

成

绩



**专业综合设计**

题目名称**酒店自助点餐系统**

学生学院 计算机学院

专业班级 13级软件工程（1）班

学 号 3113006215

学生姓名 陈永坤

指导教师 路 璐

201 7 年 1 月 2 日

**目录**

[1 绪论 1](#_Toc479896737)

[1.1 研究背景 1](#_Toc479896738)

[1.2 人体运动捕捉 1](#_Toc479896739)

[1.2.1 运动捕捉设备 1](#_Toc479896740)

[1.2.2 基于Leap Motion的手势捕捉 1](#_Toc479896741)

[1.3 论文研究意义 1](#_Toc479896742)

[1.4 论文主要内容 1](#_Toc479896743)

[2 Leap Motion 1](#_Toc479896744)

[2.1 硬件设备 1](#_Toc479896745)

[2.2 多角成像技术 2](#_Toc479896746)

[2.3 坐标系统 2](#_Toc479896747)

[2.4 运动数据追踪 3](#_Toc479896748)

[2.4.1 手部模型 3](#_Toc479896749)

[3 Unity 6](#_Toc479896750)

[3.1 Unity开发引擎 6](#_Toc479896751)

[3.2 Visual Studio集成开发环境 7](#_Toc479896752)

[4 Leap Motion应用设计 7](#_Toc479896753)

[4.1 主题设计 7](#_Toc479896754)

[4.2 基本功能设计 7](#_Toc479896755)

[4.3 构建3D场景 7](#_Toc479896756)

[4.4 界面设计 7](#_Toc479896757)

[5 详细设计与编码 7](#_Toc479896758)

[5.1 集成Leap Motion 7](#_Toc479896759)

[5.1.1 Leap Motion插件资源 7](#_Toc479896760)

[5.1.2 主要API 7](#_Toc479896761)

[5.2 代码整体框架搭建 7](#_Toc479896762)

[5.2.1 MVC框架简介 7](#_Toc479896763)

[5.2.2 框架的实现思路 7](#_Toc479896764)

[5.3 全局管理器 7](#_Toc479896765)

[5.4 应用场景模块实现 7](#_Toc479896766)

[5.4.1 菜单场景 7](#_Toc479896767)

[5.5 主场景 7](#_Toc479896768)

[5.6 资源打包与发布 8](#_Toc479896769)

[6 手势交互操作与测试 8](#_Toc479896770)

[6.1 Leap Motion驱动参数调整 8](#_Toc479896771)

[6.2 交互测试和结果分析 8](#_Toc479896772)

[参考文献 9](#_Toc479896773)

# 绪论

## 研究背景

## 人体运动捕捉

### 运动捕捉设备

### 基于Leap Motion的手势捕捉

## 论文研究意义

## 论文主要内容

# Leap Motion

Leap Motion是一款由美国Leap公司所出的专用于跟踪、捕获人体手部运动数据的体感控制器，它提供一种能够识别手势动作并进行手势交互的解决方案。[基于Leap Motion 的三维识别方法]

## 硬件设备

Leap Motion是一个基于双目视觉的小型手势识别设备，配合一条专用的USB数据线，将其连接到PC机或虚拟现实头盔，即可实时反馈传感器所捕获并构建的手部数据帧到处理器。



## 多角成像技术

Leap Motion采用基于双目视觉的多角成像技术，在设备上安装有3个LED红外光源和2个灰阶摄像头传感器，可以一次性感知所测物体的所有像素，并使用预置算法进行处理。[基于Leap Motion的数字手势识别]

具体过程是：两个摄像头对当前环境进行拍摄，得到两张具有不同视角的照片，再根据摄像头的各项已知参数和相对位置，以及相同物体在画面中的不同位置，计算出物体实际距离摄像头的深度。[带你了解世界最先进的手势识别技术 -- 微软，凌感，Leap...]

