阵方程组是一个线性回代的过程，实时性较高，对于约 2000 个顶点的模型，经过 TAUCS 图形运算库加速，刷新率可达到 15 帧/秒以上。

第六章 虚拟膝关节手术中的大范围形变的实现

本文提出的算法已应用于我们所开发的虚拟手术系统中，该系统能够将人体膝关节模型进行实时的大范围形变，具有较为逼真的虚拟效果。本章首先介绍虚拟手术系统，然后介绍在实现过程中所涉及到的相关技术，接下来简要介绍实现的过程，最后给出了实现的结果。

6.1 系统软硬件平台

“虚拟膝关节镜手术系统”是一个能在普通的 PC 机上运行，基于 Windows 操作系统平台，使用 Microsoft® Visual C++ 6.0 开发环境和结合 OpenGL 图形库进行开发的。整个系统成本比较低，通用性比较好，可移植性较强。系统还包括 PHANToM®触觉交互设备（如图 6.1）和与之相应的 GHOST SDK 力反馈应用软件开发包，以提供虚拟手术过程中触觉反馈的硬件和软件支持。

6.1.1 PHANToM®触觉交互设备

PHANToM®是由 STI 公司（SensAble Technologies Incorporated）生产的触觉交互设备，我们目前使用的是它的 DesktopTM 版本（如图 6.1）。使用时，用户通过用手操纵 PHANToM 笔尖（stylus）与设备进行交互。PHANToM® DesktopTM 设备的技术指标如表 6.1 所示[37]，可以看出，它可以接受 6 个自由度的空间定位输入信息（位置和旋转），并且可以提供 3 个自由度的力反馈输出（只包括位置）。PHANToM® DesktopTM 设备的工作区间与人的小臂活动的范围相当，位移精度为 0.02mm，最大外力可达 6.4N，可以满足虚拟膝关节镜手术和心脏介入仿真的需求。

表 6.1 PHANToM® Desktop™ 设备技术指标

项目	数据
工作区间	16×13×13cm
活动范围	手腕和前小臂弯曲的活动范围
常规位置分辨率	0.02mm
最大载荷	6.4N
持续工作载荷（24 小时）	1.7N
惯量	75g
反馈力方向	x, y, z
位置传感器	x, y, z, pitch, roll, yaw
计算机接口	通过并口与计算机连接，无特殊要求
运行平台	基于 Intel 的 PC 机

PHANToM® DesktopTM 设备通过并口与计算机连接，使用非常方便。

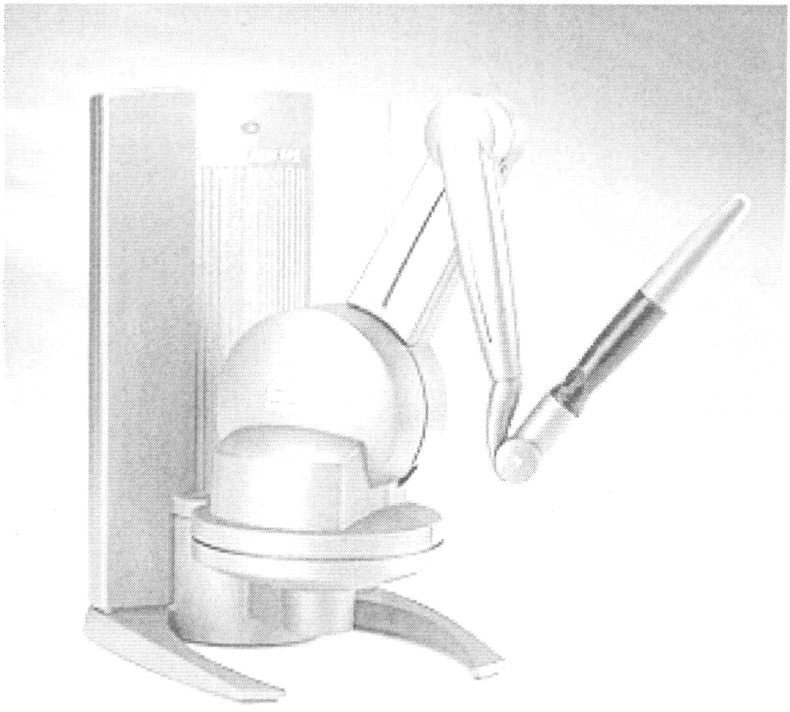

图 6.1 PHANToM® Desktop™ 触觉交互设备

6.1.2 GHOST 软件开发包

GHOST (General Haptics Open Software Toolkit) SDK 是 STI 公司为 PHANToM®设备开发的一个力反馈应用程序开发工具包，我们的系统中使用的版本是 3.0。GHOST SDK 采用 C++面向对象的设计方法，它提供了一种抽象的几何物体和空间效果的层次集合，来表示触觉环境，使得应用程序开发人员可以将主要精力放在触觉场景的产生、场景及场景中物体属性的处理和最终效果的控制上，而不必注意底层的力学和设备接口的问题。而且 GHOST SDK 的通用性很高，它可以支持所有 STI PHANToM®系列的触觉交互设备。GHOST 应用程序界面(API)使应用程序开发人员可以在物体和效果的层次上与触觉交互设备交互，便于创建触觉环境。GHOST SDK 主要有以下三个特点[38]：

