天津大学

本科生毕业设计

题目：虚拟文物展的远程渲染方法研究

学　　院　智能与计算学部

专　　业　软件工程

年　　级　2017 级

姓　　名　刘兴宇

学　　号　3017218063

指导教师　张怡

独创性声明

本人声明：所呈交的毕业设计（论文），是本人在指导教师指导下，进行研究工作所取得的成果。除文中已经注明引用的内容外，本毕业设计（论文）中不包含任何他人已经发表或撰写过的研究成果。对本毕业设计（论文）所涉及的研究工作做出贡献的其他个人和集体，均已在论文中作了明确的说明。本毕业设计（论文）原创性声明的法律责任由本人承担。

论文作者签名：

年　　月　　日

本人声明：本毕业设计（论文）是本人指导学生完成的研究成果，已经审阅过论文的全部内容。

论文指导教师签名：

年　　月　　日

摘　要

随着经济与社会的不断发展，传统的文物展览形式已经不能满足观众日益增长的需求，虚拟文物展应运而生，它通过可交互三维模型的形式为观众带来了更为丰富的参观体验。但是庞大的模型数据量对虚拟文物展的实现提出了巨大挑战，远程渲染则是应对这一挑战的一个有效的解决方案。

本文提出了一种基于轨迹分割的分段预测算法。该算法在对视点的历史操作轨迹存在的突兀转折进行轨迹分割后，使用基于线性预测和拉格朗日插值预测的分段预测方法预测视点的下一个位置。借助该算法可以提前渲染下一帧图像，从而大大减小交互延迟。

在此算法基础之上，本文针对虚拟文物展的需求，设计并实现了一套基于用户交互预测的虚拟文物展远程渲染系统。经过测试，该系统拥有较高的预测准确率，并且能够在容忍的延迟范围内实现虚拟文物展的各项功能。

关键词：远程渲染，三维渲染，交互预测，轨迹分割

ABSTRACT

With the development of the economy and society, traditional artifact exhibition can no longer meet the increasing needs of audience. So, the virtual artifact exhibition emerged as the times require. It brings a rich experience to the audience by using interactive 3D model. However, it faces a great challenge because of the huge size of models. Remote rendering is an effective solution to this challenge.

This paper proposes a segmentation prediction algorithm based on trajectory segmentation. This algorithm first segment the trajectory of historical viewpoints which has abrupt turning. Then predict the next position of viewpoint by using a segmentation prediction algorithm composed of linear prediction and Lagrange interpolation polynomial.

Through this algorithm, the next frame can be rendered in advance, which can reduce interaction delay greatly.

Based on this algorithm, this paper design and implements a remote rendering system of virtual artifact exhibition according to the requirement. After testing, this system has a high prediction accuracy and can realize the functions of virtual artifact exhibition within the tolerable delay range.

KEY WORDS: Remote rendering, 3D rendering, Operating forecast, Trajectory segmentation

目 录

第一章　绪论

1.1　研究背景

在人类文明的历史长河之中，拥有五千年悠久历史的中华文明仿佛璀璨王冠上的一颗耀眼宝石。而流传至今的珍贵文物，则是这个辉煌文明的最好见证。近年来，随着人民生活水平的日益提高，以及社会主义文化建设的不断加强，越来越多的人们对博物馆中展出的珍贵文物产生了浓厚的兴趣，如三星堆、西藏当雄吐蕃墓等文物更是成为人们热议的焦点话题。

但是，目前的文物展大多采用的还仅仅是传统的线下展览方式，通过实物配合讲解文字、图片、语音、视频等方式向观众介绍。传统方式虽然能让观众近距离接触到实物，但也有着诸多不便。例如出于安全的考虑，文物通常被放置在密封的橱窗中，不仅限制了观众的观赏角度，还无法呈现更多的细节。如果展出地点与观众距离较远，观众还将花费大量时间与金钱到展览现场观看。特别是在疫情防控的形势下，这将会带来诸多不便。此外，一些文物的保存条件比较苛刻，不适合进行展出。长时间展出的文物也会遭到不同程度的损坏，这些都不利于对文物的保护。

近年来，图形学、互联网、虚拟现实等计算机技术不断进步，使得博物馆中处于精心保护下的珍贵文物得以以虚拟文物展的形式展现在观众面前。虚拟文物展使用三维模型作为主要展出方式，虽然不能接触到文物本体、缺乏庄重肃穆感，但在知识扩展、细节观察、便捷交互等方面有着其特有的优势。参观者无需亲临展览现场，就可以近距离观赏文物，还可以通过旋转、放缩等操作，观察到普通文物展中很难观察到的角度和细节。并且，不断进步的 3D 激光扫描技术也让虚拟文物展在细节呈现上如虎添翼。

传统的三维模型呈现往往采用本地渲染的方法，即将模型文件下载到设备上，然后对模型文件进行解析并进行渲染。但是，对于以呈现高精细度模型为目标的虚拟文物展来说，传统方法存在着诸多瓶颈。首先，高精度的 3D 激光扫描技术在捕获更多细节的同时，也带来了数据量过大的问题。如美国斯坦福大学在对意大利雕塑家米开朗基罗的《大卫像》雕塑进行 0.25mm 精度的三维数据扫描时，数据量达到了惊人的 20 亿个三角形[1]。庞大的三维模型数据量虽然意味着渲染的模型精细度更高，但也对渲染设备的硬件（如 CPU、GPU、内存）提出了更高的要求，而用户所使用的个人计算机设备或移动设备显然不能满足要求，这会

导致渲染过程变得相当漫长[2]。同时由于计算能力的不足，交互过程的流畅程度也会下降，严重影响用户的操作体验。其次，高精度的三维模型文件大小通常以 GB 为单位[3]，将如此巨大的文件下载到用户设备上将会占用大量的带宽资源，同时也需要耗费相当长的时间。此外，珍贵文物的三维数据通常具有涉密性质，三维数据一旦泄露给有恶意的用户，可能造成严重的后果，必须加以保护。总而言之，传统的本地渲染方法已经不再适用。在虚拟文物展的应用背景下，将渲染与展示分离将是大势所趋，也是亟待突破的技术瓶颈。

远程渲染（remote rendering）是指在渲染服务器上运行渲染程序，然后通过网络连接将渲染结果传输给客户端进行展示的渲染系统[4]。它的概念由来已久，早在 2009 年的国际消费类电子产品展览会（CES）上，AMD 公司就展示了一套名为“AMD Fusion Render Cloud”的远程渲染系统[5]。使用一台配置了 Phenom Ⅱ X4 处理器、Radeon HD 4870 X2 的台式计算机作为服务器渲染 EA 公司的“雇佣兵 2：战火纷飞”游戏，并将渲染结果通过有线网络传输给一台 HP Pavilion dv2 超薄笔记本电脑上（该笔记本电脑的配置未达到游戏的最低配置要求）。用户在笔记本电脑中对游戏进行操作，交互指令通过网络传给服务器，服务器再传回相应的结果。

远程渲染采用客户端-服务器架构（C/S），将三维模型的渲染与展示分离。使用具有强大图形计算能力和高速网络连接的渲染服务器存储模型文件，完成对模型的渲染工作，将渲染结果发送给用户，而客户端设备只需完成展示结果的功能。与传统的本地渲染相比，远程渲染具有以下几个优势：

1) 丰富体验，借助渲染服务器的强大计算能力，它可以让硬件资源不足的设备获得更丰富的渲染体验。
2) 资源共享，对同一模型的一次渲染可以被多个用户请求共享，大大减少了重复计算。
3) 数据保护，模型文件被存放在服务器，用户收到的只有渲染结果的二维图像，做到了对模型数据的有效保护[6]。
4) 跨平台支持，由于客户端只需完成结果展示等功能，减少了为不同平台开发不同客户端的工作量[3]。

远程渲染技术将渲染与展示分离，解决了在计算资源不足的设备上进行高质量三维渲染的困难。通过分析不难发现，远程渲染技术与虚拟文物展之间有着极大的相关性。将远程渲染技术应用到虚拟文物展中，可以大幅降低对用户的设备的硬件要求，还能极大地改善用户体验。

1.2 本文研究内容和意义

1.2.1 研究内容

本文从虚拟文物展的用户需求和远程渲染技术中待解决的问题出发，分析了远程虚拟文物展远程渲染系统涉及到的关键技术，研究了远程渲染技术的分类、一般方法和国内外研究现状。针对所提供的文物数据和用户的交互习惯，提出并实现了一种基于用户交互预测的虚拟文物展远程渲染系统，包括了以下研究内容：

1) 研究了远程渲染系统的分类和国内外研究现状，分析了远程渲染技术涉及到的关键技术，介绍了使用的技术要点，包括三维模型数据结构、三维渲染的基本流程、图像压缩算法等。
2) 构建模型渲染模块。根据模型数据文件结构和计算机三维渲染过程的相关知识，使用 python 编程语言，基于 OpenGL 这一图形库以及 pygame 等辅助工具包，解析模型数据文件并根据视点位置渲染三维模型，生成并导出渲染结果的二维图像。采用压缩算法对图像进行有损压缩，并将结果转换为 base64 编码。
3) 构建交互预测模块。定义用户操作和视点坐标系，基于所定义的用户操作，以分段预测为基础，综合拉格朗日插值等方法，实现一种用户交互预测算法。根据用户之前的视点运动情况，预测下一时刻用户可能的视点位置。
4) 构建展示客户端。构建 Web 客户端，建立客户端缓存，用于展示渲染服务器的渲染结果。在前端也建立视点坐标系，通过用户交互捕获、消除抖动、坐标转换等方式保持客户端与服务器视点的一致性。
5) 将渲染模块与预测模块组合为后端程序，借助 socket 编写后端服务程序，并部署在 Django 框架上。将后端程序部署在渲染服务器上，与客户端进行连接并测试渲染效果、交互延迟和网络状况。

1.2.2 研究意义

本文以虚拟文物展的远程渲染为目标，提出了一个基于用户交互预测的虚拟文物展远程渲染系统，并测试了它的实际效果，本文的研究具有以下意义。

1) 本文的研究及应用对虚拟文物展的实现有着重要意义。本文提出的远程渲染系统解决了因文物数据过大导致的计算资源不足、响应时间长、用户体验差的问题。使得绝大多数用户可以使用普通的计算机设备和网络连接获得更加丰富的渲染效果，再次基础上还大大降低了交互延迟，提升了用户体验，也为文物研究者们提供了远程观察的方法和高精度细节

的呈现。这不仅为参观和研究提供了便利，同时也有利于文化遗产的保护，能够使更多的人了解、欣赏文物。

2) 本文的研究在文物数据保护上也有着重要意义。在本文提出的远程渲染系统中，文物数据将被存储在渲染服务器中，渲染服务器只将渲染结果的二维图像传输给客户端，用户能接收到的只有图像数据。这很好地保护了文物数据不会泄露给有恶意的用户，有效防止仿制文物的出现。

1.3 本文结构

本文共分为七个章节，每个章节的主要内容如下：

第一章介绍了本文的研究背景，进一步引出本文的主要研究内容和意义。

第二章介绍了本文相关技术的研究情况。首先介绍了远程渲染的相关技术知识，包括远程渲染系统的分类、架构和国内外研究现状。之后介绍了三维渲染的相关技术，包括 OBJ 数据模型格式、图形渲染管线和本文所用到的图形库。最后则介绍了本文使用的图像压缩算法、拉格朗日插值法等相关技术要点。

第三章对本文研究的虚拟文物展远程渲染系统的用户需求进行分析，针对需求提出了基于用户交互预测的虚拟文物展远程渲染系统，并介绍了其概要设计。

第四章介绍了系统的详细设计过程。

第五章介绍了本文提出的基于轨迹分割的分段预测算法。

第六章则是对系统的实现方法和测试过程的介绍。

第七章是对全文的总结与展望。

1.4 本章小结

本章主要介绍了虚拟文物展远程渲染的研究背景，提出了本文的研究内容和研究意义，并对本文的行文结构和各章主要内容进行了介绍。

第二章　相关技术

2.1　远程渲染相关技术综述

远程渲染是指在一台高性能计算机设备上运行三维图形渲染应用程序，然后在另一台计算机设备上通过网络连接展示渲染结果的系统[4]。运行渲染应用程序的计算机设备被称为渲染服务器（以下简称“服务器”），展示渲染结果的计算机设备被称为展示客户端（以下简称“客户端”）。而一个可交互远程渲染系统则能够通过客户端上的输入设备（如鼠标、键盘和摇杆等）捕获用户的控制操作，并将操作指令发送给服务器，服务器则根据指令重新渲染画面，再将新的渲染结果传回给客户端，更新画面。对于虚拟文物展来说，交互是必不可少的功能，因此，本文所研究的为可交互式远程渲染系统（下文中以“远程渲染系统”代替）。

