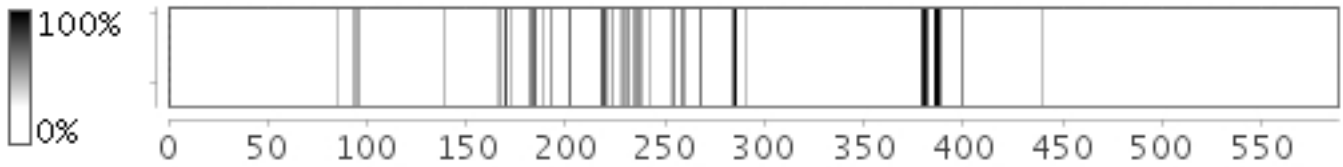
PaperPass检测报告简明打印版

比对结果（相似度）：
总　体：17 %　*（总体相似度是指本地库、互联网的综合比对结果）*
本地库：14 %　*（本地库相似度是指论文与学术期刊、学位论文、会议论文数据库的比对结果）*
互联网：4 %　*（互联网相似度是指论文与互联网资源的比对结果）*

编　号：57417512B16D61YH6
标　题：三维扫描体数据的VTK 体绘制程序设计 - 副本
作　者：蒋博洋
长　度：19349 字符(不计空格)
句子数：588句
时　间：2016-5-22 17:00:02
比对库：学术期刊、学位论文（硕博库）、会议论文、互联网资源
查真伪：http://www.paperpass.com/check

句子相似度分布图：



本地库相似资源列表（学术期刊、学位论文、会议论文）：

1. **相似度：4 % 篇名:《图像体绘制算法的分析与评价》**
   来源：学术期刊 《系统仿真学报》 2007年4期 作者: 丁庆木 张虹
2. **相似度：3 % 篇名:《体视化技术及其应用》**
   来源：学位论文 西安工业大学 2009 作者: 王预震
3. **相似度：2 % 篇名:《高动态范围体数据可视化方法研究与实现》**
   来源：学位论文 湖南大学 2010 作者: 黎佳志
4. **相似度：2 % 篇名:《体绘制技术》**
   来源：学术期刊 《计算机应用研究》 2004年10期 作者: 洪歧 张树生 王静 刘雪梅
5. **相似度：2 % 篇名:《Dividing Cubes和Shear-Warp三维重构算法研究与...》**
   来源：学位论文 哈尔滨工程大学 2008 作者: 张国庆
6. **相似度：2 % 篇名:《四种体绘制算法的分析与评价》**
   来源：学术期刊 《计算机工程与应用》 2004年16期 作者: 尹学松 张谦 吴国华 潘志庚
7. **相似度：2 % 篇名:《基于光线投射算法的医学图像三维重建研究》**
   来源：学术期刊 《仪器仪表用户》 2013年1期 作者: 汪欣 范立南 张广渊 谷文娟
8. **相似度：1 % 篇名:《医学图像可视化关键技术研究》**
   来源：学位论文 西安电子科技大学 2008 作者: 樊鹏

9. 相似度：1％ 篇名:《光线投射算法的分析与优化》
来源：学位论文 江西财经大学 2008 作者: 彭霁
10. 相似度：1％ 篇名:《面向医学断层图像的三维重建与应用技术研究》
来源：学位论文 西安理工大学 2011 作者: 曲桢
11. 相似度：1％ 篇名:《基于纹理映射的三维地震数据可视化方法研究》
来源：学位论文 南京理工大学 2008 作者: 夏冰心
12. 相似度：1％ 篇名:《三维地震数据场可视化足迹法研究》
来源：学位论文 南京理工大学 2008 作者: 李志永
13. 相似度：1％ 篇名:《基于OpenGL的医学影像三维可视化方法研究》
来源：学位论文 东北大学 2007 作者: 刘莹莹
14. 相似度：1％ 篇名:《面向太湖流域的烟雾和云的模拟》
来源：学位论文 苏州大学 2010 作者: 陈阁
15. 相似度：1％ 篇名:《基于舍弃采样的光线投射加速算法》
来源：学术期刊 《应用科技》 2008年1期 作者: 刘长征 张毅力
16. 相似度：1％ 篇名:《基于医学图像的三维非均质生物组织建模理论及方法研究》
来源：学位论文 河北工业大学 2007 作者: 郑雪虎
17. 相似度：1％ 篇名:《脑血管可视化技术研究与实现》
来源：学位论文 东北大学 2011 作者: 王刚
18. 相似度：1％ 篇名:《基于可编程图形硬件的体绘制技术研究》
来源：学位论文 西南交通大学 2007 作者: 罗艳
19. 相似度：1％ 篇名:《基于图像序列的植物维管束的三维绘制技术研究》
来源：学位论文 首都师范大学 2007 作者: 陈学峰
20. 相似度：1％ 篇名:《医学体数据三维可视化技术》
来源：学术期刊 《计算机工程与应用》 2006年18期 作者: 宋卫卫 李冠华 欧宗瑛
21. 相似度：1％ 篇名:《基于体绘制的ECT三维数据可视化研究》
来源：学位论文 哈尔滨理工大学 2009 作者: 宋伟
22. 相似度：1％ 篇名:《基于GPU的体绘制技术研究》
来源：学位论文 河南科技大学 2008 作者: 何晶
23. 相似度：1％ 篇名:《体绘制传输函数的研究与实现》
来源：学位论文 浙江工业大学 2009 作者: 谭国珍
24. 相似度：1％ 篇名:《医学断层图像快速可视化技术的研究与应用》
来源：学位论文 中南大学 2009 作者: 周敏
25. 相似度：1％ 篇名:《医学图像三维重建方法的比较研究》
来源：学术期刊 《医学信息》 2006年6期 作者: 张季 王宜杰
26. 相似度：1％ 篇名:《虚拟心脏技术中的动态图像处理》
来源：学位论文 西北大学 2005 作者: 朱洁娜
27. 相似度：1％ 篇名:《医学图像三维重建及可视化的研究》
来源：学位论文 西安电子科技大学 2009 作者: 赵奇峰
28. 相似度：1％ 篇名:《B超医学图像序列三维重建》
来源：学位论文 上海交通大学 2004 作者: 陈春涛
29. 相似度：1％ 篇名:《三维心脏超声数据的实时体绘制》
来源：学位论文 上海交通大学 2005 作者: 许健
30. 相似度：1％ 篇名:《三维标量体数据的可视化》
来源：学术期刊 《萍乡高等专科学校学报》 2011年3期 作者: 彭霁