(1) GHOST SDK 将虚拟环境中的所有物体以一棵树的形式组织成触觉场景图（haptic scene graph）。在触觉场景图中，树的非叶子节点用来将它的后代节点分组，相对于父节点旋转和缩放子树，以及为子树添加动态属性。树的叶节点表示实际的几何体。叶节点也包含相对于父节点的旋转和缩放。

(2) 触觉交互设备的终端表示成触觉场景图中的一个节点。GHOST SDK 将会根据给定的参数自动的计算出这个节点与触觉场景内的物体之间的作用力，并将这些力通过触觉交互设备表现出来。

(3) GHOST SDK 不在触觉场景图中生成物体的视觉表现。它没有对图形显示的系

统调用，图形的显示要运用其它的平台作支撑。GHOST SDK 提供了回调(callback)机制，使得触觉交互部分和图形表现部分更容易集成。现在，GHOST SDK 已被成功的与包括 OpenGL 和 Open Inventor 在内的一些图形学软件包一起配合使用。

一个标准的 GHOST SDK 应用程序至少应该包含两个进程：应用程序进程和触觉仿真进程，它们采用异步交互的方式运行。应用程序进程产生并设置一个触觉场景图，然后启动触觉仿真进程。触觉仿真进程只负责根据这个设置好的触觉场景图实现力反馈，它能维持很高的刷新频率，GHOST SDK 的标准频率为1000Hz。应用程序进程本身实现应用系统所需的其它功能，包括场景中图形的各种计算和绘制（包括形变计算），同时在需要的时候与触觉仿真进程产生联系，它一般维持 30Hz 左右的刷新频率。

6.2 相关技术

6.2.1 OpenGL 图形库

OpenGL[39]即开放图形库(Open Graphic Library)，是一个功能强大而且高效率的三维图形标准，它是在 SGI 等多家世界闻名的计算机公司的倡导下，以 SGI 的 GL 三维图形库为基础制定的一个通用共享的开放式三维图形标准。目前，Microsoft、SGI、IBM、DEC、SUN、HP 等大公司都采用了 OpenGL 做为三维图形标准，许多软件厂商也纷纷以 OpenGL 为基础开发出自己的产品，其中比较著名的产品包括动画制作软件 Soft Image 和 3D Studio MAX、仿真软件 Open Inventor、VR 软件 World Tool Kit、CAM 软件 ProEngineer、GIS 软件 ARC/INFO 等等。由于 Microsoft 公司在 Windows NT 和 Windows 95 中提供了 OpenGL 标准，而且 OpenGL 三维图形加速卡的推出，因此 OpenGL 在微机中有着广泛地应用，同时也使广大用户能够在微机上使用多数支持 OpenGL 的软件。

OpenGL 由一系列与硬件、窗口系统和操作系统相独立的 API 组成，高性能的三维绘图功能和良好的移植性使得 OpenGL 已成为目前的三维图形开发标准，是从事三维图形开发工作的技术人员所必须掌握的一类开发工具。OpenGL 主要包括三个函数库，它们是核心库、实用函数库和编程辅助库。

OpenGL 的工作顺序是从定义几何要素到把象素段写入帧缓冲区的过程。在屏幕上显示图象的主要步骤是：

第一步：由基本几何要素，如点、线、多边形、位图等建立几何模型，创建对模型的数学描述。

第二步：设定对象在三维空间上的位置，选择视点。

第三步：计算模型的颜色，光照，纹理等。

第四步：将模型的数学描述和颜色信息转换至屏幕的象素，即光栅化。

在这四步的执行过程中，也可能执行其他的一些操作，比如自动消隐处理等等。整个流程如图 6.2 所示。

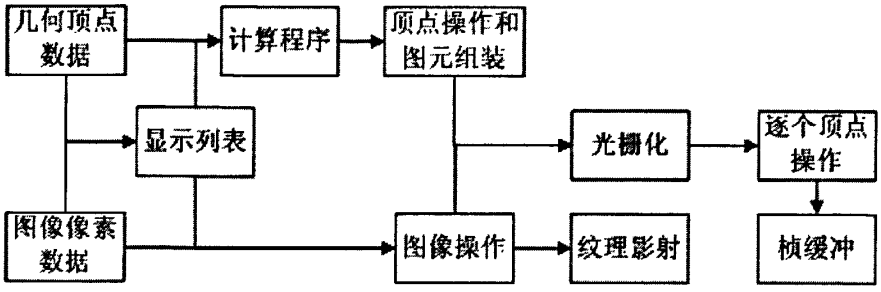

图 6.2 OpenGL 的图形绘制流程

本文所有的算法的实现，与图形有关的部分，一般都是基于 OpenGL 图形库来进行的。

6.2.2 VRML 语言

VRML[40][41][42]（Virtual Reality Modeling Language），即虚拟现实造型语言，是一种用于描述三维造型与交互环境的简单的文本语言。随着第二代 Web 把 VRML 与 HTML、Java、媒体信息流等技术有机地结合起来，形成一种新的三维超媒体 Web 并不断普及，VRML 的地位已越来越重要，它已经成为在 Internet 上建立 3D 多媒体和共享虚拟世界的一个开放标准。