Leap Motion在形成深度图像时，仅采用若干个特征点进行手型匹配，在获得这些特征点的信息后，利用IK算法即可推算出整个手部模型。由于避免了匹配手部所以点，所以Leap Motion的处理时间能够减小到10ms以内。[深度：Leap Motion手势识别大揭秘]

## 坐标系统

Leap Motion传感器以其中心为原点建立一个右手笛卡尔坐标系，X轴与传感器长边平行，指向右方，Y轴垂直屏幕指向上方，Z轴背离屏幕指向操作者的方向。若将传感器与头戴设备连接，则Y轴指向前方，Z轴垂直向上。

Leap Motion传感器的探测视野在屏幕上方形成一个倒四棱锥空间结构，视角约为150°，有效范围约为0.3米到0.6米，检测精度可达到毫米级。



## 运动数据追踪

Leap Motion所追踪的具体对象为手、手指、手势以及工具，设备定期发送所采集到的目标对象的方向、位置等信息，每份这样的信息都被保存到帧（frame）当中，并为每个被其检测到的对象分配唯一ID，只要设备能够探测到对象，ID便保持不变。假如设备失去目标，而当目标再次出现时，将重新分配新的ID。

Leap Motion视野中的实体发生位移、旋转等变化，都会引起帧信息的变化，通过将当前帧与之前帧的数据进行对比，形成运动信息。

### 手部模型

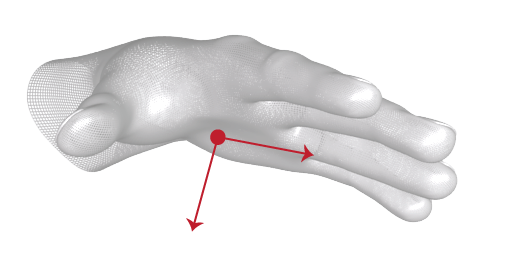
手部模型提供被检测手的ID、位置等其他信息，包括连接这只手的手指和手臂等等。

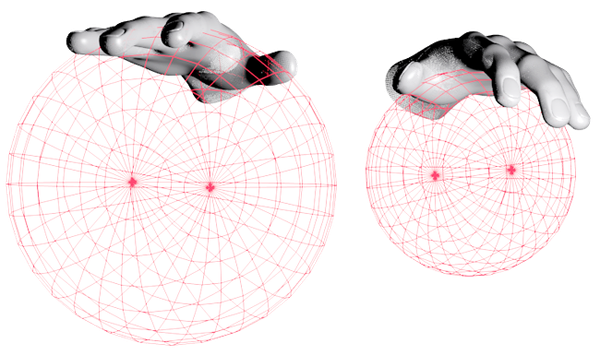
即使在部分手探测不到的情况下，Leap Motion也能结合可见部分、内部手部模型以及先前的运动信息来预判该不可见部分接下来最有可能的跟踪数据。[基于LeapMotion与Unity3D的体感游戏“Survial&Shoot”的开发]

1. 手

手对象包含用于描述一只手物理特征的各种属性。

|  |  |
| --- | --- |
| Palm Position | 手掌中心的坐标 |
| Palm Velocity | 手掌运动的速度 |
| Palm Normal | 垂直于手掌平面的向量 |
| Direction | 从掌心指向手指的向量 |
| Sphere Center | 适应手掌弧面的一个球的球心 |
| Sphere Radius | 适应手掌弧面的一个球的半径 |





1. 手指

Leap Motion可识别手上每根手指的信息。手指被控制器分为拇指、食指、中指、无名指和小拇指，如果某根手指无法被检测到，Leap Motion会根据最近观察数据与手掌的解剖模型推测出该手指的特征。

1. 工具

Leap Motion除了可以检测手指外，也可以检测手持的工具，工具被认为是比手指更细、更长而且更直的物件。



手指与工具具有一些相同的属性，Leap Motion将它们统一称为端点对象（Pointable Object）。

|  |  |
| --- | --- |
| Length | 可见长度 |
| Width | 平均宽度 |
| Direction | 与物体指向相同的单位向量 |
| Tip Position | Leap Motion坐标系下的位置 |
| Tip Velocity | 运动速率 |



# Unity

## Unity开发引擎

Unity是由丹麦Unity Technologies公司所开发的一款完整的跨平台游戏开发引擎，可以制作出3D或2D的优秀游戏，其编辑器可运行在Windows和Mac OS X下，并支持包括PC，IOS，Android，Web，XBOX等多个平台发布。