31. 相似度：1％ 篇名:《基于地质勘探数据的三维储层模拟》
来源：学术期刊 《能源技术与管理》 2009年1期 作者: 王燕红 吴堑虹 刘勤志
32. 相似度：1％ 篇名:《基于天气雷达强度数据体绘制算法》
来源：学术期刊 《计算机与数字工程》 2013年8期 作者: 闫军 孙永刚
33. 相似度：1％ 篇名:《水声场的可视化研究与实现》
来源：会议论文 2011中国西部声学学术交流会 2011-08-01 作者: 华翔 康凤举 王定华 张森
34. 相似度：1％ 篇名:《基于反距离加权插值的水声数据可视化算法》
来源：学术期刊 《计算机工程》 2015年9期 作者: 高真 叶学义 周天琪 宋倩倩
35. 相似度：1％ 篇名:《医学超声图像三维重建绘制技术研究》
来源：学位论文 山东科技大学 2007 作者: 安新军
36. 相似度：1％ 篇名:《基于VTK和MFC的医学图像三维重建研究与实现》
来源：学术期刊 《生物医学工程学进展》 2010年1期 作者: 罗火灵 许永忠 陈世仲
37. 相似度：1％ 篇名:《基于窗位窗宽变换分类的体绘制算法》
来源：会议论文 全国第13届计算机辅助设计与图形学学术会议暨全国第16届计算机科学与技术应用学术会议 2004-08-16 作者: 王雪莉 李宗民 徐嘉丽
38. 相似度：1％ 篇名:《基于GPU的医学图像算法研究与应用》
来源：学位论文 西安电子科技大学 2011 作者: 韩慧
39. 相似度：1％ 篇名:《医学三维可视化方法以及算法平台的研究进展》
来源：学术期刊 《中国医学装备》 2007年10期 作者: 王晓敏 张艳 王鹏程 谢晋东 韩彬 唐峰
40. 相似度：1％ 篇名:《人体颈动脉血流动力学数值计算和分析》
来源：学位论文 东北大学 2012 作者: 李艳
41. 相似度：1％ 篇名:《基于足迹法的三维地震数据并行可视化方法研究》
来源：学位论文 南京理工大学 2008 作者: 朱金亮
42. 相似度：1％ 篇名:《光线投射体绘制算法关键技术研究》
来源：学位论文 天津理工大学 2006 作者: 孙薇薇
43. 相似度：1％ 篇名:《基于片段融合的医学图像光线投影体绘制技术》
来源：学术期刊 《生物医学工程研究》 2009年3期 作者: 徐玲 钱志余 陶玲
44. 相似度：1％ 篇名:《改进的三维可视化用光线投射算法》
来源：学术期刊 《中国体视学与图像分析》 2008年1期 作者: 姜慧研 李宁
45. 相似度：1％ 篇名:《改进的三维可视化用光线投射算法》
来源：会议论文 第二届立体图像技术及其应用(国际)研讨会 2007-08-01 作者: 姜慧研 李宁
46. 相似度：1％ 篇名:《基于CT图像的Ray Casting算法研究》
来源：学位论文 中国石油大学(华东) 2010 作者: 汪宏楠
47. 相似度：1％ 篇名:《基于数据场特征的直接体绘制研究与实现》
来源：学位论文 山东大学 2002 作者: 赵晓峰
48. 相似度：1％ 篇名:《基于CUDA的光线投射体绘制方法研究》
来源：学位论文 南京理工大学 2011 作者: 徐赛花
49. 相似度：1％ 篇名:《基于GPU光线投射体绘制加速算法的分析与改进》
来源：学位论文 山东科技大学 2014 作者: 朱化保
50. 相似度：1％ 篇名:《基于改进PM滤波的折射体绘制》
来源：学位论文 江西财经大学 2009 作者: 朱懿敏
……

互联网相似资源列表：

1. **相似度：4% 标题：《VTKCourse VTK教程》**
   http://www.vtkchina.org/course/index.php/author/vtkcourse/
2. **相似度：3% 标题：《VTK可视化类库及其应用进展_王敏_百度文库》**
   http://wenku.baidu.com/view/60beecc376c66137ef06191c.html
3. **相似度：1% 标题：《VTK技术在雷达图像可视化中的研究与应用_雷达_军事电子技术_技术_...》**
   http://www.81tech.com/jungong-jishu/201008/06/jishu28910.html

全文简明报告：

武汉大学本科论文

三维扫描体数据的

VTK体绘制程序设计

院（系）名称： 测绘学院

专业名称： 测绘工程

学生姓名： 蒋博洋

指导教师： 张毅 教授

二〇一六年六月

摘要

{41%：随着科学技术的不断发展，电子技术、计算机技术、网络技术在测绘领域的广泛应用，获取数据的能力和手段得到了极大地丰富。} 而面对呈爆发式增长的数据量，如何在保留数据原始细节信息的基础上更加有效的显示和处理这些数据，就成了今后研究的重点。 {41%：本文的主要内容是介绍了科学可视化的体绘制方法，分析了其相比于传统面元绘制方法的优势，然后介绍了几种不同的体绘制算法和其特点。} 然后介绍了VTK可视化软件包的相关结构和功能，并使用VTK编写了体绘制实现的程序，最后分析了获得的结果图像。

关键词： 体绘制； 科学可视化； 三维体数据

ABSTRACT

With the continuous development of science and technology, electronic technology, computer technology, network technology is widely used in the field of surveying and mapping, data acquisition capability and means has been greatly enriched. To deal with those amount of data became more and more significant, how to be more effective in retaining the original details of the data on the display and processing of these data, it would be the focus of future research. The main content of this paper is to introduce the scientific visualization volume rendering method, analyzes its advantages compared to the traditional method of drawing bins, and then introduces several different volume rendering algorithm and its

characteristics. Then introduced the VTK visualization related structure and function packages, and volume rendering using VTK write a program to be realized, the final analysis result of the image obtained.

Keywords: Volume data； Volume Render； scientific visualization

目 录

(各章的名称黑体4号，其余宋体小4)

(结论、参考文献、致谢及附录黑体4号)

1 绪论

1.1 体绘制研究现状和发展趋势

{50％：科学数据可视化技术就是运用计算机的图形学、图像处理和视觉算法等方法，用来直观的把科学研究、工程设计、医学图像中的各种数据转换为图像的技术。} {41％：其本质是对数据的处理和交互，使得其更加容易理解和应用。} 如今科学数据可视化技术的研究和应用取得了飞速的发展，一方面是由于其能够将大量抽象的数据用图形图像更加直观的表现的功能； {57％：另一方面则是因为科学数据可视化的应用领域十分广泛，几乎适用于科学研究、工程技术、医学影像等等一切领域。} 在科学数据可视化中，最重要的莫过于三维数据的可视化，于是体绘制作为科学数据可视化中的一个重要组成，自然得到了高度重视。

对于体绘制的研究最早开始于70年代中期，当时医学领域CT、核磁共振和超声波等成像技术开始应用于临床领域， 为了满足临床医学对于三维影像显示的精度要求，体绘制技术随之产生。 {49％：它使用图像处理、图形学和计算机视觉等技术，能够借助三维体素将三维的离散点集数据用直观的二维图像显示在屏幕上，} {49％：从而帮助研究人员更加直观的理解和处理各类数据。} {44％：体绘制的优点是在生成高质量的图像的同时，还能够清晰显示物体的内部结构、特征信息和局部细节，这是基于面元绘制的传统计算机图形绘制方法所做不到的。} {54％：在80年代末期，在Levoy和Drebin等人又提出了直接视觉算法，之后对于体绘制的相关研究不断增多，} {49％：而且应用范围也不再仅限于医学影像而是扩展到了生物学、物理学、地理学、工业应用、工程技术和气象等更加广泛的领域。}