由于 VRML 与平台无关，是基于 Web 的，能够快速建模，并且大量的模型都是以这种形式表示的，因此在应用中具有很大的优势。

我们所采用的模型就是以 VRML 的形式表示的。

所有的 VRML 文件都是以扩展名 WRL 结尾的文本文件或以 WRZ(压缩格式)结尾的二进制文件，它一般包含如下 4 个部分：

(1) 文件头：位于 VRML 文件的首行，提供文件的版本信息。对于 VRML 2.0，它是：#VRML V2.0 utf8 。

(2) 注释：以#号开始的一段文字。

(3) 节点：场景信息的单位。可以用它来描述场景中的造型、灯光及声音等，如：Cylinder { ... }描述的就是场景中的一个圆柱体节点。

(4) 域值：域用于描述及改变节点的属性，值反应了域的大小。

如：

Cylinder { height 2.0
radius 0.5 }

表示了一个高为 2.0，半径是 0.5 的圆柱体。

我们的程序中要对膝关节模型的 VRML 文件进行读取，获取其中的顶点位置信息、边的信息、面的信息和纹理信息等等。

6.2.3 TAUCS 库

TAUCS[43][44]是以色列特拉唯夫大学的一个团队开发的函数库，可用于图形图像处理过程中的线性求解稀疏矩阵方程。目前最新版本是 2.2 版。

在 TAUCS 库中，包含了用于解对称正定矩阵方程的，目前最新的 Cholesky 分解技术，（甚至比 MATLAB6 中的稀疏矩阵 Cholesky 分解还要快几倍[44]），库中运用了松弛和合并的技术极大地提高了效率。

库中还运用了外部存储的 Cholesky 分解技术，使分解的中间结果能够保存在硬盘中，以提高下次求解的速度。

库中还提供了矩阵的相乘、三角矩阵方程的求解、矩阵的重排序等操作。

库中还提供了稀疏矩阵的输入方法，通过一个简单的文件格式，给出每个非 0 元素的行号、列号和值，就能够输入一个完整的稀疏矩阵。

库中还提供对图形的预处理的支持，能够提供对许多种图形图像计算的预处理。

TAUCS 库能够提供单精度和双精度的浮点运算。

不过，TAUCS 库在使用前需要在本地机器上生成一次库文件，以保证能够正确运行，同时也是库文件在本地系统中的优化过程。

在我们的实现中，尤其是对改进的线性微分坐标形变方法，TAUCS 库的作用非常明显，它能够极大地提高数据量较大的网格模型的形变效率，是增强系统实时性的一个很得力的工具。

6.3 实现过程

我们将按照模型的读取、模型的表示、模型的计算和模型的绘制的顺序介绍形变程序的实现过程。

6.3.1 模型的读取

我们要进行处理的模型是膝关节用双层表面网格表示的模型，模型存储在以 VRML 语言表达的 leg.wrl 文件内。为使形变更加真实，同时也以点集的形式给出

了横截面的数据，存放在 leg.cnt 文件内。下面分别介绍这两个文件的读取

首先介绍 leg.wrl 文件的读取，我们构造了一个包含与点有关的各种信息的类，它能够存放从模型文件读出的所有内容。在读取文件信息之前，构造了一个存放点类的指针的顺序表，然后开始读文件。每读入一个顶点，就申请一块内存空间存放该点对象，然后把指向这个点对象的指针加入顺序表。

leg.wrl 文件的主体部分记录了 4 类信息，包括点的位置信息，点的连接信息，法向量的方向信息和法向量的连接信息。我们对它们的读取方法如下：

(1) 顶点坐标信息

coord DEF GEOMETRY-COORD Coordinate { point [

-52.890 86.280 -227.327, -34.840 80.990 -227.327, -23.690 79.620 -227.327,

……

-156.320 -273.030 258.346, -96.580 -128.160 258.346]}

在文件中每个顶点的坐标由三个连续的浮点数表示。它们由空格隔开，每两个点的坐标之间用逗号隔开。在读取的时候，将每个以逗号隔开的三元组依次存入点坐标的三个分量。

(2) 顶点法向量信息

normal Normal { vector [0.1665 0.4943 -0.8532, 0.0994 0.5202 -0.8482, 0.2737 0.6930 -0.6669,