最初，远程渲染的概念的提出是为了解决个人计算机无法进行三维渲染的问题。随着图形学、云计算和高速网络通信技术的不断进步，远程渲染逐渐变为为没有高性能计算设备的“瘦客户端”提供高质量渲染服务的技术。本节将介绍远程渲染系统的分类、国内外研究现状和基本架构。

2.1.1　远程渲染系统的分类

一个远程渲染系统通常是为了解决特定应用场景的需求而专门设计的，这是由渲染的模型数据类型和与用户的交互能力决定的[4]。其中，数据类型是指需要渲染的模型数据是静态还是动态，这决定了模型数据是否会随着系统运行而不断更新。而交互能力则是指用户操作模型数据的能力，决定了用户是否能完全自由地进行变更摄像机位置、朝向，以及更改模型数据的操作。根据这两个因素，远程渲染系统可以被分为以下四种类型，如图 2-1 所示：

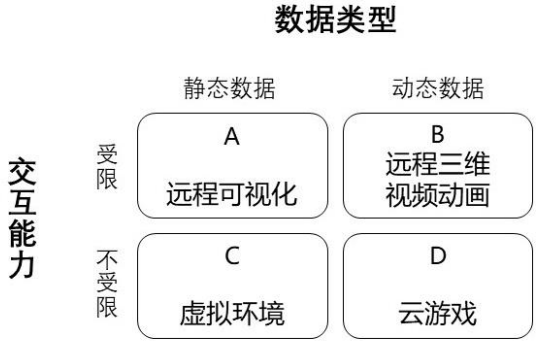


图 2-1　远程渲染系统分类示意图

类别 A：静态数据+受限制的交互能力。A 类型的远程渲染系统渲染的通常是静态的模型数据，用户只能通过有限的方式与系统进行交互。在这种类型的远程渲染系统中，用户的行为会被严格限制。例如用户只能通过几个预设的位置或在有限的范围内浏览模型数据，并且只能在限定的范围内更改模型数据。本类别的代表例子是远程可视化系统，如医疗图像、数据统计、工业设计、艺术品等。它们通常具有庞大的数据量，因此无法做到与用户自由地交互。

类别 B：动态数据+受限制的交互能力。B 类型的远程渲染系统的模型数据会随时间不断发生变化，但用户仍然只能通过有限的方式与系统进行交互。本类别的系统通常是三维视频、动画的远程渲染系统。由于它们拥有的高帧数（至少每秒 24 帧），这就需要远程渲染系统在上一帧到达前就处理好下一帧的渲染工作，因此不可能与用户进行自由的交互。

类别 C：静态数据+不受限的交互能力。C 类型的远程渲染系统 。本类别系统的代表是已经被广泛应用在虚拟现实技术（VR）中的虚拟环境技术（VE）[7]，它们渲染的虚拟环境信息仍然是由固定的静态数据产生，但用户却可以在沉浸式场景中自由进行交互，但用户仍然只能在受限制的范围内改变模型数据（如添加预设好的数据、删除已有数据等操作）。

类别 D：动态数据+不受限的交互能力。D 类型的远程渲染系统不仅模型数据会随时间发生变化，用户还能自由地与模型进行交互。本类别系统的主要应用是云游戏系统。游戏不仅具有如视频一样的高帧率，处理复杂的场景变化和运动效果，还要可以任由玩家进行操作，并且还要保证交互延迟处于较低的水平，是所有类型中最困难的类型。

2.1.2 国内外研究现状

根据 2.1.1 节的分析不难看出，虚拟文物展的远程渲染系统是对文物模型的远程可视化，因此属于 A 类别的远程渲染系统：渲染静态数据且用户的交互能力受限。下面，本文将介绍一些 A 类别远程渲染系统的国内外研究现状。

国内外早已有许多学者开发了应用于各种场景的远程渲染系统。Bethel 等人开发了一个大规模科学数据可视化远程渲染系统[8]。Engel 等人将大规模医疗体数据进行二维切片，并利用远程渲染开发了一个医疗图像可视化系统[9]。Prohaska 等人设计了一种基于远程渲染的 Mrcio-CT 医疗扫描系统，它通过只展示比源数据小的多的数据子集来加快渲染速度[10]。

在国内外对远程渲染系统的研究中，如何减少交互延迟是研究的一个核心问题。交互延迟是指从用户操作请求产生到客户端屏幕上第一帧图像更新的时间间

隔[11]，它是远程渲染系统面对的关键挑战之一，图 2-2 展示了一次远程渲染请求的延迟组成。

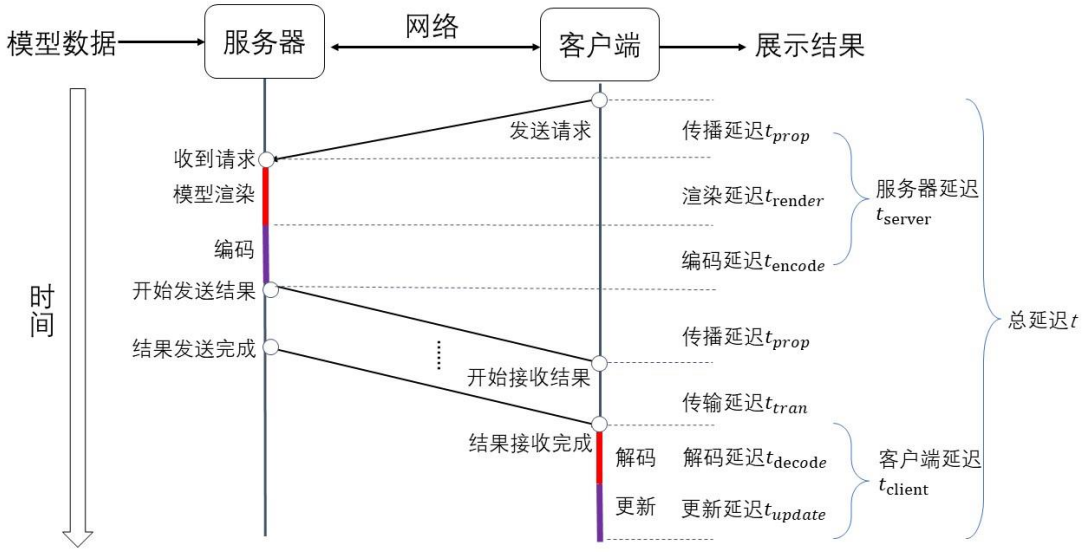


图 2-2　一次远程渲染请求延迟示意图

它由四部分组成：服务器应用程序运行延迟t_{server}，客户端应用程序运行延迟t_{client}，信号发送的传输延迟t_{trans}以及信号在物理链路上的传播延迟t_{prop}。其中t_{trans}和t_{prop}与网络因素有关，称为网络延迟。而t_{server}与t_{client}与网络因素无关，称为非网络延迟。

对于网络延迟，无论如何优化网络结构、提高带宽，也无法突破网络信号传输和传播的下限。因此，对于减少交互延迟的研究主要集中在非网络延迟方面，国内外学者提出了多种方式以降低程序运行时产生的非网络延迟。Yu 等人提出了一种自适应简化模型的方法（ASM），对模型的简化可以剔除模型数据中存在的一些空洞、不需要的细节，在尽量不降低清晰度的前提下，减小模型数据的大小，加快渲染速度[12]。Levoy 等人则提出在客户端生成简化的低质量模型并配合与高质量模型的差异图像，可以大大节约计算资源，减少渲染时间[13]。Boukerche 等人通过预拉取模型全景图的方式，覆盖了用户观察点的移动操作，从而避免了重新渲染花费的时间[14]。

基于图像的渲染（Image based rendering，简称为 IBR）是一种有效的降低交互延迟的远程渲染方法。它需要服务器在传输结果图像的基础上携带一些额外信息，根据这些信息，通过在客户端运行 IBR 算法来合成新的图像。因此可以大

幅减少网络请求的次数，非网络交互延迟可以减少到运行 IBR 算法所花费的时间[4]。

三维图像变形（3D image warping）是一种典型的 IBR 算法，它使用已有观察点的结果图像、深度图像来合成目标观察点的图像。Shi 等人提出了一种基于三维图像变形算法的渲染系统[15]，通过一次性拉取一些预设观察点的渲染结果图像及深度图像，使用三维图像变形算法拼接已有图像从而生成目标观察点的渲染结果。Mark 等人的研究与 Shi 等人的研究类似[16]。在他们的设计中，服务器仅渲染关键的图像帧，而客户端则使用三维图像变形算法在关键帧之中插入合成的中间帧，以减少渲染时间和在网络上传输的帧的数量，从而降低总延迟。

2.1.3 远程渲染系统的基本架构

本文所研究的虚拟文物展远程渲染系统属于 A 类远程渲染系统，其基本架构如图 2-3 所示：

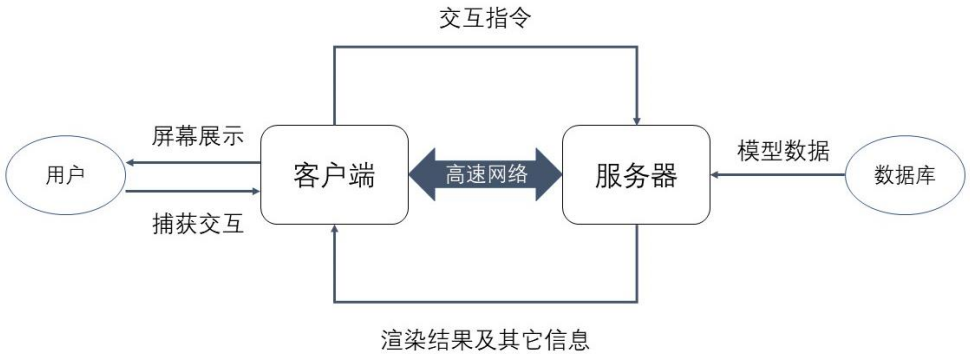


图 2-3 远程渲染系统的基本架构示意图

渲染服务器拥有强大图形计算能力，连接着存储模型数据的数据库。服务器根据客户端发送的指令，选择模型进行渲染，并根据用户交互指令调整模型，更新渲染结果，将渲染结果以二维图像的形式发送给客户端。

客户端设备一般不具备足够的图形计算能力，从服务器接收渲染结果并呈现在屏幕上。通过输入设备捕捉用户交互，将交互指令发送给服务器，接收到响应后根据更新策略更新显示结果。

服务器与客户端之间通过高速有线或无线网络连接。

本文所提出的基于用户交互预测的虚拟文物展远程渲染系统也是依据上文介绍的基本架构而设计。

2.2 三维渲染相关技术简介

三维渲染是远程渲染系统的核心技术之一，也是渲染服务器的重要模块。它是将三维模型数据构建为三维场景，并将其以二维图像的形式显示到计算机屏幕上的过程。本节将首先介绍计算机的图形渲染流水线，之后介绍 OpenGL 这一图形库，最后介绍本文研究过程中使用的 OBJ 格式模型数据。

2.2.1 计算机图形渲染流水线

计算机图形渲染流水线是计算机处理和渲染三维数据的过程[17]，是远程渲染技术的核心步骤之一。图 2-4 展示了一个典型的图形渲染流水线模型，它主要分为两个阶段：几何处理与光栅化。