体绘制技术诞生至今，在国内外各个领域科研人员努力下，已经提出了各类针对不同需求下的体绘制算法用于分析、理解和绘制三维数据场， 这些三维应用已经涉及土木工程、产品设计、医学影像、地球科学等各种领域。{60％：经过近几年的飞速发展，体绘制在软硬件和算法上都取得了长足的进步，然而作为新兴学科，仍然有很多方面都亟待提高和完善，这些也是今后这个领域研究的重点。}

今后主要的研究方向包括：

（1）体绘制的算法和数据结构的改进依然是一个重要的研究方向，对于密集的三维体数据点集的体绘制过程， 即便是在算法和数据结构上的一点小小改进也能显著加快程序的运行数度。

（2）如今的计算机运行速度和网络技术都得到了极大的提高，如何将体绘制技术与网络技术结合， 提供一

个更加高效的分析计算环境，发展更多有效的并行算法，这也是研究人员正在关注的问题。

（3）正如早期的图形硬件技术能够加快图形处理的速度一样，如何在现今的图形硬件技术下增加针对体绘制的图形绘制硬件设备，也是可视化硬件需要得到解决的。

1.2 课题研究来源和意义

测绘学科诞生至今，数据获取的手段得到了极大的丰富，借助于电子技术和计算机技术的发展，获取数据的方式从最开始的水准仪、经纬仪量测， 到全站仪、航空摄影、遥感、GPS等诸多技术的出现，再到如今基于互联网的数据网络系统，获取数据的能力得到了极大的提升， 获得的数据量也呈爆发式增长。 而如何根据各种不同的需求有效的展示这些获取得来的数据，也成为了今后研究的重点。 以地形数据为例，最开始采用的最多的还是纸质地图，但是纸质地图所展示的内容有限， 于是诞生了更加方便的电子地图，到了互联网时代，为了适应对于各种数据使用的需求，各种地图软件层出不穷， {47%：对于这些数据的使用也融入了人们的日常生活之中，极大的促进了社会的发展。} 随着对于数据展示方式的需求加剧，传统的数据展示方法已经不能够满足对于更高精度的数据的分析需求， 研究人员为此做出了各种有意义的研究，其中以科学可视化的体绘制方法为代表， {46%：更高精度的数据展示方法开始越来越得到人们的重视。}

对于三维体数据，传统的数据显示方法是采用基于面元的绘制方法，这种方式借助于中间面元对于三维点集数据的拟合， {45%：通过得到的多边形面元来实现数据的三维重建。} 这种方法算法简单，对于内存和处理器效率的要求不高，能够满足基本的三维显示需求， 但是随着CT、核磁共振仪等成像技术的发展，传统的面元绘制方式已经不能够满足高精度的三维数据显示的要求， {47%：于是基于科学可视化的体绘制方法开始得到越来越多的关注。} 体绘制方法不借助中间面元而是直接使用三维点集数据进行三维重建，这样能够最大限度的保留体数据中的细节信息和数据的各种属性特性， 从而生成更高精度的图像，为更高需求的数据显示提供了可能。

测绘学科中获取到的数据中，三维数据占了绝大部分，但是针对于如何有效的在保留高精度的前提下处理和展示这些数据，目前还有很多问题需要解决。 针对于这些获取到的三维数据，如何采用合适的显示方法，如何处理数据误差，如何分析数据之间的关系，如何提取数据之中的有效信息，这正是很多研究人员在努力解决的方向。本文就如何有效的克服传统对于三维数据显示方法的不足，介绍了科学可视化下的体绘制方法的相关内容，并就VTK下的体绘制程序实现的相关内容进行了研究。

1.3 研究的主要内容

本文是原理结合实践，以数据可视化中对三维数据的体绘制方法为研究对象，主要研究使用VTK在三维点集数据的体绘制中的实际应用。 {40%：主要研究体绘制的主要算法，以及使用VTK进行体绘制程序编程时用到的主要功能的实现方法。}

具体的研究内容为：

（1）传统三维数据绘制方法采用的是首先借助中间等值面拟合三维点集数据，再用拟合的等值面构建的多边形展现三维视觉的方式。 这种方式不仅算法简单，而且只需要存储中间面元数据，避免了对点集数据大量读写和处理，节约了内存空间和运算效率， 但也存在丢失大量细节数据、对体数据场的属性表现不够等种种不足。 {41%：本文就传统三维数据绘制方法相比体绘制方法的不足进行了分析。}

（2）体绘制自诞生之初发展至今，已经针对各种不同的需求产生了几种不同的算法思想， 例如按照视觉光

线重建三维数据的光线投射法、直接使用体数据点集进行贡献率演算的抛雪球法和使用投影变换的错切-变形法，以及使用硬件设备进行优化的三维纹理映射法。 {42%：本文将会对每个算法的原理和之间的区别进行介绍。}

{45%：（3）VTK是由美国 Kitware公司开发的用于进行图像数据处理和显示的可视化软件包，} {44%：最早应用于医学领域，用于处理 CT、核磁共振仪等仪器获得的三维数据的可视化。} 后来随着其他学科对于三维数据可视化需求的逐步增加，VTK也开始在诸如建筑、土木工程、质量检测、产品设计和气象等领域得到应用。 经过十几年的发展，VTK库的设计已经十分成熟，其中也添加了各类常用的体绘制算法，能够十分方便的编写各类体绘制程序。 本文将就 VTK的主要架构，VTK主要的功能和数据流结构，以及 VTK编程的环境配置等内容进行介绍， {42%：并使用 VTK编写程序实现体绘制的基本功能，并对得到的结果进行分析。}

1.4 论文的主要结构

{45%：本文是对科学可视化中的体绘制算法，以及使用VTK软件库进行体绘制程序实现方法的研究。} {45%：主要工作是分析了传统面元绘制方法相比体绘制方式的不足和各类体绘制算法的特点，} 介绍了 VTK软件库的程序架构和主要功能，其各种模块的相应作用，以及其数据流在可视化管线以及渲染引擎中的传递关系， {41%：最后使用最新的 VTK6.3.0版本进行编程环境的搭建和体绘制程序的编写，} 并对得到的结果进行了分析。

本论文大体模块分为五章：

{50%：第一章为绪论，介绍了本课题的国内外研究现状，课题的来源与意义和主要研究内容等。}

第二章对传统面元绘制方法的不足进行了分析，并对体绘制四种主要算法原理进行了介绍，并讨论了这几种方法的区别。

第三章是对VTK软件库的介绍，介绍了其架构和主要功能，其各个模块的作用和数据读写的特点，最后借助一个简单的VTK程序介绍了其体绘制程序代码的结构。

第四章是使用 VTK进行体绘制程序编写的实践，使用 CMAKE进行了最新版本的 VTK编程环境的搭建，编写 Python程序实现了体绘制程序的基本功能， 并对最后使用程序对三维数据进行处理后的成果图像进行了分析。