……

-0.2865 -0.3161 0.9044, 0.8985 0.3207 -0.2999,] }

normalPerVertex TRUE

与顶点的坐标信息类似，每个顶点的法向量由三个连续的浮点数表示。它们由空格隔开，每两个法向量坐标之间用逗号隔开。在读取的时候，将每个以逗号隔开的三元组依次存入点的法向量的三个分量。

(3) 顶点连接信息（三角面片信息）

coordIndex [31, 30, 101, -1, 44, 30, 31, -1, 43, 42, 113, -1, 43, 32, 42, -1,

……

1962, 1892, 1891, -1, 1966, 1886, 1885, -1]

这组数据表明了顶点与顶点之间的关系，即三角面片的信息。每个面片由三个点组成，文件里用三个点的序号构成，序号由逗号隔开，每个面片的信息由-1 隔开。

(4) 法向量的邻接信息

normalIndex [36, 35, 106, -1, 49, 35, 36, -1, 48, 47, 118, -1, 48, 37, 47, -1,

……

2009, 1928, 1927, -1, 2013, 1921, 1920, -1,]}

与顶点的邻接信息类似，这组数据表明了顶点法向量与顶点法向量之间的关系。在这里每个面片由三个点组成，文件里用三个点的序号构成，序号由逗号隔开，每个面片的信息由-1 隔开。

接下来介绍 leg.cnt 文件的读取。leg.cnt 文件中的数据是一系列横截面上的采样点集，其表达格式如下：

```
S  30
v 80 z -227.586334
{-52.84 86.23
……
-57.59 48.39}
v 82 z -210.749664
{-55.69 89.38
……
-59.42 52.56}
……
```

S 表示每个横截面的点的个数，z 是横截面上所有点的共同的坐标分量 z，在“{}”中每一行都存放着对应点的另外两个分量 x，y 的值。我们把它们读取之后存放在另一个线性表中。

6.3.2 模型的表示

我们用 Vertex3d 类存储模型的顶点坐标信息，主要包括点的坐标，点的法向量，点的邻接点等等，并封装了对点的相关信息进行的操作。如表 6.2，6.3 所示：

表 6.2 Vertex3d 类主要数据成员的意义

数据成员名称	意义
m_Coord[3]	点的坐标
m_Normal	点的法向量
m_Color	点的颜色
m_Flag	点的标志位
m_ArrayVertexNeighbor	一环邻域内的点
m_ArrayFaceNeighbor	一环邻域内的面
……	……

Vertex3d 类可以存储一个顶点的所有位置、方向和与外界连接情况等信息，如果构造一个线性表将一系列 Vertex3d 类的对象存储起来，就能够将所有顶点组织成一个完整的网格模型。

表 6.3 Vertex3d 类主要函数的意义

函数名称	意义
Get	得到点的坐标
Set	设定点的坐标
GetNormal	得到点的法向量
SetNormal	设定点的法向量
HasNeighbor	判断一环邻域是否有点
GetVertexNeighbor	得到一环邻域上的点
GetFaceNeighbor	得到一环邻域上的面
NbVertexNeighbor	一环邻域内的点的个数
NbFaceNeighbor	一环邻域内的面的个数
AddNeighbor	在一环邻域内增加点
RemoveNeighbor	在一环邻域内删除点
GetNormal	一环邻域内的面
GetNormal	一环邻域内的面
……	……

Vertex3d 类封装了大量对顶点的操作，这提供了对上层的操作的透明性，编写形变程序的时候可以不会被大量对顶点的底层操作所困扰，为编写形变程序过程中的各个步骤提供了巨大的方便。

对于从 leg.cnt 文件读取到的横截面上点的坐标，由于我们只需通过它们得到骨架，因此我们把它们存放在三维线性表 m_Section 中。

6.3.3 模型的计算

模型的数据在内存中组织起来之后，接下来的工作就是对这些数据进行处理。我们可以按照我们提出的方法通过横截面上的点的坐标构建骨架，然后根据骨架的位置产生模型上的选定的控制点的新坐标。从力反馈设备上获得用户输入的控制信息决定骨架的运动，由骨架的运动带动控制点，产生控制点的新坐标，结合原始模型的信息，按照我们给出的形变算法，能够对模型上所有点的位置信息进行计算，形变计算的结果就是最后的形变效果，可以直接向视觉和触觉渲染设备输出，也可以再添加纹理后输出。这个过程如图 6.3 所示：