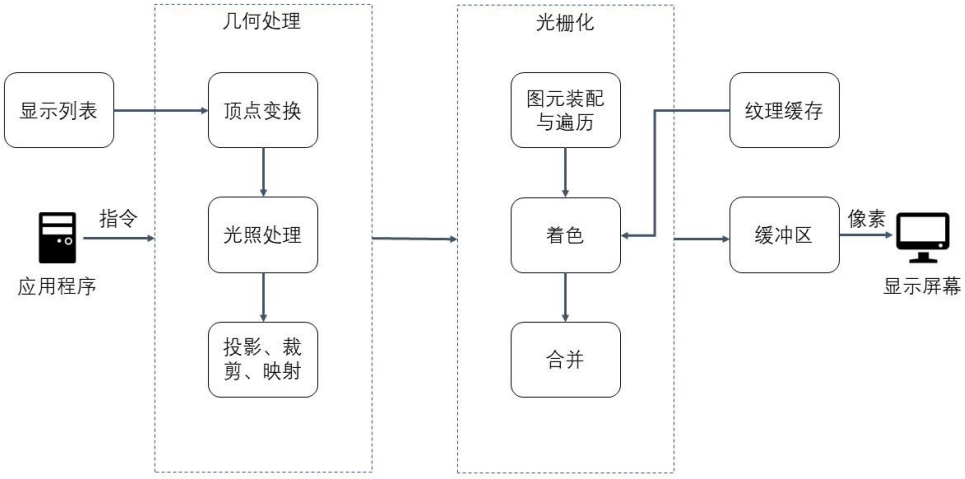


图 2-4 计算机图形渲染流水线模型示意图

在几何处理阶段，主要对模型数据进行变换与加工，包括以下几个阶段：

1) 顶点变换：将模型原数据的顶点坐标由模型自己的坐标系变换为世界坐标系（图 2-5(a)），再进一步变换到放置在世界坐标系中的视点坐标系（图 2-5(b)）。其中，视点指观察者的位置、视线方向。
2) 光照处理：决定光线在每个顶点上的效果，为后续的光栅化做好准备。
3) 投影、裁剪与映射：这个阶段的作用是将三维的模型数据转换为二维的数据形式，其中分为三个子阶段：
   a) 投影：将三维可视顶点映射到一个二维平面中，包括平行投影和透视投影两种方式。
   b) 裁剪：保留在视点可视范围内的顶点。

c) 映射：将顶点坐标映射为屏幕坐标系（图 2-5(c)）的二维坐标。

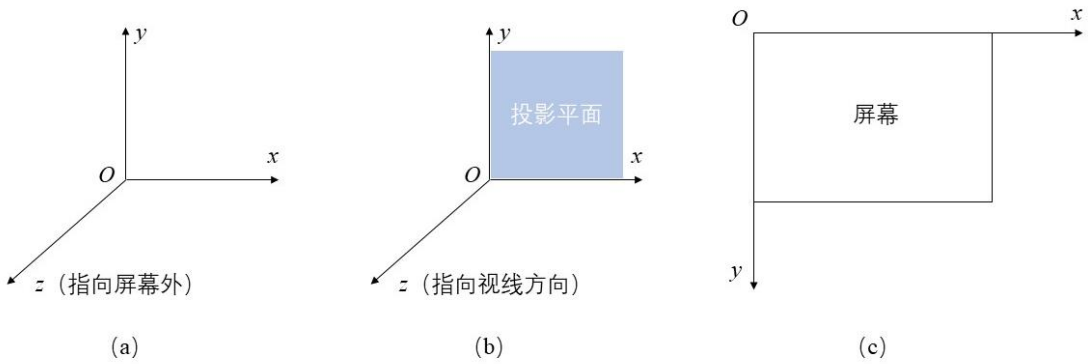


图 2-5 计算机图形渲染流水线使用的三种坐标系

(a) 世界坐标系；(b) 视点坐标系；(c) 屏幕坐标系

在光栅化阶段，流水线对模型的几何数据进行最后加工并最终转换为像素，分为三个阶段：

1) 图元装配与遍历：将顶点数据根据连接关系还原为像素网格结构，决定每一个像素在哪一个由顶点组成的三角形片元之中。
2) 着色：根据颜色、纹理索引，对像素进行颜色、纹理插值。
3) 合并：从纹理缓存中提取数据，将像素与着色阶段产生的颜色合并，并进行一些测试。将结果发送至帧缓冲区。

最终，由帧缓冲区将图像显示在屏幕上，完成一次渲染流水线的过程。

2.2.2 OpenGL

OpenGL（Open Graphics Library），也被称作“开发图形库”，是用于渲染 2D、3D 矢量图形的一种跨语言、跨平台的应用程序编程接口[18]，是对 2.2.1 章节中介绍的计算机图形渲染流水线的一种实现。

OpenGL 拥有以下几个优势：

1) 拥有高图形质量、高性能，渲染速度更快。
2) 良好的稳定性，有统一的规范组织，处于不断更新之中。
3) 强大的可移植性和跨平台性，OpenGL 不考虑具体的实现方式，采用 OpenGL 标准编写的应用程序可以运行在任何支持 OpenGL 标准的硬件上。

OpenGL 因其自身的优势而在工业界和学术界都得到了广泛的应用，本文所研究的虚拟文物展远程渲染系统也正是使用了该编程接口。

2.2.3 OBJ 模型简介

本文的研究中，使用的是以".obj"扩展名结尾的三维模型数据文件。它是由 Wavefront 公司开发的一种 3D 模型文件格式规范。一个典型的 obj 模型包括四种文件：模型文件、材质文件、贴图文件和法线贴图文件。

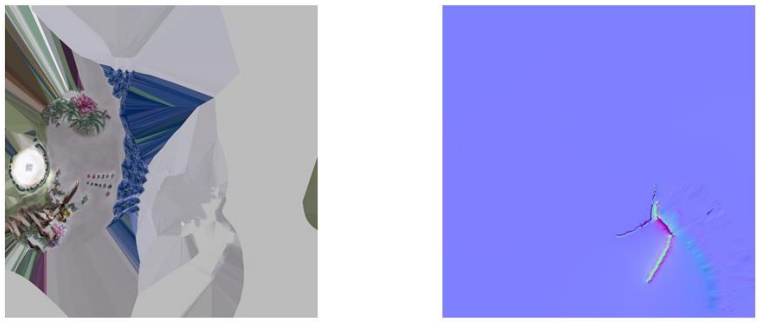
模型文件：以".obj"作为扩展名，主要描述模型的结构信息，包括顶点坐标、纹理坐标、法线向量、多边形的顶点索引等信息。

材质文件：以".mtl"作为扩展名，主要描述物体的材质信息，包括光照信息、纹理贴图映射、法线贴图映射等信息。

纹理贴图：图像格式的文件，是覆盖在模型多边形上的图案，图 2-6(a)是一个典型的纹理贴图示例。

法线贴图：图像格式的文件，通过 RGB 颜色通道来标识每个顶点的法线方向，用于在渲染中增强模型表面光照与反射效果。图 2-6(b)是一个典型的法线贴图示例。

obj 文件使用文本存储模型数据信息，非常利于读取与解析。根据 obj 规范构造的模型数据，可以被许多知名 3D 模型软件导入与导出，许多激光扫描技术生成的也是 obj 格式的模型文件[19]。



(a) (b)

图 2-6 两种贴图文件

(a) 纹理贴图；(b) 法线贴图

2.3 图像压缩算法

对于远程渲染系统，减少带宽使用量是减少交互延迟的有效方法，对服务器的渲染结果图像进行压缩可以有效减少传输的数据量。图像压缩算法分为有损压

缩和无损压缩两种，在本文的研究过程中使用了一种有损图像压缩算法 JPEG，算法的基本流程如下：

颜色模式转换：将原图片像素颜色值由 RGB 色彩空间转换为 YUV 色彩空间（Y 代表亮度信息，U、V 代表色差信息）。

分块：将原图像分割为 8*8 像素的块。

变换：对每个块，进行离散余弦变换（DCT），将高频值集中到块右下方。

量化：使用量化表对不同频率的像素值进行运算压缩。

编码：对压缩后的数据进行编码。

JPEG 算法通过利用人眼对计算机色彩中的高频信息不敏感的特点，来去除图像中的冗余信息。与其它压缩算法相比，拥有压缩率高、色彩丰富、尺寸可变的优点。远程渲染系统借助该算法可以有效减少结果图像的文件大小，减少带宽占用量，加快传输速度。

2.4 拉格朗日插值法

在本文所提出的基于用户交互预测的虚拟文物展远程渲染系统中，提出并实现了一种基于轨迹分割的分段预测算法，在该算法中使用了拉格朗日插值法作为分段预测中某一种情况下的预测方法。

拉格朗日插值法是以 18 世纪法国著名科学家约瑟夫·路易斯·拉格朗日的名字命名的一种多项式插值方法。可以根据两个变量的多个不同观测值，通过插值拟合出变量之间的一个多项式关系，使得该多项式恰好满足各个观测值，这样的多项式就被称为拉格朗日多项式。根据这个多项式，可以得出一个二维平面上恰好通过若干已知点的多项式函数，其数学定义如下：

一般地，若已知函数$y = f(x)$在$[a, b]$上有定义，且有$n + 1$个互异的点$x_0, x_1, \dots, x_n$处的函数值为$y_0, y_1, \dots, y_n$，则可以构造一个次数不超过n的多项式$y = P_n(x)$满足：

$$P_n(x_k) = y_k, k \in [0, n] \tag{2-1}$$

函数$y = P_n(x)$称为$f(x)$的插值函数，$x_0, x_1, \dots, x_n$为插值点。

下面为拉格朗日多项式的求法，设$D_n = \{0, 1, \dots n\}$，对于任意$k \in D_n$，都有$p_k(x), B_k = \{i \mid i \neq k, i \in D_n\}$，使得：

$$p_k(x) = \prod_{i \in B_k} \frac{x - x_i}{x_k - xi} \tag{2-2}$$

满足任意$m \in B_k, p_k(x_m) = 0$且$p_k(x_k) = 1$，最终可得：

$$L_n(x) = \sum_{n}^{j=0} y_j p_j(x) \tag{2-3}$$

形如$L_n(x)$的多项式被称为拉格朗日插值多项式。利用该多项式，就可以对下一个点的坐标进行预测。设下一点坐标为x_m，则$L_n(x_m)$值即为预测的函数值。

根据拉格朗日插值法获得的拉格朗日插值多项式，就可以对用户操作轨迹上的下一个点的位置进行预测并提前渲染结果图像，从而达到减少交互延迟的目的。

2.5 本章小结

本章主要介绍了本文研究中使用到的相关技术，包括远程渲染的相关技术：远程渲染的分类、基本架构和国内外研究现状；三维渲染的相关技术：计算机图形渲染流水线、OpenGL 和 OBJ 模型；JPEG 图像压缩算法；拉格朗日插值法。

第三章　需求分析与概要设计

本文的研究目标是开发一个适用于虚拟文物展的远程渲染系统，本章主要分析系统的用户需求，并对系统进行概要设计。

3.1　需求分析

3.1.1　功能需求

虚拟文物展远程渲染系统的主要涉众只有一类群体，即参观虚拟文物展的观众，此类用户的用例情况如图 3-1 所示：

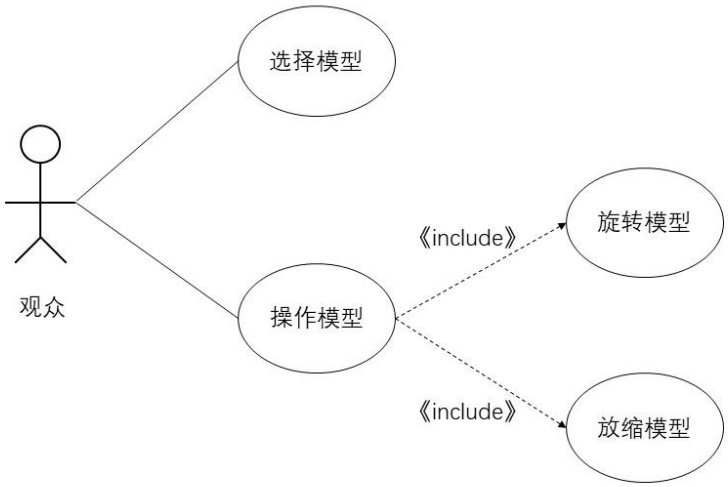


图 3-1　观众用例图

根据用例图可以看出，观众主要有三个用例：选择模型、旋转模型、放缩模型，表 3-1、3-2、3-3 分别对每个用例进行了描述。

表 3-1　选择模型用例描述

编号：1	**用例名称：选择模型**
执行者	观众
前置条件	系统处于就绪状态或交互等待状态

表 3-1　选择模型用例描述（续）

编号：1	**用例名称：选择模型**
主事件流	1. 用户打开模型列表，浏览模型名称 2. 用户根据模型名称，选择模型 3. 系统根据模型名称渲染模型 4. 系统显示用户所选模型的初始画面
辅助事件流	3. 系统根据模型名称渲染模型 如果用户选择的是空模型，则什么也不做，显示空画面并回到就绪状态
后置条件	若选择的模型不为空，系统处于交互等待状态

表 3-2　旋转模型用例描述

编号：2	**名称：旋转模型**
执行者	观众
前置条件	系统处于交互等待状态
主事件流	1. 用户按下鼠标左键，并向任意方向拖动 2. 系统根据用户操作方向，重新渲染模型 3. 系统显示新的画面
辅助事件流	无
后置条件	系统处于交互等待状态

表 3-3　放缩模型用例描述

编号：3	**名称：放缩模型**
执行者	观众
前置条件	系统处于交互等待状态
主事件流	1. 用户滚动鼠标滚轮 2. 系统根据用户滚轮滚动方向，重新渲染模型 3. 系统显示新的画面
辅助事件流	2. 系统根据用户滚轮滚动方向，重新渲染模型 如果滚轮的滚动达到了系统设定的某一方向的边界值（最大值或最小值），则设么也不做，画面保持不变
后置条件	系统处于交互等待状态