{44%：第五章是结语部分，对全文的内容进行了整理和总结，提出了一些展望。}

2 体绘制原理

2.1传统绘制方法的不足

传统的三维数据可视化过程是借助中间图元来完成的，首先提取三维体数据的等值面曲面作为二维图元， 再将提取出来的二维图元连接成多边形最终渲染到屏幕上。 这种方法虽然具有简单、易实现、图像质量较高的优点，但是也具有以下等不足：

（1）、传统的面绘制技术实际是采用大量三角形面片拟合三维数据中提取出的等值曲面， 因此如果部分细节部分的变化小于所确定的阈值就会被等值面忽略掉， 这样会丢失微小面元的准确位置和方向信息。

{47%：（2）、在生成三角面片时，对于多个顶点的连接方式并不唯一，如果这种情况出现在相邻两个单元的共享面上时，就可能产生实际不存在的空洞。}

{44%：（3）、对于大量密集的三维体数据，在进行面绘制时会产生大量三角形面片，} {60%：对于一个单元的等值面，最多可使用四个三角形来拟合，例如从一个典型的256×256×113高分辨体数据场中构造的一个3D面，} 要求718964个三角形，并且这些三角形要存储需要25 MB内存(每个三角形要求36个 By tes); {96%：另外，大量的三角形意味着需要更多的时间来绘制。}

{54%：（4）、传统面元绘制算法将等值面上的样本点等值于等值面中的点，如果三角形退化成一条直线时会有冗杂多边形产生，} 这些冗杂多边形对于最后的结果图像没有作用而且还占用内存。

{62%：与传统的中间面元绘制方法相比，体绘制直接使用来自一个物体表面和内部的数据，而不是仅仅限制于阈值确定的等值面上的数据，} 因此体绘制能够显示三维体数据中各种局部细节的信息，而且成图原理更加接近人眼的成像原理， {61%：在体数据处理和特征信息表现方面上具有更多的优势。}

2.2 体绘制几种算法介绍

体绘制的思想是不借助中间面元而是直接使用三维点集数据，并赋予这些点集数据以不同的属性值，最后根据得到的属性值生成最终图像。 在这个过程中对每一个体素都进行了至少一次的遍历，保证了对所有数据中有效信息的提取， 而且尽可能的保留了其中体素之间的联系关系，使得最终图像能够尽可能的对数据中的细节信息和数据场属性进行描述。 根据这个思想研究人员提出了各种体绘制算法，这些算法的主要内容就是如何尽可能的提取点集中的有效信息，并最大限度的提升算法的效率和最终图像的质量。

2.2.1 光线投射法

{57%：光线投影法（Ray Casting）是体绘制中最早提出来的算法之一，这种算法以图像空间为基准，} {52%：从每一像素出发，向观察的视角点也就是相机的位置发出一条射线穿过三维体数据点集，} {74%：沿着这条射线选择 K个等距离的采样点，每个采样点的颜色值和不透明度值由距离这个采样点最近的8个数据点做三线性差值得到。} {72%：求得这 K个采样点的颜色值和不透明度值之后，再将每条射线上各个采样点的颜色值和不透明度值按由前向后或者由后向前的顺序加以合成，} {60%：最终得到发出该射线的像素的颜色值，并最终在屏幕上显示出来。}

图2.1.1 光线投射法

{58%：光线投射法考虑了数据场中所有体素对生成图像的贡献，尽可能的利用了原始信息，因而产生的图像更加真实并且具有较高的质量。} {51%：但是这种算法需要遍历每一个体素，当观察方向改变时，每个像素发出的射线经过的点集也会发生变化，这样就需要重新采样，计算量庞大而且占用内存。} 针对这种情况，人们也提出了相应的解决办法。

（1）、光线提前终止

{55%：光线提前终止能够有效提高体绘制的效率，它按照从前到后的顺序跟踪每个像素发出的射线，} {57%：射线穿过每个体素时，这条光线的阻光度就随之累计，一旦阻光度到达设定的阈值，} {49%：射线即停止传播，这样就屏蔽掉了之后的体素，减少了采样的区域从而减少了计算量。}

（2）、空间数据结构

{57％：在给定的体数据集中，一些体素能够提供有用的信息，而另外的一些则不能提供。} 空间数据结构的作用就是将相关的一致的有用数据编码，更加有效的精选出需要的数据。 这些空间数据包括： 八叉树、k-d树、金字塔结构等。 {50％：这些数据结构研究数据内部的相关性，允许跳过不重要的体素或者相对均匀的区域，从而减少计算量。}

2.2.2 抛雪球法

{75％：抛雪球法(Splatting)跟光线投射法不同，它是反复对体素进行运算，使用一个称为足迹(Footprint)的函数计算每个体素投影的影响范围，} {65％：然后用高斯函数定义强度分布（中间大周围小），计算出最终其对图像的总体贡献，} 合成得到最终图像。 这种方法把每个体素看作能量源，实际是计算了每个体素的能量由中心向四周扩散的状态，能量的传播随着距离而减少，最终的图像由这些能量生成。 {46％：除了一般体绘制的特点，这种方法还具有以下的特点：}

图2.2.2 抛雪球法

（1）、速度快。 {48％：相比光线投射法，抛雪球法的优势之一就是速度快。} 光线投影需要计算每条射线沿线的采样点，每个采样点又需要进行三次卷积，这样时间复杂度就大大提高了。 {45％：另一方面，抛雪球法中，卷积则是被预先计算了，每个体素只运算一次，这样计算量就被减少了，并且算法适合并行计算。}

（2）、渐进细化。 {61％：因为抛雪球法按照严格的顺序生成图像，这样就能够在观察图像的逐步生成的过程中观察到数据场的内部信息，而这在图像空间为序的技术中是做不到的。} 因此这种方法能够看见后面被隐藏的物体结构。

{47％：但是这种方法也有其不足，理论上使用同样的重构函数权值，这种算法生成的图像的质量应该是能够达到光线投射法同样质量的。} 但在实际中，权值计算比较困难，因此常常使用近似计算，造成图像质量的下降。

2.2.3 错切—变形法

{46％：平行投影的错切-变形法（Shear-wrap）基本的思想包括三个步骤：}

{87％：（1）、将体数据变换到错切后的物体空间，对每一层数据重采样}

{75％：（2）、按照从前往后的顺序将体数据投射到二维中间图像平面}

{85％：（3）、通过变形变换，将中间图像投射到像空间产生最终图像}

图2.2.3 错切—变形法

这是种多次重复采样的体绘制方法，不过不同的是只需要两次重采样，少于其他算法所需要的三次或更多的重采样； {52％：其中第二次重采样是二维的弯曲变换，只遍历了一次体数据，这大大降低了计算量。} {47％：