Unity提供易于上手的操作界面，内置功能强大的UI系统，并支持物理引擎、多人网络连线、地形处理和角色动作编辑等各项功能，可大幅度降低游戏的开发门槛，缩短游戏的开发时间，并达到降低制作成本的目的，目前已有80%的手机游戏使用了Unity引擎作为开发工具。

Unity不仅只限于游戏行业，在3D设计、工程模拟和虚拟现实等方面也有着广泛的使用。国内常用Unity开发虚拟现实三维展示甚至是军事演练模拟等项目。



## Visual Studio集成开发环境

Microsoft Visual Studio（以下简称VS）是美国微软公司所推出的一套完整的开发工具集，其包含了软件生命周期所需要的大部分工具，如集成开发环境、代码管理工具和UML工具等，其发布出来的产品适用于绝大部分平台，包括Windows、Windows Phone、Android、IOS、Web。

Visual Studio是当前Windows平台下最流行、最强大的集成开发环境，并可通过安装扩展插件进行更多功能的定制。目前，VS通过自带扩展便可与Unity引擎无缝对接，并支持断点，成为取代MonoDevelop编辑器的强大利器。



# Leap Motion应用设计

## 主题设计

## 基本功能设计

## 构建3D场景

## 界面设计

# 详细设计与编码

## 集成Leap Motion

Leap Motion提供了适用于Unity应用开发的资源包，其中包括了核心代码脚本和通用模型，通过导入并使用插件资源，可以快速构建出Leap Motion应用。

### Leap Motion插件资源

Leap Motion插件资源包括了以下五个部分：

Core Assets：核心功能包，是使用Unity开发Leap Motion应用的必要插件。

Interaction Engine：交互引擎模块，是一套支持与场景中物体进行交互的更简便的方案，例如进行抓取、抛掷等动作，要求手部与物体直接接触。

Attachments Module：用于控制场景中物体或UI各项属性变换的功能模块，如旋转、缩放、颜色渐变或等，不要求手部与物体直接接触，符合控制条件即可进行变换。

Hands Module：手型资源包，内含Leap Motion提供的一些手部模型，包括可见的骨骼驱动的人手、程序生成的多边形手、组件构成的机械手以及不可见的物理手等。

Detection Examples：使用Detection Utilities（核心包中监测具体动作行为的脚本集）开发的Demo。

本应用开发过程中将会使用到核心包、交互引擎及手型资源包，其余资源可作为参考下载。

下载地址：https://developer.leapmotion.com/unity

### 主要API

LeapServiceProvider：LeapProvider唯一的具体类，其中保留着由传感器传输到应用的帧数据。

|  |  |
| --- | --- |
| 方法或属性 | 解释 |
| Controller GetLeapController () | 返回Leap Controller实例 |
| bool IsConnected () | 是否已正确连接Leap Motion设备 |
| bool \_isHeadMounted | 是否头戴式 |
| Frame CurrentFrame | 当前帧数据 |

HandPool：手部模型的对象池，包含一个ModelGroup列表，可配置多种不同的手部模型，它们将会被同时激活或禁用。

ModelGroup：一组手部模型，包括左手和右手。

|  |  |
| --- | --- |
| 方法或属性 | 解释 |
| string GroupName | 组别名称 |
| bool CanDuplicate | 是否可以复制新的实例并使用 |
| bool IsEnabled | 是否启用这组手模型 |
| IHandModel LeftModel | 左手模型 |
| IHandModel RightModel | 右手模型 |

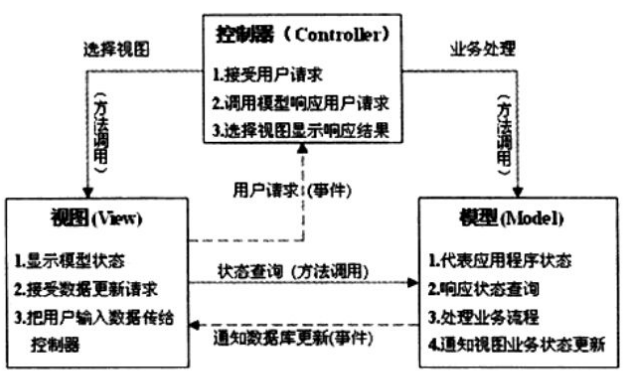
InteractionManager：交互引擎管理器，用于设置接触式交互的各项参数。