根据上述用例描述，可以得出观众参加虚拟文物展的活动图如图 3-2 所示：

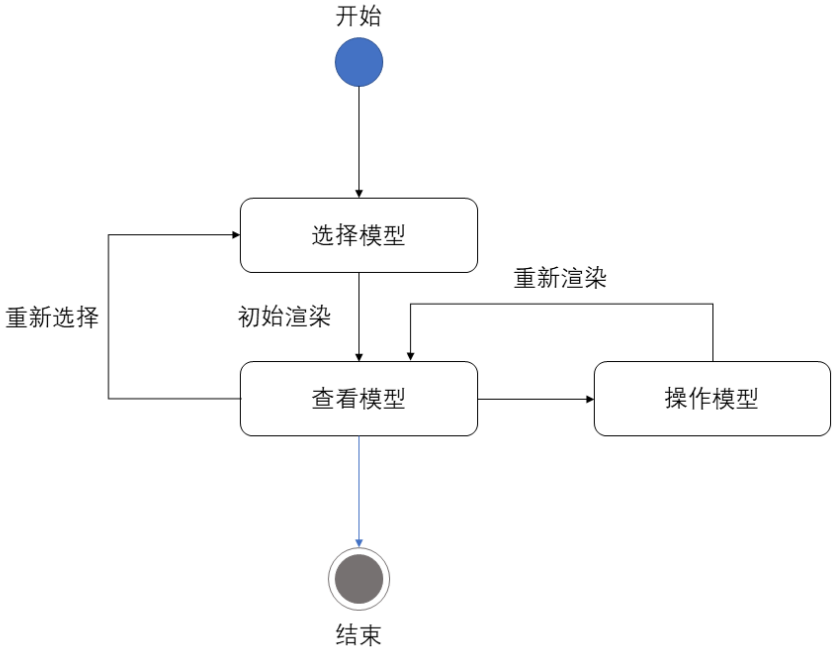


图 3-2　观众参加文物展过程的活动图

3.1.2　性能需求

对于远程渲染系统，交互延迟是一个重要的性能指标。在本文研究的系统中，为了达到较为流畅的用户体验，平均交互延迟应该被限制在 1 秒以内。

3.1.3　其它需求

本系统应该使用“瘦客户端”，即不需要用户额外安装插件等应用程序，用户设备只需连接网络就可以直接访问本系统。

3.2　概要设计

根据 3.1 章节对用户需求的具体分析，本文提出了一个基于用户交互预测的虚拟文物展远程渲染系统（以下简称“本系统”）。本系统遵循 2.1.3 小节中介绍的可交互远程渲染系统的基本框架，采用客户端-服务器架构（C/S）。本系统被划分为客户端和服务器两个部分，其体系结构如图 3-3 所示：

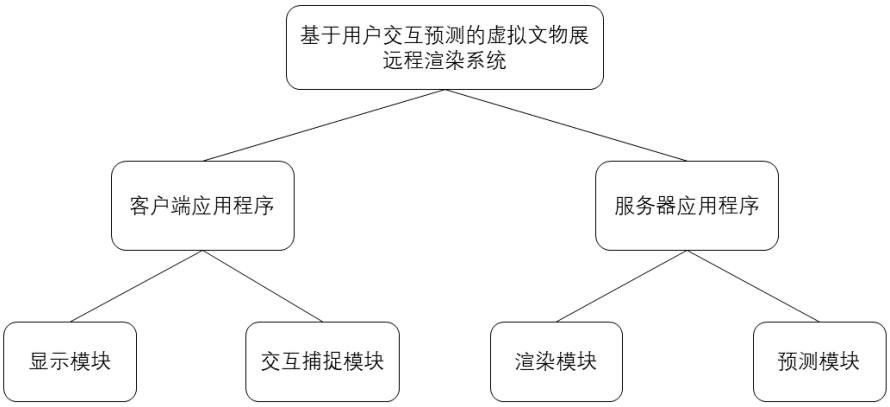


图 3-3　本系统体系结构

客户端应用程序的主要作用是处理用户交互操作，将它们发送给服务器，并向用户展示从服务器接收到的渲染结果，它包括两个模块：显示模块与交互模块。显示模块负责维护图像缓存，更新客户端显示画面；交互模块负责捕捉用户交互（包括更改模型、旋转和放缩模型），平滑用户操作轨迹，生成交互指令。

服务器应用程序的主要作用是接收用户的交互指令并生成对应的渲染结果，将渲染结果发送给客户端。它也包括两个模块：渲染模块与预测模块。渲染模块负责根据模型信息和视点坐标信息渲染三维模型，生成二维图像结果，并进行压缩和编码；预测模块负责预测用户下一个视点坐标。

本系统提出了“视点参数”的概念，“视点参数”是指包括视点位置坐标、朝向在内的用于描述模型变换信息的参数。为了保持服务器渲染内容与客户端显示画面的一致性，服务器与客户端应该拥有同样的视点参数，即客户端画面中摄像机所处位置、朝向应该与服务器渲染程序中摄像机所处位置、朝向保持一致。

3.3　本章小结

本章分析了虚拟文物展远程渲染系统的用户需求，进行了用例分析，构建了活动图，同时分析了系统性能需求和其它需求。之后提出了基于用户交互预测的虚拟文物展远程渲染系统，并进行了概要设计，设计了系统的体系结构和模块组成，分析了服务器和客户端拥有的功能。

第四章　详细设计

根据第三章对用户需求的具体分析以及系统的概要设计，本章对本系统进行了详细设计。在接下来的章节中将对本系统的各个模块详细设计进行介绍。

4.1　服务器设计

服务器部分由渲染模块、预测模块和服务器应用程序组成。服务器的主要工作是从客户端接收用户的交互指令，生成对应的渲染结果，将渲染结果发回客户端。并预测用户的旋转和放缩操作，提前将结果缓存发送给客户端。本节将分别对服务器各个部分的设计进行介绍。

4.1.1　渲染模块设计

渲染模块的主要工作是根据模型信息和视点参数渲染三维模型，生成二维图像结果，并对结果进行压缩和编码。渲染模块的工作流程如图 4-1 所示，该模块以视点参数为输入，输出渲染结果图像的字符串编码。

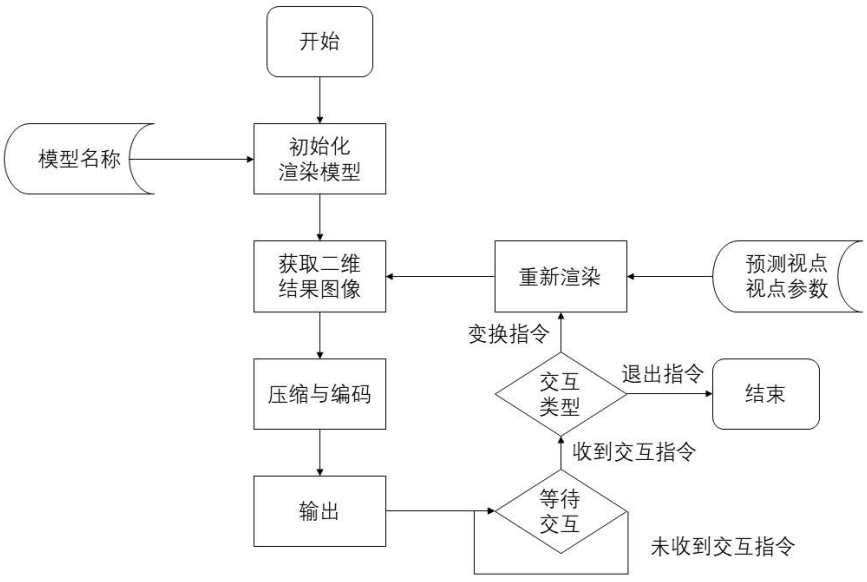


图 4-1　渲染模块工作流程图

当收到渲染模块程序启动指令后，程序初始化渲染环境并根据模型名称对该模型进行初始渲染，获取初始结果的二维图像，经过压缩与编码后以字符串形式输出。之后程序进入交互等待状态，等待来自客户端的交互指令。在收到交互指令后首先判断指令类型，如果为退出指令则结束程序。否则如果为变换指令，则根据接收到的视点参数对模型进行变换后重新渲染，之后获取新的结果图像，压缩与编码后输出，再次回到交互等待状态，直到接收到退出指令。

4.1.2 预测模块设计

预测模块的主要工作是预测用户旋转操作中的下一个视点参数。预测模块的工作流程如图 4-2 所示，该模块以用户当前视点参数为输入，输出预测的下一个视点参数中的旋转角度坐标。预测模块使用的基于轨迹分割的分段预测算法将在第五章中详细介绍。

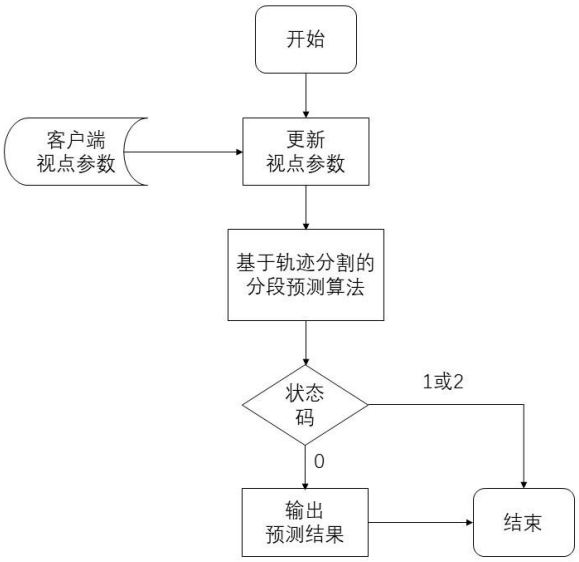


图 4-2 预测模块工作流程图

当收到预测模块程序启动指令后，程序根据从客户端接收到的用户当前位置视点参数，运行基于轨迹分割的分段预测算法，输出状态码。若得到状态码 0，则再输出预测结果。

4.1.3 服务器应用程序设计

服务器应用程序是运行在服务器上的应用程序，它将渲染模块、预测模块组装到一起，维护服务器的视点参数信息，完成接收与响应网络请求、路由跳转、

模块调用与模块间数据传输等功能。服务器应用程序的工作流程如图 4-3 所示。

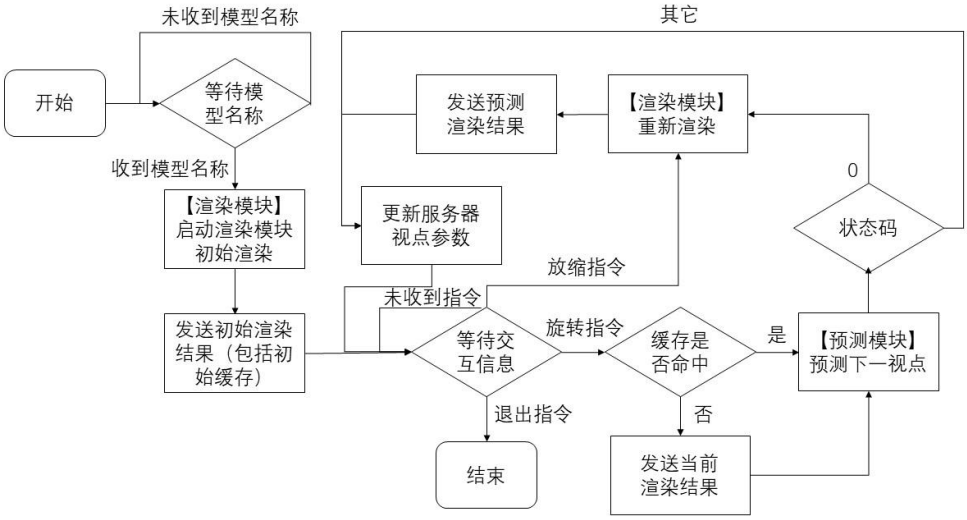


图 4-3　服务器应用程序工作流程图

服务器应用程序在收到客户端的连接请求后启动，然后等待客户端发送模型名称。当收到用户选择的模型名称时，服务器应用程序启动渲染模块进行初始渲染，同时渲染相同旋转角度、大一级和小一级放缩等级的图像作为初始缓存，并将它们发送给客户端。之后进入交互等待状态，等待客户端的交互指令。

当收到旋转操作的交互指令时，服务器应用程序首先判断指令中是否标注了“当前视点缓存未命中”，即客户端并有没有储存用户当前视点的渲染结果。若标注了“缓存未命中”，服务器应用程序会先调用渲染模块渲染并发送当前视点的结果，若未标注则跳过该步骤。之后将当前视点参数发送给预测模块进行预测，若状态码不为 0，代表无法进行预测，则完成本次交互响应，返回交互等待状态。若状态码为 0，则根据预测的新视点参数调用渲染模块重新渲染，并发送新的渲染结果，之后更新服务器视点参数后返回交互等待状态。