不过这种算法的不足就是特定视角下出现阶梯状走样，图像放大之后会变得模糊等。｝

2.2.4基于硬件的三维纹理映射法

｛46％：体绘制目前的问题在于算法的速度，为了生成图像需要对每个体数据进行计算，并且体素化之后的重定位体素又会消耗大量的计算。｝｛40％：针对这个问题的解决办法就是各种改进的算法，也就是上面提到的各类算法。｝｛43％：这些算法都是基于软件基础上来提升算法速度的，而三维纹理映射法(Hardware-assisted 3D texture-mapping)就是基于硬件上的体绘制算法。｝｛62％：其中所谓的基于硬件，指的是将纹理空间中的重采样的插值运算和不透明度图像合成等一系列操作交给硬件来完成，从而极大的提高了运算速度。｝｛46％：这种算法的流程是，首先将体数据作为三维纹理图装入纹理内存，之后再体数据内部定义一系列采样多边形采取物体纹理，｝｛78％：最后通过查找表将采样得到的数据转换成相应颜色值和不透明度，于是就可以按照从后往前的顺序进行图像合成，｝ 投影到视平面生成最终的图像。

｛63％：为了使生成的图像显示出明暗的立体效果，提升图像质量，又提出了一种基于三维映射纹理硬件支持的带有敏感效果的提绘制方法。｝｛53％：首先将密度数据和梯度数据装入纹理内存；｝｛62％：然后用纹理硬件沿着光线投射方向进行重采样，将得到的重采样数据存储进内存；｝｛57％：最后在CPU中对这些数据进行明暗处理和合成，得到每个像素的RGB值，放入缓存中，使用硬件显示出图像。｝｛55％：这种方式的优点是用随着视角变化的光线对每个数据进行明暗处理，因此能够得到高质量的图像。｝

然而基于硬件的三维纹理映射算法也有其不足，这是因为计算机的纹理硬件有限， 大规模进行体绘制时只能够将所有数据分块装入内存，而这会导致频繁的输入和输出， 再加上重采样的运算从而使得整个体绘制程序性能的大幅下降。 ｛50％：这个问题的解决办法是利用空间跳跃技术来有效降低纹理内存的数据交换量，从而提升程序的绘制速度。｝｛40％：另一个方案是采用四级体数据导入流水线，在绘制前对数据进行有效的预处理；｝ 并且因为使用了索引结构加速分类，选择对最终图像有用的体素装入纹理内存，从而降低内存占用，提升了绘制速度。

2.3 四种体绘制算法的比较

2.3.1 光线投射法

光线投射法主要的不足就是存取体数据的方式，因为是按照视线方向来提取数据的，而这个方向会根据视角的变化发生改变， 导致视线跟体素可以从任何方向相交，所以这种算法不能按照体数据存储的物理顺序读取数据。 一旦视线发生改变，都可能会导致所需的体素发生改变，就需要重新计算采样点的位置，以及其属性值。 这所导致的另一个问题就是会导致数据在内存中频繁的读写，由于视角变化的任意性，体数据的空间结构就无法发挥其优势。

针对这个问题的改进是对于光线提前终止条件的改进，以及各类优化的空间数据结构，这些都能够有效减少计算量，提升体绘制程序的运行效率。

2.3.2 抛雪球算法

｛53％：抛雪球算法的特点是按照基于体数据的空间存储顺序来读取和存储对象，而且只使用与最终图像相关的体素，从而大大减少计算量。｝｛66％：此种算法适合并行操作，但随着视角方向发生改变，就需要重新计算重构函数卷积域在平面上的投影面积，｝｛57％：然后重新对每个对应的体素进行选择以及比例变换，所以计算

量也相当大。｝

2.3.3 错切-变形法

｛41％：错切-变形法分两个步骤，首先将离散的三维体数据场投影变换进行剪切变换，然后再对二维图形进行变形，｝｛69％：于是三维空间的重采样被转换成了二维平面上的重采样，这降低了计算量。｝｛48％：这种算法的优点是三维数据的体绘制可以以接近于实时渲染的速度在图形工作站上进行实现，而且不会损失最终图像的质量。｝

｛53％：这种算法的局限在于，三维数据在进行错切空间变换是，观察的视角方向必须跟三维坐标系的某一轴重合，否则就不能发挥其优势。｝

2.3.4 基于硬件的三维纹理映射法

｛54％：基于硬件的三维纹理映射法是利用图形硬件的纹理内存，在纹理空间中进行体数据重采样的插值运算，以及不透明度的合成等操作，使用硬件来提升运算速度。｝ 特别是是面对表面细节复杂的数据时，这种方法更具有其明显的优势。 但是这种算法会受到所使用的硬件的限制，要得到高质量的图像，进行体绘制时所用到的图形硬件就必须满足一定的要求。

视化工具包VTK

3.1 VTK介绍

｛73％：VTK(Visualization Toolkit)由美国Kitware公司开发，其特点是开源代码、并且面向对象式的可视化工具包。｝｛64％：VTK将许多常用的图形处理和模型生成算法封装，有很多可以直接使用的类和函数，并且是基于OpenGL开发而成。｝ VTK的使用十分方便，因为大多数经常使用的细节算法部分都被封装进类中从而使得开发者不需要了解其中各个算法实现的细节， 而可以专注于解决实际需要解决的问题上，能够方便的对各类型的图像数据进行变换个操作。

｛51％：VTK的核心部分采用 C++语言开发，也能够使用 Java、 Python等各类语言环境，可以在各类程序之间进行代码的转换，｝ 支持使用 MFC、 Qt进行 GUI界面的编程，这样能够满足各种工程的需要。 同时VTK支持跨平台，Windows和Unix系统上都能够轻松胜任，满足不同开发需求。 ｛56％：VTK代码是完全开源的，所以程序开发人员能够根据项目需要在VTK原码的基础上修改已有的组件或者是开发新的类库。｝ VTK具有较高的算法执行效率以及高度的灵活性，能够在生成高质量图像的同时保持较高的速度， 这使得很多数据可视化的从业研究人员都开始着手进行 VTK的使用和研究。

3.2 VTK程序架构

3.2.1 VTK的主要架构

｛62％：VTK包括两类的对象模型，分别是构成所有图形的基础图形模型，以及可视化过程中用到的各类数据流模型。｝ 其中图形模型的功能主要是构成图形系统，并实现图像生成以及用户交互等功能，包括9个基本的对象： vtkRenderer渲染器进行坐标定位， vtkRenderWindowInteractor实现窗口的用户交互， vtkRenderWindow用于管理显示设备窗口， vtkProperty用于设置 vtkActor的表面属性， ｛70％： vtkMapper表示实体的几何形状， vtkLight灯

光、 vtkCamera照相机、 vtkActor演员分别用来设置场景的光照度、视角焦点和实体， } vtkTransform定义vtkActor、 vtkCamera和 vtkLight的位置和方向。