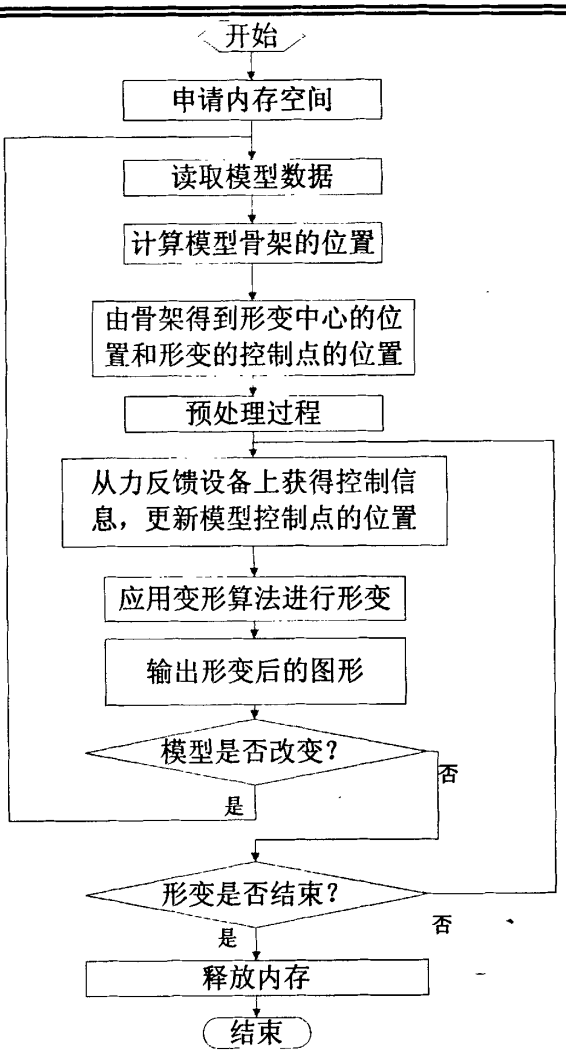

图 6.3 主程序流程图

6.3.4 模型的输出

模型的绘制流程如图 6.4 所示。形变程序与器官模型类 CSoftOrganModel 有一个接口，可以通过这个接口将形变后的数据送到整个系统中，以供显示、碰撞检测和进行力反馈计算。

这个流程主要负责将模型数据的计算结果传递到视觉和触觉渲染设备上，是能够形成与用户之间交互的反馈回路必不可少的部分，它效率的高低也会对实时性产生影响。在实际应用中可以考虑使用多线程技术与形变程序并发运行。

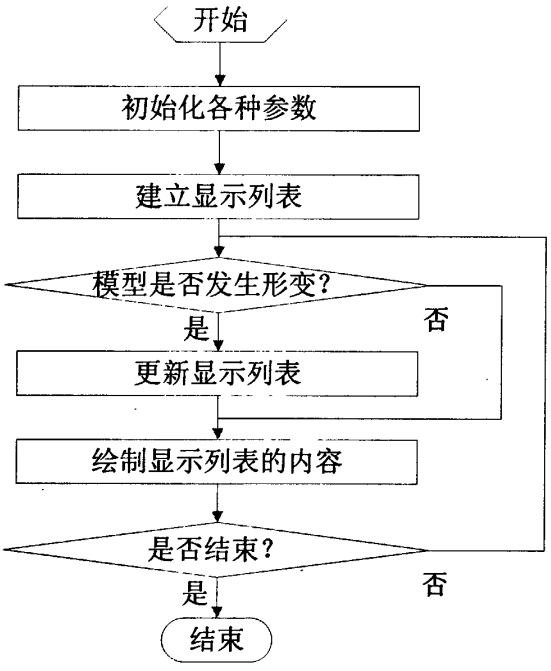

图 6.4 模型绘制的流程图

6.4 实现结果

该系统可以对膝关节表面网格模型进行符合人体构造的形变。

系统中，能够从多个角度动态显示膝关节的形变效果，原模型的弯曲角度大约是 90 度，在系统中能够弯曲 30 度到 180 度之间的任意角度，可以通过人工交互来控制形变的过程，这个膝关节动态形变系统也可以自动播放描述形变过程的动画，可以调节形变的速度的快慢，设定形变区域，人工修改形变中心，同时能够提供网格的平滑处理和光照效果。系统执行效果如图 6.5，本文展示了分别从正面和反面两个视点观察到的形变效果。

该系统能提供比较符合物理特性的形变，关节骨架弯曲的角度也能基本够包含实际操作中的所有情况，对模型形变的模拟具有比较高的逼真性，并且系统刷新频率在 15~40 帧，具有较好的实时性。