|  |  |
| --- | --- |
| 方法或属性 | 解释 |
| LeapProvider \_leapProvider | 所使用的LeapProvider实例 |
| HandPool \_handPool | 所使用的HandPool实例 |
| string \_ldatPath | 位于StreamingAssets文件夹下的Ldat文件路径 |
| string \_brushGroupName | HandPool下BrushHand手模型所在组别名 |
| bool \_contactEnabled | 是否启用接触，禁用将无法对添加了InteractionBehaviour组件的物体产生作用 |
| bool \_autoGenerateLayers | 是否自动创建交互层 |

## 代码整体框架搭建

### MVC框架简介

MVC是一种软件的架构模式。它把软件系统划分成模型（Model）、视图（View）和控制器（Controller）三个部分，从而将应用程序的输入、处理和输出分离开来。



Model：存储应用程序的核心数据。

View：用户界面，显示Model数据。

Controller：处理用户输入。

Model、View和Controller，再加上用户，则成为了程序流程的一个闭环。用户通过View进行输入操作，向Controller发出请求事件，Controller根据用户请求调用相应方法更改Model数据和状态，Model发生变更后通过事件去通知相关的View更新界面。这样，便完成了从用户输入开始到数据处理与存储再到返回请求结果的整个过程。

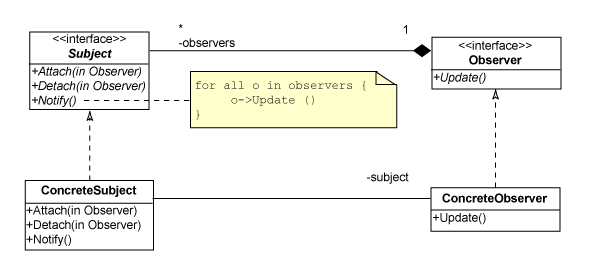
在本程序开发中使用MVC模式，旨在提高代码的复用性和可维护性，同时，会进行一定程度的优化以减少使用该模式所带来的负面作用。

### 观察者模式

模型存储着系统数据，当模型中的数据发生变更时，需要通知视图进行更新。因此，模型宜采用观察者模式实现通知。

观察者模式，又称为订阅-发布模式或模型-视图模式，是常用的软件设计模式之一。当被观察对象的状态发生变更时，会去通知它的所有观察者进行更新。

观察者模式常用“注册——通知——撤销注册”的形式实现。

IMG_256

被观察对象保留一份观察者列表，一个被观察对象可以有多个观察者，观察者将自身注册到被观察对象当中。当被观察对象发生了某种变化后，遍历列表内所有已注册的观察者，通知它们更新。

### 框架实现思路

实现MVC框架时结合观察者模式，用户界面作为观察者，而业务数据则是被观察对象。利用C#委托事件的特性，能够简化观察者模式的设计。通过实现接口方法，完成对事件的注册、调用和注销。由于C#类实例方法只归属于类的具体实例，指定了实例方法便指定了某个实例，一个委托事件中可注册多个具体实例的多个实例方法，因此，被观察对象可只保留一个事件清单，不必保留观察者列表，便能根据完成对多个观察者（即实例方法的归属对象）进行特定事件的通知。

## 全局管理器

## 应用场景模块实现

### 菜单场景

## 主场景

## 资源打包与发布

# 手势交互操作与测试

## Leap Motion驱动参数调整

## 交互测试和结果分析

参考文献

[1] 王珊，萨师煊. 数据库系统概论（第4版）[M]. 高等教育出版社. 2006-05

[2] 谭云杰. 大象：Thinking in UML（第2版）[M]. 中国水利水电出版社. 2012-03

[3] Jeffrey Richter. CLR via C# 4th Edition [M]. Microsoft Press. 2012-12

[4] Bruce Johnson. Visual Studio 2015高级编程（第6版）[M]. 清华大学出版社. 2016-05

[5] M.Minasi, Kevin, Greene, Christian. 精通Windows Server 2012 R2（第5版）[M]. 清华大学出版社. 2015-04