{ 68 %： VTK的数据流模型包括两类基本的对象，即数据对象以及处理对象。 } { 98 %： 数据对象代表了进入可视化网络的数据集类型，包含体数据(vtkImageData)等五种类型； } { 100 %： 处理对象按功能分为数据源(Source)、过滤器(Filter)和映射器(Mapper)三种。 } 其中Source指用于创建数据（例如vtkCylinderSource）或者读取数据（例如vtkMetaImageReader）的类的统称，是VTK数据流中的数据源。 { 46 %： Filter负责对读取的数据进行各类处理，得到新的数据； } 得到的新数据或直接写入文件，或是经Mapper变换之后倒入渲染引擎中进行渲染显示。

3.2.2 可视化管线

{ 65 %： 可视化管线是指用来获取或者创建数据、处理数据然后将数据写入文件或者传入渲染引擎进行显示的这样一种结构， } 包括数据对象（ Data Object）、处理对象（ Process Object）和数据流方向（ Direction of Data Flow）。 要进行数据的处理就必须用到可视化管线，可视化管线可以看做VTK中图像数据处理的数据流结构。

图3.2.2 VTK程序的可视化管线及渲染引擎

3.2.3 VTK的体绘制可视化管线

三维数据的体绘制过程跟二维图像的几何图形数据的可视化不同，主要有以下两点的区别：

几何绘制中通常使用vtkActor来渲染几何图形数据，使用vtkImageActor渲染图像数据； 在体绘制中，则是使用vtkVolume来对体数据进行渲染。

几何渲染中采用vtkPolyDataMapper实现输入数据向图元数据的转换； 体绘制管线中则是使用vtkVolumeRayCastMapper，因为体绘制需要使用各种体绘制算法，需要使用不同的Mapper类。

图3.2.3 VTK体绘制程序的可视化管线

3.3 VTK的数据读写

3.3.1 VTK的数据读写结构

{ 54 %： VTK应用程序所需的数据可以通过两种途径获取： } 第一种是VTK自己生成的模型，这些模型数据由VTK自建的类生成（如vtkCylinderSource生成的多边形数据）； 第二种是外部读取导入的相关数据文件（如vtkBMPReader读取的BMP格式图像）。

VTK处理完成的数据可以由VTK自建的类写入单个文件，也可以是将渲染的场景导出，以备之后的后续处理工作。 从可视化管线的角度来看，VTK的数据流以数据的读取或模型生成为起点，以数据的导出或Mapper导入渲染引擎为终点。

3.3.2 VTK的数据读写主要步骤

要将外部数据读取导入可视化管线，主要用到的是VTK的Reader类，主要的步骤如下：

1）实例化Reader对象。

2）指定需要读取的文件名。

3）调用Update（）方法更新数据流，促使管线的执行。

同样的，当需要导出处理后的数据，使用Writer类的主要步骤如下：

1）实例化Writer对象。

2）输入需要导出的数据以及导出数据的文件名。

3）调用Writer（）方法促使VTK开始导出操作。

3.3.3 VTK的数据读写的具体类介绍

VTK针对各类数据格式提供不同的Reader/Writer类，本文主要就使用的MHD（*.mhd）格式文件的读写操作进行详细介绍。

VTK中对于MHD格式的读写主要是用使用自建类型的vtkMetaImageReader/vtkImageWriter类读取。 MHD格式包括两个文件，分为图像信息头文件和实际图像文件，实际图像文件即RAW格式的图像（*.raw），MHD文件则主要是记录实际图像文件的相关信息。 以本文所使用的backpack8.mhd文件为例，包含以下的内容：

NDims = 3

DimSize = 512 512 373

ElementSize = 0.9766e+000 0.9766e+000 1.25e+000

ElementSpacing = 0.9766e+000 0.9766e+000 1.25e+000

ElementType = MET_UCHAR

ElementByteOrderMSB = False

ElementDataFile = backpack8.raw

其中各个标签的意义如下：

NDims ： 表示图像的维度。

DimSize ： 表示各个维度的大小。

ElementSize ： 表示各个像素的大小。

ElementSpacing ： 表示像素间的间距大小。

ElementType ： 存储数据所用的像素值的数据类型，MET_UCHAR指所使用的是unsigned char，也就是1个字节以无符号字符的方式存一个像素。

ElementByteOrderMSB ： 表示字节存储的顺序。

ElementDataFile ： {44％：存储图像数据的文件位置，一般都是跟MHD文件同名的RAW文件，并且放在同一个目录下。}

3.3.4 图像数据的类型转换

在处理数据时有时需要用到数据的类型转换，例如常用的图像算子（例如梯度算子等）在计算后为了精度的考虑， 会将最后的结果存储为 float或者 double类型，但是图像的显示最后一般是 unsigned char类型， {59％：这种时候就需要进行数据的类型转换。} VTK中最简单也最常用的类型转换工具类就是vtkImageCast，主要的用法如下：

```
vtkSmartPointer[vtkImageCast] imageCast =

vtkSmartPointer[vtkImageCast]:  New();

imageCast-]SetInput((vtkDataObject *)reader-]GetOutput());

imageCast-]SetOutputScalarTypeToFloat();

imageCast-]Update();
```

使用的时候只需要将SetOutputScalarTypeToXXXX()设置成相应的类型即可。 {43％：图像数据的类型转换一般都紧接在在读取数据之后。}

3.4 一个简单的VTK工程

```
1:  #include ” vtkRenderWindow.h ”

2:  #include ” vtkSmartPointer.h ”

3:  int main()

4:  {

5:  vtkSmartPointer[vtkRenderWindow] renWin =vtkSmartPointer[vtkRenderWindow]:  New();
```

6: renWin-]Render();

7:

8: std: cin.get();

9: return 0;

10: }

{100%：第1、2行，包含头文件，因为要用到VTK里的vtkRenderWindow和vtkSmartPointer两个类，所以包含相应的头文件。} {100%：VTK对类的命名都是以小写的vtk开头，每个类的关键字的首字母大写。}

{100%：第5行，用智能指针定义了一个类型为vtkRenderWindow的对象，这是VTK的类实例化对象的基本方法。} {51%：因为VTK里每个类的构造函数都定义为保护成员，要构造VTK的对象可以用第5行的方法，或者用以下的方法：}

vtkRenderWindow*renWin = vtkRenderWindow: New();

{100%：第6行，调用vtkRenderWindow里的方法显示并渲染VTK窗口。}

{97%：第8行，没有其他特别的意义，只不过是让程序暂停下来，等待接受用户的输入。}

{100%：这个程序非常简单，就一个VTK窗口，其他什么也没有。} {100%：但它确实是一个VTK的工程，至少使用了两个VTK类，调用了一个VTK的方法。}

VTK下体绘制实践

4.1 环境配置

4.1.1 获取VTK源码

VTK是开放源码的，因此我们可以从VTK官方网站（http: //www.vtk.org/）下载源代码。 本文所使用的版本是VTK 6.3.0，以下的内容均以此版本作为标准。 所需要下载的是 VTK的源码 VTK-6.3.0. zip， 除此之外可选的下载项有 VTKData-6.3.0. zip（VTK自带的示例以及测试程序运行时所 需要的数据）、 vtkpython-6.3.0- Windows-32 bit. exe（VTK编程所使用的 Python接口）、 vtkDocHtml-6.3.0. tar. gz（VTK的文档文件， {83%：可以查看VTK各个类以及程序接口详细的使用介绍）。}