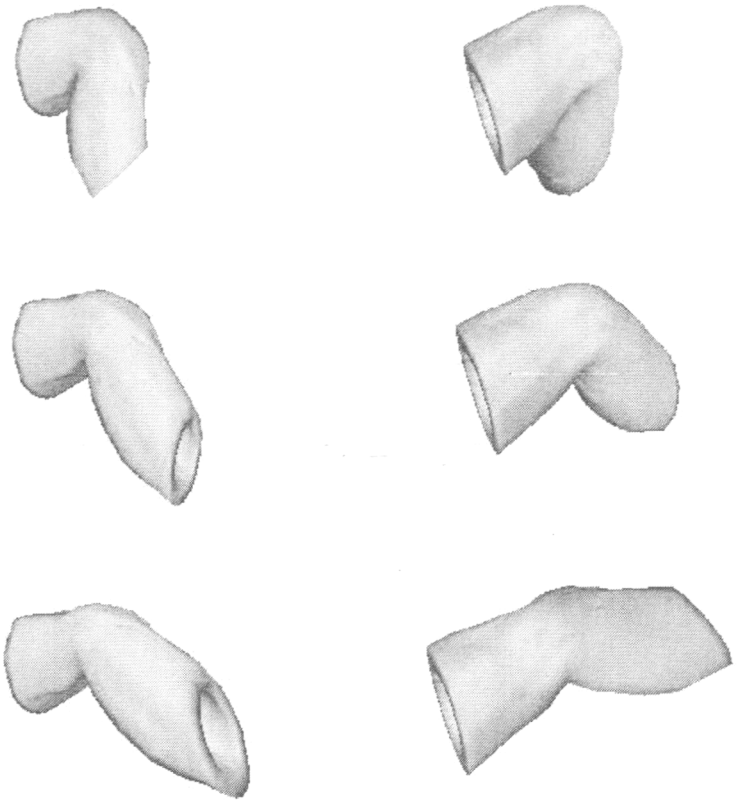

图 6.5 系统执行效果演示图

第七章 总结与展望

7.1 工作总结

在一个成功虚拟手术系统中，对人体模型进行逼真的、实时的形变是必不可少的，本文着重研究了人体表面模型的大范围实时形变方法。在研究的过程中，我们能够针对模型和虚拟手术的特点，充分地学习、利用和改进现有的技术，能够使研究和实践相结合，解决虚拟膝关节手术大范围形变的问题。本文的工作主要有以下三个方面：

(1) 提出了由模型的截面构建骨架的方法。形变的过程是一个由骨架驱动的过程，数据中没有给出模型的骨架，可是我们可以知道模型横截面的信息。因此，可以通过对模型横截面信息的处理得到它的骨架。我们通过对截面中心点进行划分和线性拟合，得到了模型的骨架信息。

(2) 提出了柱坐标平滑变化的形变方法。根据膝关节 3D 模型的特点，利用模型上的顶点的柱坐标的变换来实现大范围形变，变换的过程能够保持模型在柱坐标系下的局部细节，从而使最终在笛卡儿坐标系下的形变结果保持令人满意的局部细节特征，算法能满足虚拟手术的实时性和真实感的要求。这种方法也能推广应用于其它类似的关节弯曲运动的实时形变。

(3) 对现有的线性微分坐标形变方法进行了改进。线性微分坐标的形变方法对模型的细节保持的很好，而且允许用户定义模型上的某些点的形变趋势，并具有较小的计算开销，然而它的形变结果可能在较大范围上出现失真。因此，我们对它进行了改进，添加了骨架姿态和表面硬度对形变的约束。改进后的方法较好地克服了原方法出现的失真问题。

除了应用于虚拟膝关节手术的形变模拟，本文的方法也可以用来解决结构类似于关节的模型的形变问题，如肘关节形变等等。

7.2 工作展望

根据虚拟手术系统的具体需求和当前形变技术的发展趋势，在今后的工作中需要考虑和研究的问题包括以下几个方面：

(1) 实时性更强的形变。视觉所能感知到的连续性要求刷新率达到 25Hz 以上，我们的算法刷新率高于 25Hz，因此能够达到视觉上的连续性。然而在力反馈设备上，刷新率要达到 300Hz 以上，触觉才能够感知到连续性。如果要通过力反馈设备和正在形变的模型动态地进行交互，目前还没有找到合

适的实时形变方法。因此，我们可以进行进一步研究，找到有更高的刷新率同时又能保证较好形变效果的方法。

(2) 用 GPU 对算法进行加速。近年来出现了许多在 GPU 上计算形变效果的方法，它们利用了 GPU 在矩阵运算方面的优势，将大量的复杂的矩阵运算放到 GPU 上完成，为了提高实时性，可以考虑以后将我们的方法中的矩阵运算在 GPU 上完成。

(3) 提出通用性更强的方法。我们的方法目前在表面模型上能表现出较好的效果，但在模型结构或者形变动作发生改变的情况下可能会失效。比如柱坐标平滑变化的形变方法不适用于更复杂的姿态，比如关节的轴向旋转等等。而改进的线性微分坐标形变方法对于体模型或者其他类型的模型，可能不太适用。因此可以考虑研究更为通用的方法。

致 谢

在此，我首先感谢我的导师熊岳山教授给予我的悉心指导。熊老师孜孜以求的敬业精神、富有前瞻性的敏锐洞察力、求真务实的严谨学风、审时度势的统筹意识，都给我留下了极其深刻的印象，也使我从中汲取了不少宝贵经验，这些都将使我受益终身，同时也将激励我奋力拼搏、不断进取。在学习研究方面，论文选题、资料查询、课题研究，直到最终的论文撰写，熊老师宽广深厚的专业知识给了我极大的支持，严谨求实的作风更使我受益匪浅。生活方面，熊老师平易近人，给了我很多关怀，为我排除了很多干扰，他的鼓励使我充满信心。熊老师渊博的学识、勤勉的作风，永远是我们学习的榜样。在学位论文即将完成之际，谨向多年来培养和关怀我的熊岳山教授致以由衷的敬意和诚挚的谢意！