由于放缩操作定义，用户只能增加/减少固定数值的放缩倍率，因此不会出现缓存未命中的现象。因此当收到放缩操作的交互指令时，服务器应用程序只需调用渲染模块，渲染比当前视点位置大一级（或小一级，取决于用户的操作方向）放缩等级的结果，并将其发送给客户端，之后更新服务器视点参数后返回交互等待状态。

当收到退出指令时，服务器应用程序终止与客户端的连接，程序结束。

4.2 客户端设计

客户端部分由显示模块、交互捕捉模块和客户端应用程序组成。客户端的主要工作是处理用户交互操作，将它们发送给服务器，并向用户展示从服务器接收到的渲染结果。本节将分别对客户端各个部分的设计进行介绍。

4.2.1 交互捕捉模块设计

交互捕捉模块的主要工作是捕捉用户交互（包括更改模型、旋转和放缩模型），平滑用户操作轨迹，生成交互指令。交互捕捉模块的工作流程如图 4-4 所示，该模块以用户的交互操作为输入，输出一个视点参数或模型名称。

当一个交互操作产生时，交互捕捉模块程序启动。根据不同的交互操作，程序进行不同处理。收到选择模型的操作，输出模型名称和代表操作类型的类别标识符 0；收到放缩操作，输出新视点参数和类别标识符 1。

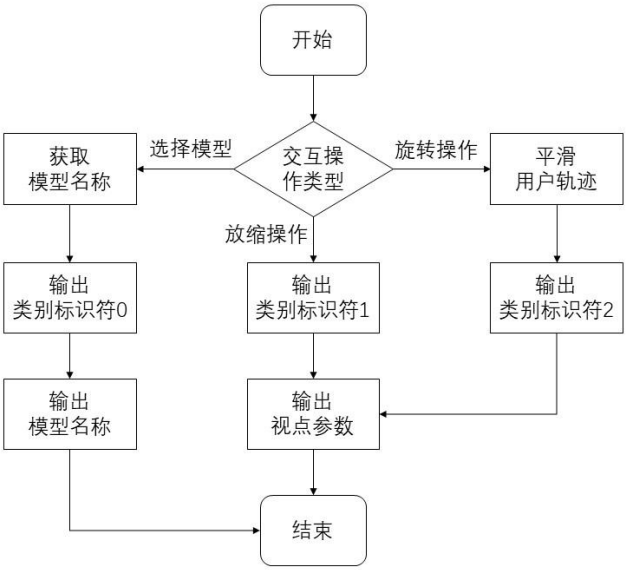


图 4-4　交互捕捉模块工作流程图

由于用户在使用鼠标进行旋转操作时，鼠标的运动会有微小的抖动。如果不对这些微小抖动进行处理，很容易在服务器进轨迹预测时产生过拟合现象，影响预测的准确性。因此需要将轨迹进行平滑处理。

本模块采用了增大分度值的方法，降低在客户端屏幕上鼠标移动相同距离对应的视点旋转角度坐标值的变化量，从而使得小于分度值大小的抖动被忽略，实现平滑处理。一个平滑处理的实例如图 4-5 所示。

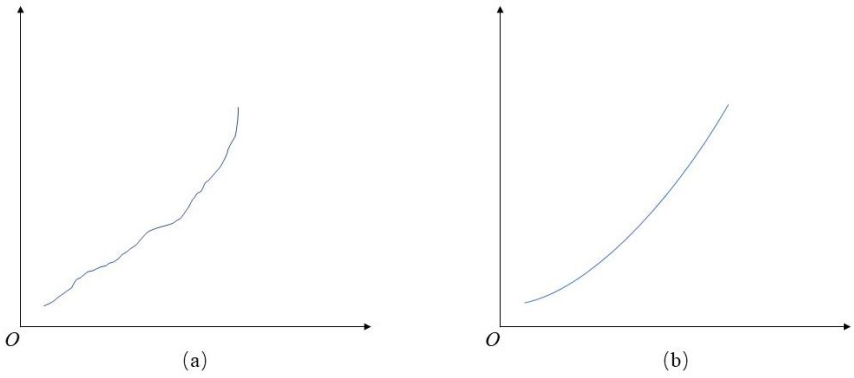


图 4-5　鼠标轨迹的平滑处理

(a)　处理前；(b)　处理后

在平滑处理后，输出新视点参数和类别标识符 2。

4.2.2　显示模块设计

显示模块的主要工作是维护图像缓存，更新客户端显示画面。显示模块的工作流程如图 4-6 所示，该模块以从服务器收到的图像作为输入，向屏幕输出画面。显示模块维护客户端的图像缓存 B，B 是一个大小为 360*360*10 的三维缓存，B[a][b][c]存储视点参数为：旋转角度(a,b)，缩放等级 c 的结果图像。

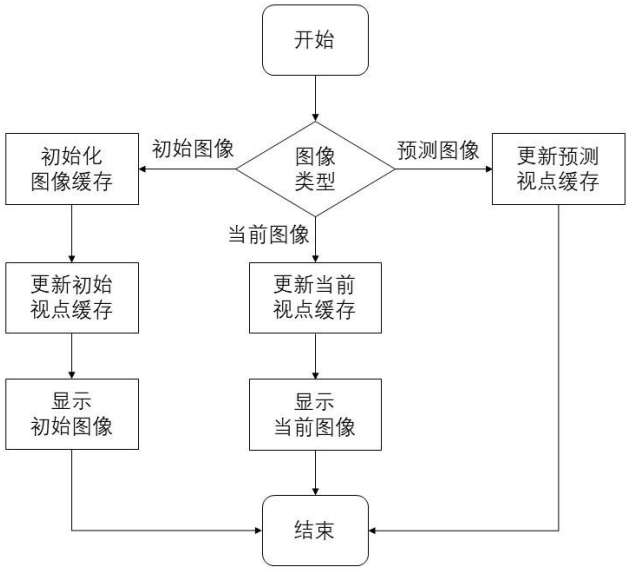


图 4-6　显示模块工作流程

当收到服务器传来的图像时，显示模块程序启动。根据不同的图像类别，程序进行不同处理。

收到初始图像，初始化清空图像缓存，将初始图像和初始放缩缓存存入初始视点缓存中，在屏幕上显示初始图像。

收到当前图像，更新当前视点缓存，在屏幕上显示当前图像。

收到预测图像，更新对应预测视点参数位置的缓存，不改变屏幕显示。

4.2.3 客户端应用程序设计

客户端应用程序是运行在客户端上的程序，它将交互捕捉模块、显示模块组装到一起，维护客户端的视点参数信息，完成处理用户交互操作，将它们发送给服务器，并向用户展示从服务器接收到的渲染结果的功能。客户端应用程序的工作流程如图 4-7 所示：

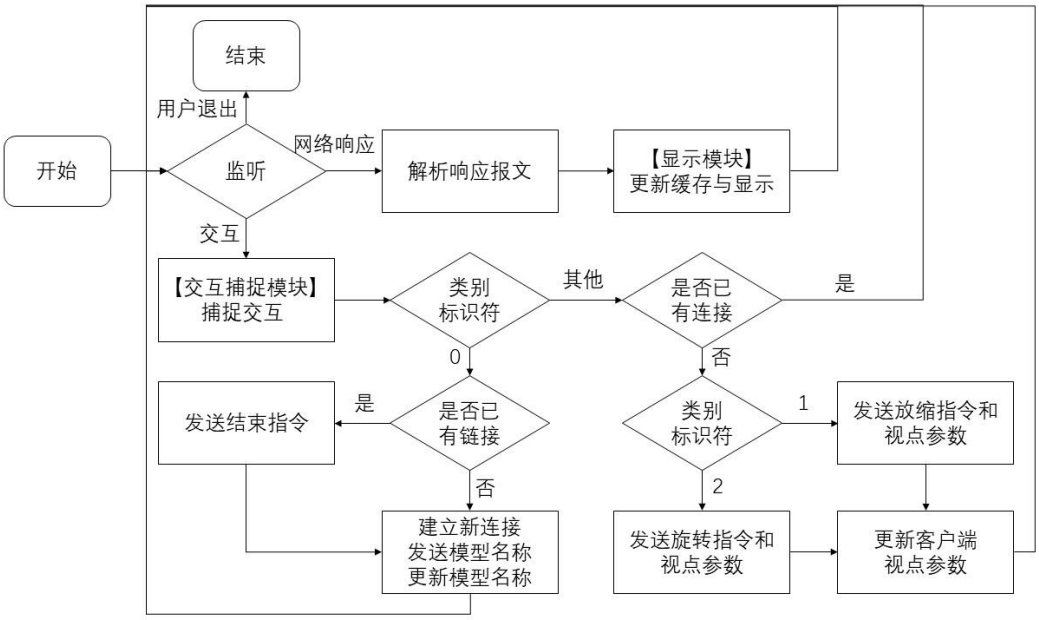


图 4-7　服务器应用程序工作流程

客户端应用程序随用户启动系统而启动，启动后对用户操作和网络进行监听。当监听到用户交互时，就将调用交互捕捉模块捕捉交互，根据输出类别标识符进行如下处理：

- 0：代表用户选择或更改了模型，此时客户端先判断是否已经与服务器建立了一个网络连接，如果存在连接则先发送一个结束指令终止连接。

之后，客户端与服务器建立新的网络连接，启动新的服务器应用程序并将模型名称发送给它。程序返回监听状态。

- 其他：代表用户进行了模型操作，但此时客户端仍需先判断是否已经与服务器建立了一个网络连接，即是否选中了一个模型。如果没有选中模型则任何模型操作都没有意义，将返回监听状态。如果已经建立了网络连接，则再根据输出类别标识符进行处理：
  - 1：代表用户进行了缩放操作，客户端发送一个放缩指令和对应的视点参数。
  - 2：代表用户进行了旋转操作，客户端发送一个旋转指令和对应的视点参数。

  之后，客户端更新客户端视点参数，返回监听状态。

当监听到服务器发送的网络响应时，先解析响应报文得到结果图像信息，再调用显示模块跟新缓存并刷新屏幕画面，返回监听状态。

当监听到用户退出时，终止与服务器应用程序的连接，程序结束。

4.3 本章小结

本章在第三章的需求分析和概要设计基础上，对本文提出的基于用户交互预测的虚拟文物展远程渲染系统进行了详细设计。在服务器中运用了第四章提出的基于轨迹分割的分段预测算法，设计了包括渲染模块和预测模块的服务器应用程序。在客户端设计了包括交互捕捉模块和显示模块的客户端应用程序。

第五章　基于轨迹分割的分段预测算法

本章提出了一种基于轨迹分割的分段预测算法，它主要用于在远程渲染系统服务器应用程序预测模块中，对用户的旋转操作进行预测。通过使用该算法，远程渲染系统可以根据用户的历史操作，预测出用户下一个可能的操作，从而提前渲染操作后的结果图像，减少系统的交互延迟。

本章将首先对用户的交互操作给出规范性定义，之后介绍用户旋转操作中可能产生的突兀转折以及对应的轨迹分割方法，最后详细介绍算法的工作流程。

5.1　用户操作定义

为了更准确的描述用户的交互动作，定义用户操作如下：

视点：与 2.2.1 章节中视点概念相同，指用户观察模型时所处的位置和视线方向。

旋转操作：定义旋转操作为从用户按下鼠标左键开始，到抬起鼠标左键这段时间内移动鼠标位置的操作。其中定义鼠标向屏幕坐标系 x 轴正方向移动为视点在视点坐标系 x 轴和 z 轴组成的平面上（视点水平面），以模型中心点为中心，模型中心点到视点的距离为半径顺时针（向视点坐标系 y 轴负方向看去时）旋转，等价于模型沿与视点水平面平行的方向逆时针（向视点坐标系 y 轴负方向看去时）旋转；鼠标向屏幕坐标系 y 轴正方向移动为视点在视点坐标系 y 轴与 z 轴组成的平面上（视点垂直面），以模型中心点为中心，模型中心点到视点的距离为半径顺时针（向视点坐标系 x 轴负方向看去时）旋转，等价于模型沿与视点垂直面平行的方向逆时针（向视点坐标系 x 轴负方向看去时）旋转。

放缩操作：定义放缩操作为用户旋转鼠标滚轮的操作。其中定义滚轮向上滚动为视点向视点坐标系 z 轴正方向平移（即模型放大），滚轮向下滚动为视点向视点坐标系 z 轴负方向平移（即模型缩小）。

根据上述定义，算法使用量化的方式重新定义了 3.2 中提出的“视点参数”，用于描述模型变换，包括两个数值：

旋转角度(rx, ry)：定义旋转角度为视点相对于初始位置，在初始位置视点坐标系视点水平面、视点垂直面方向旋转过的角度。体现了模型旋转过的角度，其取值范围为[0, 360)。