4.1.2 编译VTK的准备工作

CMAKE是一个开源的跨平台配置程序，它使用 CMAKE的配置文件对代码进行管理，能够根据平台的不同和各种设置生成相应的工程文件， 相比于 Unix的 Make和 Windows的 Makefile， CMAKE采用更加简单易懂图形界面，这样即便是初学者也能很方便的使用 CMAKE进行程序的配置。 使用的时候需要使用一种建构软件专用的特殊

编程语言写的 CMake脚本，自带 C语言、 C++、 Fortran、 Java的自动相依性分析功能， 经由 CMake脚本语言支持 SWIG、 Qt、 FLTK，并且支持微软 Visual Studio. NET和过去的Visual Studio版本，可以根据需要产生相应的.dsp、.sln和.vcproj文件。

VTK是使用CMake来管理工程的，在编译VTK之前需要先下载并安装CMake（https: //cmake.org/）。 本文所使用到的CMAKE是3.5.2版本的。

4.1.3 编译VTK的详细步骤

第一步、使用CMAKE对下载的代码进行编译

首先将下载下来的代码解压到某个目录，为了使后面的工作便于区分，可以新建两个文件夹 VTK6.3.0和 VTK6.3.0_ bin， 其中前一个文件夹放解压的代码，后面一个放编译后的工程文件，如果下载了示例跟文档的也放在第一个文件夹里面。

打开 CMAKE，在最上面的路径中分别输入 VTK6.3.0和 VTK6.3.0_ bin两个文件夹的路径， {55 %： 接着点击左下的“ Configure ”按钮进行编译的配置，在弹出的对话框选择所使用的编译器， } 这里选择的是 VS2010，可以根据自己安装的版本进行选择。

第二步、代码编译的配置

等配置好之后的界面如图，接下来就是对于建立的编译工程的配置选项的说明：

BUILD_EXAMPLES： 默认值是False，如果打开就会编译下载的示例，如果想要学习VTK编程的例子的话，建议打开这个选项。

BUILD_SHARED_LIBS: 默认关闭，这个时候VTK是静态编译的； 如果打开的话则是变成动态编译。

VTK_DATA_ROOT： {58 %： VTKData-6.3.0.zip解压之后所在的路径，一般都可以自动检测到。 }

CMAKE_INSTALL_PREFIX： VTK的安装路径，默认C: /Program Files/VTK。

VTK_USE_QT： VTK支持Qt的接口，如果需要使用Qt开发的话可以选上。

VTK_WRAP_JAVA、VTK_WRAP_PYTHON、VTK_WRAP_TCL： 分别是Java、Python、TCL的语言开发接口，如果需要可以选上。

图4.1.3 配置完成之后的CMAKE界面

再选择右上角的Advance，就可以看到更多的选项，这里面有几个需要说明：

BUILD_DOCUMENTATION： {57 %： 默认关闭，选择的话则会编译下载的vtkDocHtml-6.3.0.tar.gz帮助文档。 }

VTK_USE_GUISUPPORT： 是否让VTK支持GUI编程，如果需要使用VS的MFC编程的话需要选上。

第三步、CMAKE和VS编译

确认了每个配置选项之后，再次点击“ Configure ”继续配置，点击之后出现的红色项目表示新增加的部分，{64%：等到点击“ Configure ”之后不再出现红色选项之后就可以点击“ Generate ”按钮开始 CMAKE的编译生成VS工程文件。}

CMAKE编译完成之后能够在之前的VTK6.3.0_bin文件夹内找到编译好的VS2010工程文件，接着打开目录下的vtk.sln文件进入VS2010内。

在VS内的解决方案管理器内单击右键，在右键菜单内选择“生成解决方案”，之后会自动编译VTK文件，时间大概十几分钟到半个小时不等。

编译完成之后会在VTK6.3.0_bin目录下的Debug文件夹内生成编译完成的文件，如果是动态编译的话则会生成对应的dll文件。 到这里VTK的工程编译就完成了。

4.2 程序运行流程

4.2.1主程序程序Python代码

```
def vtkVolumeRender(fname):

##################################################################

#读取，使用的是vtkMetaImageReader类读取MHD格式

file_name = fname

reader = vtkMetaImageReader()

reader.SetFileName(file_name)

#vtkImageCast进行数据类型转换，这里转换成unsignedShort

cast = vtkImageCast()

cast.SetInputConnection(reader.GetOutputPort())

cast.SetOutputScalarTypeToUnsignedShort()

cast.Update()

output = cast.GetOutputPort()
```

```
#########################################################################
#定义体绘制算法函数
rayCastFun = vtkVolumeRayCastCompositeFunction() #光线投射法
##rayCastFun = vtkVolumeRayCastMIPFunction() #最大密度法
##rayCastFun = vtkVolumeRayCastIsosurfaceFunction() #特定等值面法
##rayCastFun.SetIsoValue(100) #特定等值面的数值
#设置体绘制的Mapper，有两个输入
volumeMapper = vtkVolumeRayCastMapper()
volumeMapper.SetInputConnection(output) #第一个是输入图像数据
volumeMapper.SetVolumeRayCastFunction(rayCastFun) #另一个是设置体绘制的光线投射函数
#######接下来设置体绘制的各种属性#################################
#设置光线采样距离
volumeMapper.SetSampleDistance(volumeMapper.GetSampleDistance()*4)
#设置图像采样步长
volumeMapper.SetAutoAdjustSampleDistances(0)
volumeMapper.SetImageSampleDistance(4)
#体绘制属性设置
volumeProperty = vtkVolumeProperty()
volumeProperty.SetInterpolationTypeToLinear()
##volumeProperty.ShadeOn() #打开或者关闭阴影测试
volumeProperty.SetAmbient(0.4)
volumeProperty.SetDiffuse(0.6)
```

```
volumeProperty.SetSpecular(0.2)
#灰色不透明函数
compositeOpacity = vtkPiecewiseFunction()
compositeOpacity.AddPoint(70,  0.00)
compositeOpacity.AddPoint(90,  0.40)
compositeOpacity.AddPoint(180,  0.60)
volumeProperty.SetScalarOpacity(compositeOpacity) #灰色不透明函数导入体绘制属性
#颜色传输函数
color = vtkColorTransferFunction()
color.AddRGBPoint(0.000,  0.00,  0.00,  0.00)
color.AddRGBPoint(50.00,  1.00,  0.00,  0.00)
color.AddRGBPoint(100.0,  0.00,  1.00,  0.00)
color.AddRGBPoint(150.0,  0.00,  0.00,  1.00)
color.AddRGBPoint(200.0,  1.00,  1.00,  1.00)
volumeProperty.SetColor(color) #导入颜色函数
#梯度不透明函数
volumeGradientOpacity = vtkPiecewiseFunction()
volumeGradientOpacity.AddPoint(10,  0.0)
volumeGradientOpacity.AddPoint(90,  0.5)
volumeGradientOpacity.AddPoint(110,  1.0)
##volumeProperty.SetGradientOpacity(volumeGradientOpacity)#导入梯度不透明效果
#vtkVolume类型，相似于vtkActor，接受两个输入
```

```
volume = vtkVolume()

volume.SetMapper(volumeMapper) #设置Mapper对象

volume.SetProperty(volumeProperty) #设置属性对象

############################渲染引擎设置###################

ren = vtkRenderer()

ren.SetBackground(1.0,  1.0,  1.0)

ren.AddVolume(volume)

renWin = vtkRenderWindow()

renWin.AddRenderer(ren)

renWin.Render()

renWin.SetWindowName( " VolumeRenderingApp " )

iren = vtkRenderWindowInteractor()

iren.SetRenderWindow(renWin)

ren.ResetCamera()

renWin.Render()

iren.Start()
```