我还要感谢先期进入课题的徐凯师兄、王彦臻师兄，同时他们也是我的好朋友，在课题研究过程中，他们毫无保留地把广博的知识和宝贵的经验告诉我，对我的研究起到了重要的帮助作用。感谢陈欣师姐和姜菲师姐给予我的大量帮助与支持，在学习和研究方面，她们，对我有很多帮助，也教会我许多东西。同时非常感谢我的亲密战友康勇，他勤奋刻苦，好学上进，勇于创新的精神非常值得我学习，与他们一起探讨课题和和切磋技术的情景至今仍历历在目，他对于我的学习和生活中遇到的问题都给与了极大的帮助和支持。同样感谢实验室的同学们，室友们，与他们相处的快乐时光永远无法忘怀。

感谢李佑国，崇金山，刘东师弟对我的帮助，他们积极踊跃，善于思考和发现问题，在讨论问题的时候，他们给了我很多启发；在我遇到难题的时候，他们又能够及时出手相助。

更要感谢我在远方的家人，父母给了我生命，养育我长大。他们是我的第一任老师，教会我做人的道理，而且一直鼓励我要专心致志，学有所成。感谢我的爱人尚洁女士，是她不断地鼓励我，支持我，主动为我分忧解愁。我因为研究工作没有好好地和她在一起的时候，她也没有半句怨言，她为我的研究工作的确做出了很大的牺牲。

参考文献

[1] Richard A. Robb，Jon J. Camp，Dennis P. Hanson，Computer Aided Surgery and Treatment Planning at the Mayo Clinic http://www.mayo.edu/bir/HTMLs/robb_CAS_95.htm

[2] Yamashita Juli，Yamauchi Yasushi，Mochimaru Masaaki，Fukui Yukio，Yokoyama Kazunori，Real-time 3-D model-based navigation system for endoscopic paranasal sinus surgery，IEEE Transactions on Biomedical Engineering，1999，46(1)：107-116

[3] G. Picinbono，J.-C. Lombardo，H. Delingette，N. Ayache. Improving realism of a surgery simulator：linear anisotropic elasticity，complex interactions and force extrapolation，Technical Report，INRIA，2000

[4] U. Kühnapfel，H.K. Çakmak，H. Maass.，Endoscopic surgery training using virtual reality and deformable tissue simulation，Computers & Graphics，Elsevier，2000，24(5)：671-682

[5] 李艳，王兆其，毛天露，三维虚拟人皮肤形变技术分类及方法研究，计算机研究与发展，2005 ，42（5）：888-896

[6] N. Magnenat-Thalmann，R. Laperrière，D. Thalmann，Joint-dependent local deformations for hand animation and object grasping，Proceedings on Graphics interface 88：26-33

[7] K. Komastu，Human skin model capable of natural shape variation，计算机科学 2005，8：265-271

[8] J.P.Lewis，M.Cordner，N.Forg，Pose space deformation：A unified approach to ship interpolation and skeleton-friben deformation，Siggraph 2000，Proceedings of the 27th annual conference on Computer graphics and interactive techniques：165-172

[9] Peter-Pike J. Sloan，Charles F. Rose III，Michael F. Cohen，Shape by example，Proceedings of the 2001 symposium on Interactive 3D graphics：135-143

[10] Alex Mohr Michael Gleicher，Building efficient，accurate character skins from examples，ACM Trans Graphics(TOG)，2003，21：562-568

[11] Botsch M.，Sorkine O.，On linear variational surface deformation methods，IEEE Transactions on Visualization and Computer Graphics，Jan/Feb，2008，vol. 14，no. 1：213-230

[12] Shin Yoshizawa，Alexander G. Belyaev，Hans-Peter Seidel，Free-form skeleton-driven mesh deformations，In Proceedings of the eighth ACM symposium on Solid modeling and applications，2003：247-253

[13] Sederberg T. W., Parry, S. R., Free-form deformation of solid geometric models, In Siggraph 86 Conference Proceedings: 151-160

[14] George Celniker, Dave Gossard, Deformable curve and surface finite-elements for free-form shape design, In Proceedings of ACM Siggraph, ACM Press, 1991: 257-266

[15] Yoshizawa S., Belyaev A., Seidel H.-P., Free-form skeleton-driven mesh deformations, In Proceedings of 8th ACM Symposium on Solid Modeling and Applications, 2003: 247-253

[16] Botsch M., Kobbelt L., An intuitive framework for real-time freeform-modeling. ACM Trans Graphics(TOG) 2004, 23(3): 630-634