缩放等级 zpos：定义缩放等级为视点与模型中心点的距离。体现了模型放缩的程度，数值越大视点距离模型中心点越近，其取值范围为[1, 10]。

5.2 轨迹分割

5.1 中给出了对旋转操作的定义。但在实际操作中，用户会有鼠标按键按下而不抬起的习惯。在一次旋转操作后，用户可能会在不松开按键的情况下进行下一次不同目的的旋转操作，而这种操作通常会带来视点运动轨迹的突兀转折[20]。有突兀转折的轨迹会导致预测算法的准确性大大降低，因此本算法提出了轨迹分割的优化方法。如果轨迹发生了方向上的突兀转折，就以转折点分界，将轨迹分割成两次不同的操作，并对转折后的操作进行预测，如图 5-1 所示：

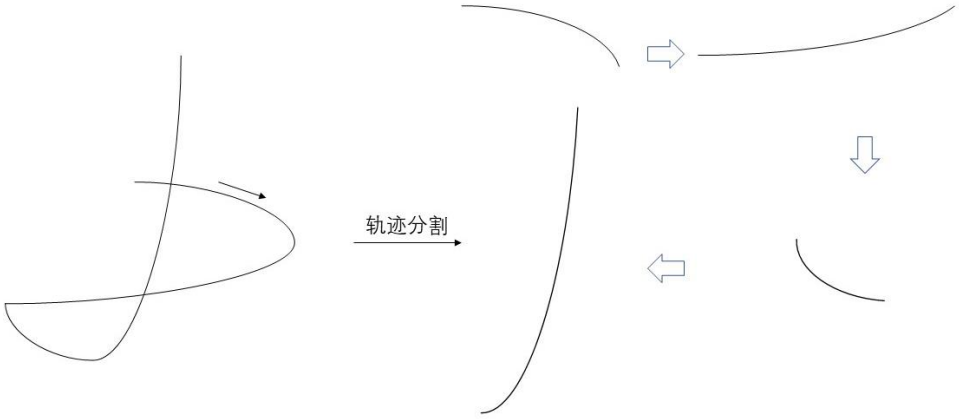


图 5-1 对有突兀转折的轨迹进行轨迹分割

轨迹分割的具体过程如下：设 A(rx_1, ry_1)，B(rx_2, ry_2)，C(rx_3, ry_3)为算法保留的三个视点参数（A 为最早接收到的视点参数），则可求出$\overrightarrow{AB}$、$\overrightarrow{BC}$，设$\overrightarrow{AB}$、$\overrightarrow{BC}$夹角为θ，根据向量数量积的定义：

$$cos\,\theta = \frac{\overrightarrow{AB} \bullet \overrightarrow{BC}}{|\overrightarrow{AB}| \times |\overrightarrow{BC}|} \tag{5-1}$$

可以求出$\cos\theta$的值，若为负数（即$\theta > 90^{\circ}$)则视为存在突兀转折，丢弃 A 视点参数，完成轨迹分割。否则不进行轨迹分割。

5.3 算法流程

算法工作流程如图 5-2 所示：

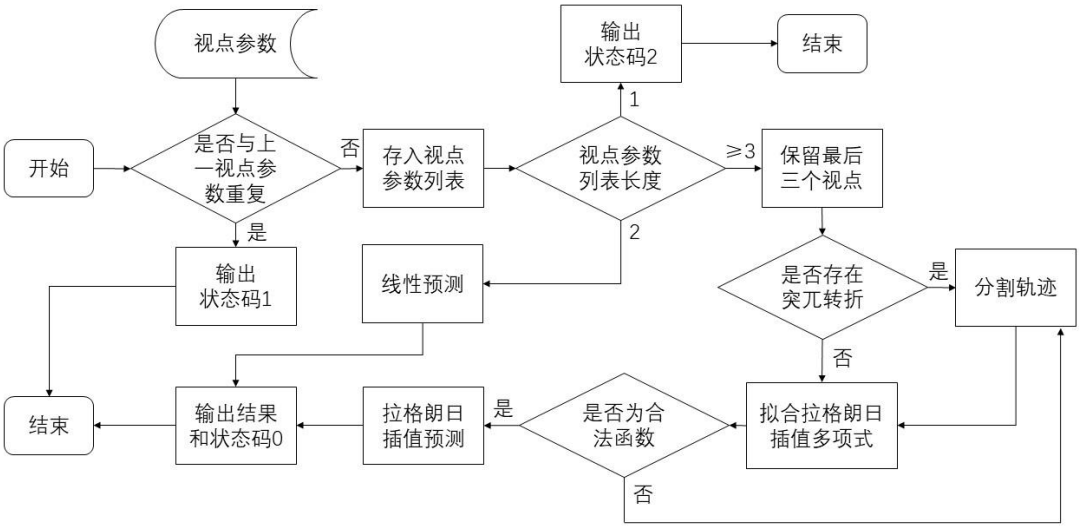


图 5-2 基于轨迹分割的分段预测算法流程图

算法维护了一个视点参数列表 L，按接收时间顺序储存接收到的用户视点参数。算法同时还定义了一个预测步长K_s，代表了预测的下一个视点与当前视点之间的距离。这个距离不是两个视点在世界坐标系中的真实距离，而是使用两视点参数中旋转角度(rx, ry)在二维坐标系中的距离代替这一真实距离。根据 5.1 节中对旋转角度的定义，这一替换是等价的。

算法以用户当前视点参数为输入。当算法收到一个新的视点参数时，为了避免重复预测，首先判断它是否与上一个接收到的视点参数（即 L 末尾的元素）是否相同，如果是相同的则直接结束算法，输出状态码 1。否则，将该视点参数存放在 L 末尾。此时，算法根据 L 中元素个数 n 决定下一步操作。

n=1 时，无法进行预测，算法结束并输出状态码 2。

n=2 时，算法进行线性预测。首先拟合出这两个视点之间的一次函数关系 y = f(x)。之后由两视点旋转角度 rx 的变化量Δrx判断视点运动方向，再根据K_s和Δrx计算得出下一个视点的旋转角度坐标(new_rx, new_ry)，满足公式(5-2)：

$$\begin{cases} new\_ry = f(new\_rx) \\ \sqrt{(new\_rx - last\_rx)^2 + (new\_ry - last\_ry)^2} = K_s \\ (new\_rx - last\_rx) * \Delta rx > 0 \end{cases} \tag{5-2}$$

其中，$(last_rx, last_ry)$为 L 末尾视点的旋转角度。算法结束并输出(new_rx, new_ry)和状态码 0。

n >= 3 时，为了简化预测过程，同时避免预测中产生过拟合现象，算法只保留最后接收到的 3 个视点参数，将多余参数丢弃。然后对保留的三个视点参数进行 4.2 中提出的轨迹分割过程。轨迹分割过后，根据 2.4 章节介绍的拉格朗日插

值法拟合拉格朗日插值多项式函数 y=g(x)。之后由列表 L 末尾的两视点旋转角度 rx 的变化量Δrx判断视点运动方向，再根据K_s和Δrx计算得出下一个视点的旋转角度坐标(new_rx, new_ry)，满足公式(5-3)：

$$\begin{cases} \text{new\_rx} = g(new\_ry) \\ \sqrt{(new\_rx - last\_rx)^2 + (new\_ry - last\_ry)^2} = K_s \\ (new\_rx - last\_rx) * \Delta rx > 0 \end{cases} \tag{5-3}$$

其中，$(last_rx, last_ry)$为 L 中末尾视点的旋转角度。算法结束并输出(new_rx, new_ry)和状态码 0。

但是存在一种特殊情况，算法在拟合出 g(x)时，可能无法产生合法的函数表达式，图 5-3 就是一个例子。图 5-3 的轨迹不存在突兀转折，但不能用一个函数表达式来表示，因为对于某个自变量 x 存在多个对应的函数值 y。当出现这种情况时，需要进行轨迹分割，舍弃 L 开头的视点参数（图中为 A 点），再重新拟合 g(x)。

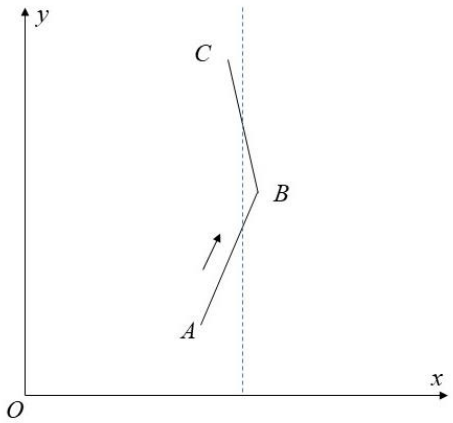


图 5-3　无法进行拉格朗日插值拟合的情况

算法输出的状态码的含义如下：

0：正常退出，进行了预测。

1：视点参数重复，未进行预测。

2：视点参数的个数不足，无法进行预测。

5.4　本章小结

本章研究了一种基于轨迹分割的分段预测算法，用于预测远程渲染用户交互操作中的旋转操作。定义了用户的操作，将用户的交互操作进行量化表示。分析

了用户使用鼠标操作中因操作习惯而可能出现的突兀转折，并提出了使用轨迹分割的方法处理突兀转折。研究了分段预测算法的基本流程，通过线性预测、拉格朗日插值预测的方法实现算法。

第六章　系统实现与测试

本章主要介绍本系统各个应用程序的实现方法和关键细节以及测试情况。应用程序的工作流程已经在第四章中详细介绍，本章将不再赘述。

6.1　服务器实现

本系统的服务器应用程序及其各个模块采用 python 语言编写。python 作为一种十分流行的解释性编程语言，拥有可读性强、移植性好、免费开源的特点，并且拥有大量的封装好的基础库供开发者使用。对于如本系统服务器应用程序的具有多种功能模块的系统，使用 python 编写可以更加便于各个模块之间的组装。

6.1.1　渲染模块实现

本系统借助 pyopengl、pygame、PIL 等工具包实现了渲染模块。pyopengl 是 OpenGL 规范在 python 上的实现，是实现模型渲染的关键。pygame 是用于电子游戏开发的第三方库，本系统中借助 pygame 的 OpenGL 模型，通过 pygame 的游戏窗口将渲染的模型显示到屏幕上，以获取模型渲染结果的二维图像。PIL 是用于图像处理的第三方库，本系统借助 PIL 实现了 JPEG 图像压缩算法。

程序先读取 obj 和 mtl 文件，逐行分析并根据关键字提取数据，将它们存储起来。之后启动 pygame 窗口，向其中绘制各个顶点和三角形，并对模型的光照、贴图等参数进行设置，最后根据输入的视点参数调整模型位置，刷新缓冲区将图像显示出来。然后从缓冲区捕获图像，压缩后转码，输出转码后的结果字符串。

6.1.2　预测模块实现

本系统借助 scipy 工具包实现了预测模块。scipy 是一个用于数学、科学、工程领域的常第三方库，可以处理插值、积分、优化、图像处理、常微分方程数值解的求解、信号处理等问题。本系统借助了 scipy.interpolate 中的 lagrange 函数构造拉格朗日插值函数。程序使用 python 中的 tuple 表示视点参数中的旋转角度，使用以 tuple 为组成元素的 List 作为预测算法中存储历史视点参数的列表。

6.1.3　服务器应用程序实现

为了实现服务器应用程序模块调用和网络请求的功能，本系统使用 socket

编程实现了服务器应用程序，并将其部署在了 Django 框架上。Django 是一种广泛使用的 python web 框架，用于处理服务器接收到的网络请求。socket 编程则通过监听传输层 TCP 连接以更加方便的处理连接请求。

Django 框架持续监听来自客户端的指令，当一个客户端发送了连接建立指令是，启动一个新线程运行服务器应用程序。服务器应用程序启动后通过监听 socket 端口，接收客户端发送的模型名称、视点参数等数据，并调用渲染与预测模块进行处理。

6.2 客户端实现

为了满足 3.1.3 中“瘦客户端”的需求，本系统的客户端使用 web 页面实现。用户设备只需连接网络，就可以通过浏览器访问 IP 地址得到客户端页面。web 页面采用 HTML+CSS+JavaScript 编程语言编写，使用 JQuery 框架实现鼠标操作捕捉、页面元素更新等操作，并采用 ajax 和 fetch 完成网络请求。