4.2.2 程序流程图

图4.2.2 程序流程图

4.3 结果分析

4.3.1 实验数据说明

本次试验使用的是由 University of Erlangen的 The Volume Library在其官方网页上提供的 CT扫描数据，所扫描的物体是一个小型的盆栽，　数据的空间大小为256 x256 x256，数据格式是. raw文件，存储类型8 bit unsigned char。根据网页提供的数据说明编写了相应的.mhd文件，实际程序运行时读取MHD文件就可以开始整个体绘制程序的绘

制过程。

4.3.2 程序功能说明

{46%：本程序采用 VTK可视化软件包编写体绘制程序，实现了光线投射法的基本功能，} 采用的编程语言Python并配合 CMAKE脚本生成相应的工程文件，主程序代码在上文中提供。

本程序主要功能：

（1）能够读取.mhd文件并根据其存储格式转换数据类型。

{43%：（2）实现了光线投射法的三种光线算法，并根据设置选择其中一种算法进行绘制。}

{43%：（3）提前设置好了不同灰度范围对应的颜色传输函数、不透明传输函数和梯度传输函数数值，其中颜色传输函数和不透明传输函数是必需的，梯度传输函数则是可选的。}

（4）搭建好了整个程序的VTK体绘制可视化管线，根据数据流随时进行数据更新。

（5）编写了程序的基本交互界面，能够在读取数据后使用鼠标对生成的图像进行视角的变换和视点距离的变化。

4.3.3 程序结果图像

本程序实现了光线投射法三种不同的光线算法，可以根据数据的不同灰度范围赋予相应的颜色传输函数、不透明度传输函数和梯度传输函数，{47%：并实现了数据流的 VTK可视化管线，以及基本的界面交互功能。} 下面是程序运行的结果图像，并根据设置的不同显示不同的结果。

（1）简单光线投射法

{40%：简单光线投射法是最基本的光线投射法的实现，主要原理是从观察的视点出发发射许多光束，} 这些光束穿过点集数据后会根据设置的颜色传输函数和不透明传输函数以及梯度传输函数赋予不同灰度值相应的属性值，{41%：最后整条光线的属性值用于生成最终图像相应像素的像素颜色值。}

图4.3.3（1）a 简单光线投射法（梯度函数关闭）

图4.3.3（1）b 简单光线投射法（梯度函数打开）

其中第二幅图像是打开梯度传输函数的结果图像，梯度函数是用来显示点集数据内部的点集数据之间灰度变化的函数，因此生成的图像显示的是整个数据中灰度值变化最大的部分。

（2）最大密度法

最大密度法是另一种光线投射算法，主要原理是投射光线最终生成的像素的颜色值由这条光线中灰度值最大的体素数据决定，{41%：因此这种算法生成得到的图像显示的是体数据中密度最大的部分。}

图4.3.3（2）最大密度法

（3）特定等值面法

特定等值面算法是一种特殊的算法，根据设定的灰度值，投射的光线最终生成的图像的像素值只由满足设定的灰度值的体素提供，最终得到的图像只会显示满足设定的灰度值的体素数据生成的图像。

图4.3.3（3）特定等值面法

结语

5.1 总结

随着科学技术水平的不断提高，使用各种仪器获取的数据量也不断增加，而如何有效的根据需要在{47%：保留原始数据细节的基础上以最大效率的方式展示这些数据就是科学可视化的主要研究内容。}其中如何显示获取到的各种三维数据，传统的面元绘制方法已经不能满足对于越来越高精度的研究和应用需要，为了克服传统的基于面元绘制的方法的不足，研究人员提出了三维体数据科学可视化的体绘制方法。这种方法不借助于中间面元而直接使用三维点集数据，通过各种不同的算法赋予这些体素数据相应的属性值，最终生成能够直观理解的二维图像。{54%：在体绘制中，最有代表性的是光线投射法，抛雪球法，错切—变化法和基于硬件的三维纹理映射法，}这几种方法各自具有不同的特点，在这些方法的基础上又发展了许多更加有效和高效的改进算法。本文主要就使用VTK可视化软件包的编程环境搭建，其主要架构以及数据流结构进行了介绍，{44%：并使用VTK的Python接口编写体绘制程序，实现了基于光线投射法原理的三种光线算法，}对体绘制程序的主程序代码主要功能进行介绍，并分析了获得的图像结果。

5.2 展望

测绘学科的两大部分分别是数据的获取以及对获取的数据进行处理，随着科学技术水平的不断提高，电子技术、计算机技术、网络技术等的普及，获取数据的能力和手段也不断丰富，获取得到的数据量也呈现出爆发式的增长，而如何处理和应用这些数据便成为了今后研究工作的重点。其中三维数据对于描述现实世界空间关系有着巨大的作用，而如何在克服传统面元绘制的不足的基础上研究出更加高效和准确的显示方式就是本文所介绍的内容。使用科学可视化的体绘制方法，能够尽可能的提取每个体素数据的信息，尽可能的保留点集数据场属性关系，因此这种方法得到了越来越多研究人员的重视，也根据不同的需求提出了相应的各类原理和算法。本文在介绍这些原理和算法的基础上结合实际，使用VTK可视化软件包实现了光线投射法的三种光线算法的基础功能，介绍了VTK编程环境搭建、程序结构和数据流特点等相关内容。主要的目的是通过本文的抛砖引玉，让更多的人能够了解这种体绘制的思想，并将其运用到实践中，发展出更多更有效的具体算法，并丰富这种算法的应用范围，{42%：更多的发挥其优势，使其在实际工作中获得更大范围的应用。}

MDH格式文件读取

数据格式转换

VTK体绘制函数设置

VTK体绘制渲染

体绘制属性设置

设置颜色传输函数

设置不透明度传输函数

VTK渲染引擎

开始程序绘制