[17] Welch, W., Witkin A., Variational surface modeling, In Proceedings of Siggraph 92, 157-166

[18] Boier-Martin I., Ronfard R., Bernardini F., Detail-preserving variational surface design with multiresolution constraints, In Proceedings of the Shape Modeling International 2004: 119-128

[19] Kun Zhou, Xin Huang, Weiwei Xu, Baining Guo, Heung-Yeung Shum, Direct Manipulation of Subdivision Surfaces on GPUs, July 2007, ACM Transactions on Graphics (TOG), Volume 26 Issue 3, Article No. 91

[20] Alexa M., Differential coordinates for local mesh morphing and deformation, The Visual Computer, 2003, 19(2): 105-114

[21] Yu Y., Zhou K., Xu, D., Shi X., Bao H., Guo B., Shum, H. Y., Mesh editing with poisson-based gradient field manipulation, ACM Transactions on Graphics (TOG), 2004, 23 (3) : 644-651

[22] Sorkine O., Cohen-OR D., Lipman Y., Alexa M., Rossl C., Seidel H.-P., Laplacian surface editing, In Eurographics Symposium on Geometry Processing 2004: 175-184

[23] Lipman Y., Sorkine O., Levin D., Cohen-OR D., Linear rotation-invariant coordinates for meshes, ACM Transactions on Graphics (TOG), 2005, 24 (3) : 479-487

[24] Lipman Y., Cohen -OR D., Gal R., Levin, D., Volume and shape preservation via moving frame manipulation., January 2007, ACM Transactions on Graphics (TOG), Volume 26 Issue 1, Article No. 5

[25] Zorin D. Schrodeer P., Sweldens W. , Interactive multiresolution mesh editing, In Siggraph 97 Conference Proceedings: 259-268

[26] Kobblet L., Campagna S., Vorsatz J., Seidel H.-P., Interactive multi-resolution modeling on arbitrary meshes, In Siggraph 98 Conference Proceedings: 105-114

[27] Guskov I., Vidimce K., Sweldens W., Schroder P., Normal meshes, In Siggraph 2000 Conference Proceedings: 95-102

[28] Marinov M., Botsch M., Kobbelt L., Gpubased multiresolution deformation using approximate normal field reconstruction, Journal of Graphics Tools, 2007, 12 (1): 27-46

[29] Sorkine O., Laplacian Mesh Processing, Eurographics 2005

[30] Botsch M., Pauly M., Gross M., Kobbelt L., Primo: Coupled prisms for intuitive surface modeling, In Eurographics Symposium on Geometry Processing, 2006: 11-20

[31] Mario Botsch, Mark Pauly, Martin Wicke, Markus Gross, Adaptive Space Deformations Based on Rigid Cells, Computer Graphics Forum , 26 (3): 339-347

[32] Oscar Kin-Chung Au, Hongbo Fu, Chiew-Lan Tai, Daniel Cohen-Or., Handle-Aware Isolines for Scalable Shape Editing, ACM Siggraph 2007

[33] Robert W., Sumner Johannes, Schmid Mark Pauly, Embedded Deformation for Shape Manipulation, ACM Siggraph 2007

[34] Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, Heung-Yeung Shum, Mesh editing with Poisson-based gradient field manipulation. ACM Transactions on Graphics, 2004, 23(3): 644-651

[35] Lipman Y., Sorkine O., Cohen-OR D., Levin D., Rossl C., Seidel H.-P., Differential coordinates for interactive mesh editing, In Proceedings of Shape Modeling International 2004, IEEE Computer Society Press: 181-190

[36] Marc Alexa, Differential coordinates for local mesh morphing and deformation, The Visual Computer, 2003, 19(2): 105-114

[37] SensAble Technologies, Inc. Products & Services: PHANToM Desktop Technical Specifications, [2006-10-30] http://www.sensable.com/haptic-phantom-desktop.htm

[38] GHOST SDK Programmer's Guide, Version 3.0, SensAble Technologies, Inc. 1999

[39] Addison-Wesley, OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 1.4, Fourth Edition. 2003

[40] http://cic.nist.gov/vrml/vbdetect.html

[41] http://www.bitmanagement.com/developer/contact/userguide/6.1-zh/index.html

[42] 赛博科技工作室，VRML 与 Java 编程技术，人民邮电出版社，2002

[43] http://www.tau.ac.il/~stoledo/taucs/

[44] Sivan Toledo，TAUCS A Library of Sparse Linear Solvers，2003

[45] Floater M. S.，Mean value coordinates，CAGD 20，1 (2003)：19-27

[46] Zayer R.，Rossl C.，Karni Z.，Seidel H.-P.，Harmonic guidance for surface deformation，In Computer Graphics Forum，Proceedings of Eurographics 2005，Eurographics, Blackwell：601-609

作者在学期间取得的学术成果

[1] 徐兴华，熊岳山，虚拟膝关节手术中的大范围实时形变方法研究[J].计算机工程，（已录用）