客户端界面如图 6-1 所示：

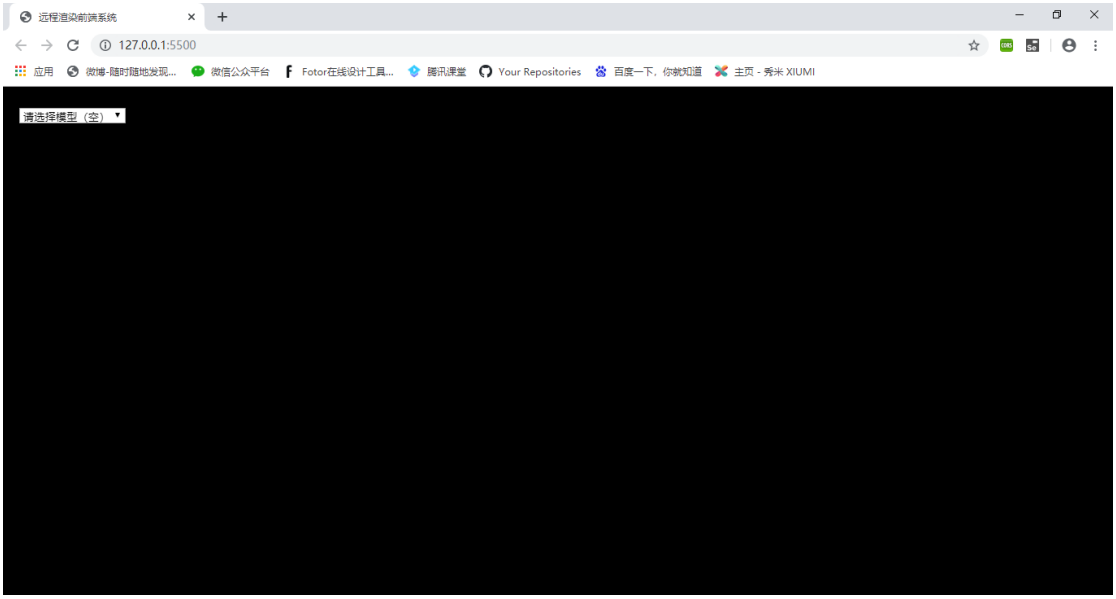


图 6-1 客户端界面

用户进入系统后，不显示任何图像。点击左上角的模型选择下拉菜单，可以选择对应的模型，选择模型后会显示该模型的初始图像，如图 6-2 所示：

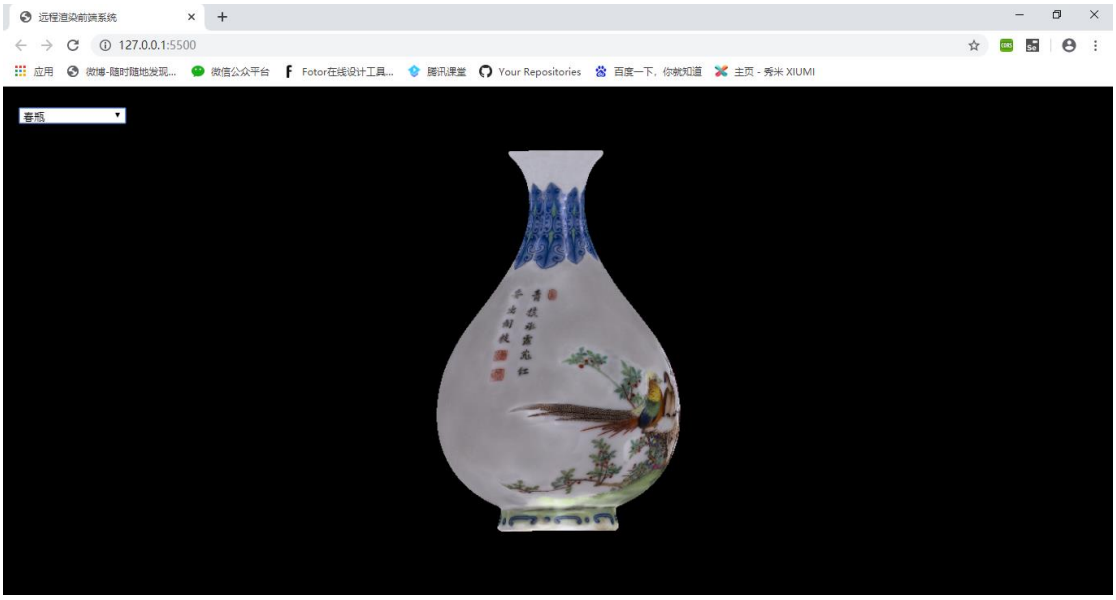


图 6-2　客户端显示模型初始图像

在任意时刻，用户可以再次更换模型。用户按下鼠标按键并移动鼠标可以进行旋转操作。旋转后的效果如图 6-3 所示：

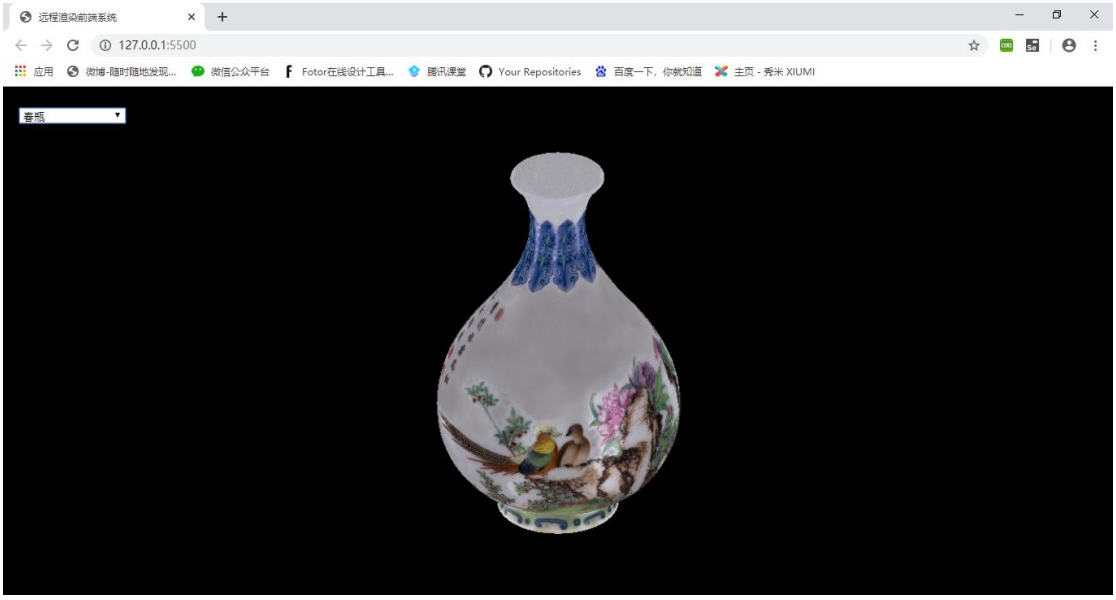


图 6-3　客户端显示旋转操作后的图像

用户滚动鼠标滚轮可以进行放缩操作。放缩后的效果如图 6-4 所示：

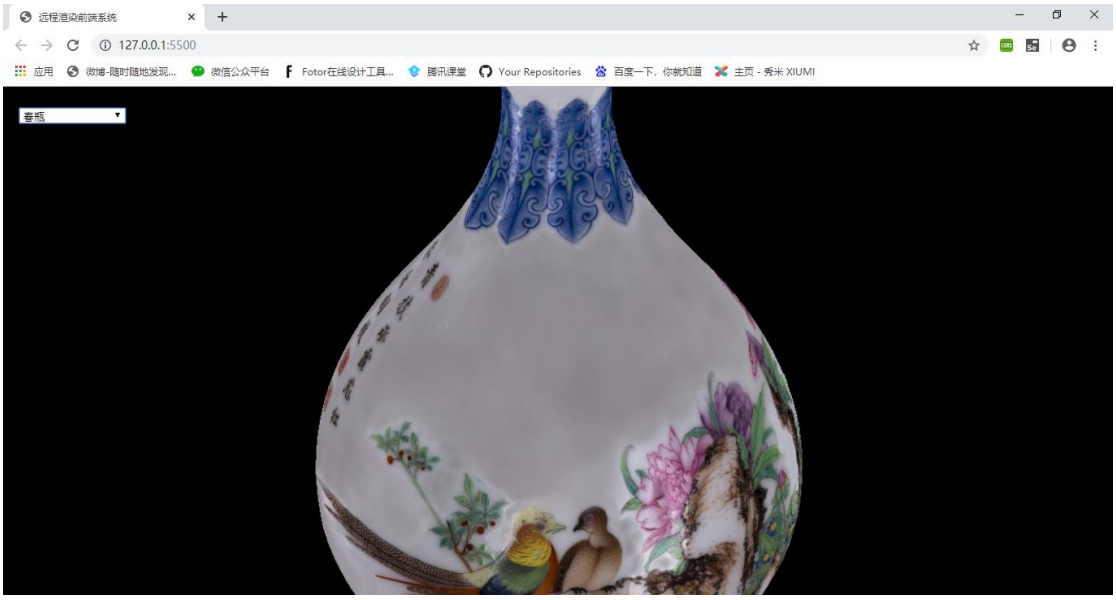


图 6-4　客户端显示放缩操作后的图像

6.3　系统测试

6.3.1　测试环境

测试环境如表 6-1 所示：

表 6-1　测试环境参数表

参数名	值
设备名称	Dell Inspiron 5468 笔记本电脑
处理器	Intel Core i7-7500U CPU @ 2.70GHz 2.90GHz
图形卡	AMD Radeon R7 M340
图形内存大小	2GB
物理内存大小	12GB
网络带宽	上行100Mbps，下行80Mbps
操作系统	Windows 10（64位）专业版
浏览器	Chrome 78.0.3904.108

6.3.2　渲染结果测试

在上述测试环境下，首先对渲染结果进行了测试。分别使用文物“太保鼎”的两个不同分辨率的模型数据：10 万个三角形级别的模型数据（99912 个三角形）和 1000 万个三角形级别的模型数据（10389490 个三角形）进行渲染，比较渲染

结果的差异，如图 6-5 所示。可以看出，1000 万个三角形级别的模型数据渲染的模型精细度显著高于 10 万个三角形级别的模型数据。

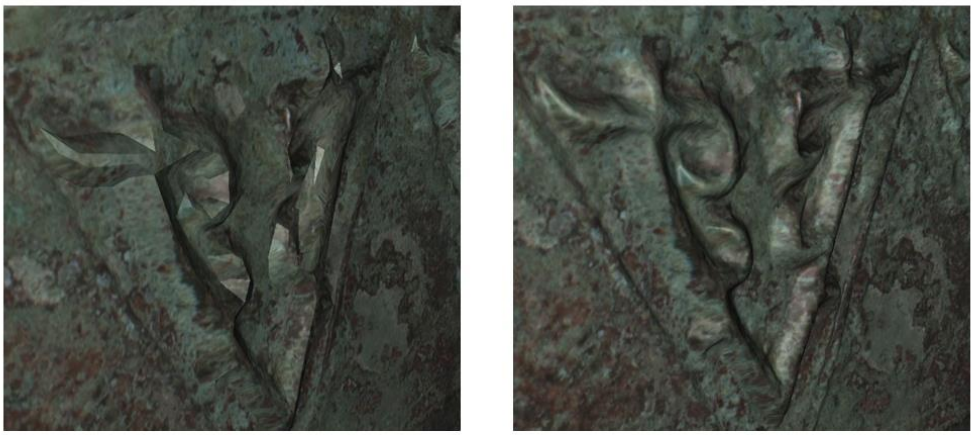


图 6-5　不同分辨率的太保鼎渲染结果（局部）
(a)　10 万三角形级别；(b)　1000 万三角形级别

6.3.3　预测准确度测试

之后，对于预测算法的准确性进行了测试，分别记录 100 次旋转操作中使用不带轨迹分割的分段预测算法和使用带轨迹分割的分段预测算法命中次数、未命中次数。实验结果如表 6-2 所示：

表 6-2　预测算法命中率测试

测试条件	预测命中次数	命中率
不使用轨迹分割	51	51%
使用轨迹分割	73	73%

根据实验结果可以看出，本文提出的基于轨迹分割的交互预测算法命中率可以达到 73%，能够较为准确的预测用户的旋转操作，并且轨迹分割也能显著提升算法预测的正确率。

对于预测未命中的情况，经分析有以下四点原因：

1) 由于预测算法是根据历史视点位置拟合出具有函数关系的轨迹曲线，而用户的实际操作轨迹与拟合出的曲线不完全相同。因此使用拉格朗日插值预测法拟合出的拉格朗日插值函数曲线，与真实操作轨迹存在一定误差，导致预测出现误差。

2) 预测阈值选择不够精确，用户的操作速度直接影响了视点在轨迹上移动的步长。而系统采用的是固定阈值，当用户的操作速度发生变化时，很容易导致较大误差。如用户的操作速度变慢时，同一时间间隔在轨迹曲线上移动的距离减小，此时如果阈值不变，则预测的点就会比实际点在轨迹曲线上更远。
3) 冷未命中。当程序刚启动时，或刚进行完轨迹分割之后，此时历史视点的个数较少，无法进行预测或只能进行线性预测，因此会导致预测未命中的发生。随着历史视点变多，冷未命中问题会得到缓解。
4) 在进行拉格朗日插值预测出现的特殊情况中（4.3.1 末尾提出的情况），算法在用户轨迹不存在突兀转折的情况下，由于无法拟合出合法的拉格朗日插值函数，因此进行了轨迹分割。这会导致算法由拉格朗日插值预测变为线性预测，从而降低了预测的命中率。

6.3.4 交互延迟测试

对系统的交互延迟进行测试，选取同一文物不同分辨率级别的模型数据。分别记录 100 次旋转操作中不使用预测算法、不使用轨迹分割的分段预测算法和使用轨迹分割的分段预测算法的平均交互延迟，结果如表 6-3 所示：

表 6-3 交互延迟测试

测试条件	10 万三角形	20 万三角形	100 万三角形
不使用预测算法	1.52 秒	1.60 秒	2.21 秒
不使用轨迹分割	0.71秒	0.77秒	1.59秒
使用轨迹分割	0.49秒	0.52秒	0.91秒

根据实验结果可以看出，本文的预测算法显著降低了平均交互延迟。在使用带轨迹分割的预测算法的前提下，本系统能将不同分辨率的模型交互延迟保持在 1 秒以内，满足了性能的要求。

6.4 本章小结

本章根据第五章对系统的详细设计，介绍了实现系统使用的关键技术，包括使用的编程语言、框架、函数和库。之后对系统进行了测试，比对了不同分辨率模型的渲染结果的差异，验证了基于轨迹分割的分段预测算法的准确性，分析了算法未命中可能的原因，测试了系统在不同情况下的交互延迟。根据实验结果，

算法的预测准确性较好，交互延迟处于合理范围内，符合用户需求。

第七章　总结与展望

针对虚拟文物展远程渲染的需求，本文主要研究了远程渲染的相关技术和实现方法，设计并实现了一套基于用户交互预测的虚拟文物展远程渲染系统。具体内容如下：

1) 本文研究了远程渲染系统的分类和国内外研究现状，分析了远程渲染系统的基本架构，提出了远程渲染系统中普遍存在的交互延迟问题，研究了交互延迟的构成和减少方法。本文还对三维渲染、图像压缩和预测的方法进行了研究。
2) 本文提出了一种提出了基于轨迹分割的分段预测算法。该算法通过定义用户操作，得出了视点参数的概念，并维护了用户的历史视点参数列表。通过对历史视点轨迹的突兀转折进行轨迹分割后，使用基于线性预测和拉格朗日插值预测的分段预测方法预测下一个视点位置。
3) 本文分析了虚拟文物展的用户需求，并根据需求设计了采用客户端-服务器架构的基于用户交互预测的虚拟文物展远程渲染系统。系统服务器应用程序使用 Python 实现，包括预测模块和渲染模块；系统客户端应用程序使用基于 JQuery 的 web 实现，包括交互捕捉模块和显示模块。客户端捕捉用户的旋转、放缩等操作，将操作指令和视点参数发送给服务器，服务器根据视点参数渲染产生结果图像发回给客户端。同时服务器预测用户旋转操作的下一步操作，提前渲染预测的结果图像，将预测的图像发送给客户端缓存中存储起来。在缓存命中的情况下，客户端通过预测缓存将画面更新延迟降低到客户端读取缓存所花费的时间，从而大大降低了交互延迟，改善了用户体验。

通过本文的提出的系统，虚拟文物展的用户可以获得较好的用户体验，但是本系统还存在着诸多不足，在以下几个方面还有可改进的地方。

1) 预测算法的准确率仍然有待提升。根据实验结果，本文提出的基于轨迹分割的分段预测算法准确率只有 70%左右，这仍然不能满足最好的用户体验需求。为了实现简化预测流程，本文使用的轨迹分割方法十分简单。只根据了向量夹角去判断突兀转折，并使用线性和拉格朗日插值方法进行预测。但实际上用户的旋转操作轨迹更加复杂，这种算法在一些情况下无法准确预测，因此在预测算法的改进上可以进行进一步研究。

2) 卡顿问题仍然存在。当缓存未命中时，系统会花费较长时间渲染当前帧，这会导致画面的短时间卡顿和操作上的停滞，影响用户体验。无论如何优化预测算法，都不能完全避免缓存未命中的产生，因此需要研究其它方式来减少卡顿带来的影响。Shi 等人提出了一种 3D warping 的算法[15]，根据一些预设的固定视点的模型深度图来合成其它视点的图像。可以利用这一算法，在缓存未命中时使用 3D warping 合成的图像更新画面，能够将卡顿时间减少到运行该算法花费的时间。
3) 图像压缩算法可以继续改进。本系统中仅仅使用了经典的 JPEG 压缩算法，它是一种有损压缩算法，仍然会对图像清晰度产生影响，尤其是在源图像是高精度模型（如百万三角形级别）时，会丢失大量细节。因此下一步的研究还可以进一步优化图像压缩算法，在保证算法效率的前提下减少对图像清晰度的损伤。

参考文献

[1] Levoy M, Pulli K, Curless B, et al. The digital Michelangelo project: 3D scanning of large statues[C]//Proceedings of the 27th annual conference on Computer graphics and interactive techniques. 2000: 131-144.

[2] 金平, 张海东, 齐越, 等. 基于远程渲染的三维模型发布系统[J]. 北京航空航天大学学报, 2006, 32(3): 337-341.

[3] Liu Gang，Zhang Jun，Diao Changyu．Study 3D Digitization Techniques Applying to Dunhuang Caves[J]．DunhuangResearch，2005，(4)：1 04—109．

[4] Shi S , Hsu C H . A Survey of Interactive Remote Rendering Systems[M]. ACM, 2015.

[5] Zhao Z, Hwang K, Villeta J. Game cloud design with virtualized CPU/GPU servers and initial performance results[C]//Proceedings of the 3rd workshop on Scientific Cloud Computing. 2012: 23-30.

[6] Koller D, Turitzin M, Levoy M, et al. Protected interactive 3D graphics via remote rendering[J]. ACM Transactions on Graphics (TOG), 2004, 23(3): 695-703.

[7] Singhal S . Networked virtual environments designed and implementation[J]. ACM Press. SIGGRAPH Series, 1999.

[8] Bethel W . Visapult: A Prototype Remote and Distributed Visualization Application and Framework. office of scientific & technical information technical reports, 2000.

[9] Engel K , Hastreiter P , Tomandl B , et al. Combining Local and Remote Visualization Techniques for Interactive Volume Rendering in Medical Applications. IEEE, 2000.

[10] Prohaska S , Hutanu A , R Kähler, et al. ABSTRACT Interactive Exploration of Large Remote Micro-CT Scans[J]. IEEE Visualization, 2008:345-352.

[11] 应超. 博物馆移动导览中的远程展示技术研究及系统实现[D]. 浙江大学, 2015.

[12] Zy A , Hsw B , Hong P A , et al. ASM: An adaptive simplification method for 3D point-based models - ScienceDirect[J]. Computer-Aided Design, 2010, 42(7):598-612.

[13] Levoy M . Polygon-assisted JPEG and MPEG compression of synthetic images. 1995.

[14] Boukerche A , Pazzi R . Remote rendering and streaming of progressive panoramas for mobile devices[C]// Acm International Conference on Multimedia. ACM, 2006.

[15] Shi S, Nahrstedt K, Campbell R. A real-time remote rendering system for interactive mobile graphics[J]. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 2012, 8(3s): 1-20.

[16] Mark W R . Post-Rendering 3D Image Warping: Visibility, Reconstruction, and Performance for Depth-Image Warping. The University of North Carolina at Chapel Hill, 1999.

[17] 郑斌. 面向手持终端的复杂三维场景远程渲染系统设计与实现[D]. 浙江大学, 2014.

[18] Wikipedia. “OpenGL”. [EB/OL].http://en.wikipedia.org/wiki/OpenGL

[19] 王金峰, 姚国清. 三维模型文件中的 OBJ 格式在 OpenGL 中的输入与处理[J]. 电脑知识与技术, 2011(10):2393-2396.

[20] 应超. 博物馆移动导览中的远程展示技术研究及系统实现[D]. 浙江大学, 2015.

致　谢

时光荏苒，转眼间我的大学四年生活就要接近尾声了。随着毕业的临近，我不由得感叹：时间过得真是太快了，大一入学的场景还还历历在目，而现在同学们就要各奔东西。这四年里，我从青涩走向成熟，从迷茫走向坚定。过去的近五个月的时间里，我完成了本科学业的最后一个部分——毕业设计。虽然过程有着诸多坎坷，但在许多人的帮助之下，我最终顺利的完成了项目。在本文的最后，我想对曾经帮助我的人一一表示感谢。

首先，我要感谢在大学四年里辛勤教导我的老师和辅导员们。感谢老师们的精心培养，让我在本科四年中学到了如此多的知识，为我打开了通往计算机世界的大门。感谢李庚、葛云丽、胡君颖三位辅导员老师，为我的本科生活保驾护航，在我的学习、生活特别是思想上提供了许多帮助，也让我坚定了要用一生所维护的共产主义信仰。我还要特别感谢我的导师张怡老师，张怡老师是一位性格非常温和的老师，她为人和善，对待学生十分温柔，总是耐心而又细致地解答我的问题。同时她也是一位治学严谨、工作认真负责的老师。她对我的关怀和指导，对我毕设的顺利完成起到了重要作用，在此我表示深深的感谢。

其次，我要感谢我的父母和我的家人。在四年的大学生活中，父母和家人不仅时刻关心我的健康状况，还在我遇到困难时为我排忧解难，是我最坚定的后盾。我尤其要感谢我的母亲和我的父亲。我的母亲勤劳慈祥，是她昼夜操劳将我带大，照顾我的生活和学习。在离开家的大学生活中，她还时刻不忘我的身体状况。在我被压力深深困扰时，也是我的母亲倾听我的心声，不断安慰和鼓励我。我的父亲正直幽默，是他为我的人生保驾护航。在我遇到人生的关键路口时，总是我的父亲为我指明前进的方向。我的父亲为人正直又十分幽默随和，为我的生活带来了许多欢笑与美好。特别是当他作为一线干警，投入到抗击新冠疫情的第一线战斗时，为我树立了宏伟的榜样，也激励着我不断努力。我的父母生我养我，我的家人们关心我爱护我，在此我也要表示深深的感谢。

之后，我要感谢我的三位室友：马天宇、孙翰宇、孙浩铭，在大学四年的生活中能包容我的缺点，给我提供了诸多学习、生活上的帮助，为我的大学生活留下了一段美好的回忆。我特别要感谢孙翰宇在疫情期间对我心理上的疏导和精神上的支持，在那段时间我迷茫与困惑之时，是他帮助我走出了困境。我还要特别感谢孙浩铭在我完成毕设期间对我的支持与陪伴。

我还要感谢青年文化促进会 18 届部长主席团新媒体综合事务部的 7 位同学，让我在青促感觉到了家的温暖，不仅帮我锤炼了组织与沟通能力，还为我提供了各种的帮助，让我的大学社团生活更加丰富多彩。在此之中我还要特别感谢机械工程学院 17 级能源与动力工程的郑瑞凡同学。在从疫情到现在的一年多时间里，他不仅在生活中给我带来了很多欢乐，还在精神上给予了我很大鼓励和支持，支撑我渡过了一道道难关。虽然我们的专业不同，但他也为我的毕设提供了大量帮助，他的勤奋刻苦与钻研精神，也是我不断学习的榜样。

最后，我还要感谢我头顶的璀璨星空。天文学是我从小以来的爱好，我深爱着这片星空。每当我迷茫与退缩之时，特别是在我的毕设遇到瓶颈时，我都会躲在学校里的黑暗角落，仰望天空中那一个个未知的世界，那一颗颗闪光的星星，仿佛在向我诉说着他们的故事。在浩瀚无垠的宇宙中，人类不过是寄住在这颗脆弱蓝色星球上的匆匆过客。自身的渺小和生命的短暂，让人类在浩瀚宇宙面前显得那么无力。但人类又何曾畏惧过这一切，我们拼尽全力钻研科技，飞向太空，只为窥得一角真理，与命运抗争。人类文明注定会灭亡，但在那之前绝对不会停下探索科学的脚步。每当我想到这，心底就会重新燃起斗志，激励着我向着目标继续奋斗。

大学四年，相遇即是缘分。无论是老师、同学、学长学姐、学弟学妹、那些只有一面之缘甚至素未谋面的陌生人，都曾经给予过我的或多或少的帮助。感谢你们出现在我的人生中，让我的大学生活精彩而美好。正如卡尔·萨根所说，在广漠的空间和无限的时间中，能与你同享一颗星球的一段时光，是我的荣幸。这段美好的时光，就是我获得的最好的毕业礼物。

刘兴宇

2021 年 5 月于北洋园