

电子科技大学  
UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

# 硕士学位论文

MASTER DISSERTATION



论文题目: 移动增强现实技术的应用研究

学科专业: 计算机应用技术

指导教师: 陈雷霆

作者姓名: 周一舟

班学号: 200620603011

分类号\_\_\_\_\_ 密级\_\_\_\_\_

UDC\_\_\_\_\_

# 学 位 论 文

## 移动增强现实技术的应用研究

(题名和副题名)

周一舟

(作者姓名)

指导教师姓名 陈雷霆 教授

电子科技大学 成都

(职务、职称、学位、单位名称及地址)

申请学位级别 硕士 专业名称 计算机应用技术

论文提交日期 2011.04.20 论文答辩日期 2011.05.19

学位授予单位和日期 电子科技大学

答辩委员会主席\_\_\_\_\_

评阅人\_\_\_\_\_

年 月 日

注 1 注明《国际十进分类法 UDC》的类号

## 独 创 性 声 明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其它人已经发表或撰写过的研究成果，也不包含为获得电子科技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。

签名：\_\_\_\_\_ 日期：\_\_\_\_\_ 年 \_\_\_\_\_ 月 \_\_\_\_\_ 日

## 关于论文使用授权的说明

本学位论文作者完全了解电子科技大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权电子科技大学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。

（保密的学位论文在解密后应遵守此规定）

签名：\_\_\_\_\_ 导师签名：\_\_\_\_\_

日期：\_\_\_\_\_ 年 \_\_\_\_\_ 月 \_\_\_\_\_ 日



## 摘 要

增强现实（Augmented Reality-AR），又称混合现实，是将虚拟物体与真实环境进行融合的一种技术，具有虚实结合、实时交互、三维注册的特点。增强现实的应用目前还处于萌发阶段，主要应用于一些专业领域，在手机、平板电脑等手持设备上也发布了一些面向公众的增强现实应用软件。随着手持设备的发展，其处理能力和续航能力不断提高，“无处不在的计算”概念越来越多地体现在实际应用中，在手持设备上的增强现实技术应用，即移动增强现实技术，其价值和市场将是巨大的。本文研究的目标是，通过对当今移动增强现实技术应用的研究，根据手持设备的性能，针对地理位置服务应用领域，提出基于传感器增强现实技术的应用解决方案。

本文首先对当今移动增强现实的应用类型进行介绍，概述当今移动增强现实技术的发展状况。其次对基于机器视觉的增强现实技术进行研究，主要利用图像识别和摄像机姿态定位来计算目标物体从真实环境到摄像机图像平面的坐标变换，从而来定位虚拟物体的渲染位置。然后提出两种基于传感器的增强现实方法，主要应用于地理位置服务，一种是利用传感器计算任意目标物体在摄像机图像平面的投影坐标来定位虚拟物体的渲染位置，另一种是利用传感器将三维虚拟路径叠加在摄像机图像平面中的真实道路上。最后，从用户交互设计到系统实现，详细阐述基于传感器增强现实技术的地理位置服务应用系统。该系统运行于 Android 操作系统手持设备上，由用户层、封装层、功能层和网络层组成。

论文中所描述的两种基于传感器的增强现实方法由作者独立设计和实现，其实验效果很好地满足了地理位置服务应用需求。作为发明人，有增强现实应用中的三维注册方法和一种基于真实景象的导航方法两项专利正在申请中，这也是该论文的创新成果。

**关键词：**增强现实；机器视觉；传感器；地理位置服务

## ABSTRACT

Augmented Reality (AR), also called Mixed Reality, is a technology integrating virtual object into real world, with characteristics as combination between virtual and real, interaction in real-time and registration in 3D. Currently the AR application is just beginning, and mainly focused on some professional areas. There are some AR applications which are released on handheld devices such as mobile phone, tablet PC, etc. With development of handheld devices the processing capability and battery life are improved. The concept of ubiquitous computing is reflected in real applications more commonly. The AR on handheld devices is named Mobile AR with huge value and market opportunity. Through researching on the current applications of AR technology and the capability of handheld devices, this paper aims to propose an application solution of sensor-based AR technology which is specific to Localization Basis Service (LBS).

Firstly, the application category and development status of Mobile AR are introduced. Secondly, the machine vision-based AR technology is researched, which computes the coordinates transformation of target object from real world to camera image plane, mainly through image recognition and camera pose estimation, in order to determine the rendering position of virtual object. Thirdly, two sensor-based AR methods are proposed which are mainly specific to LBS. One is to compute the projection coordinate of any target object in camera image plane to determine the rendering position of virtual object by sensors. Another is to overlay the 3D virtual route on the real road in camera image plane by sensors. Finally, a LBS application system with sensor-based AR technology is described in details from user interaction design to system implementation. This system runs on Android handheld devices, consisting of user layer, encapsulation layer, function layer and network layer.

The methods of the two sensor-based AR in this thesis are designed and implemented by myself independently, and the experimental results meet the demands of LBS application well. As inventor, there are 2 pending patents which are 3D registration method in Mobile AR and navigation route overlaying in Mobile AR.

## ABSTRACT

---

These are also innovative achievements for this paper.

**Key words:** Augmented Reality; Machine Vision; Sensor; Localization Basis Service

# 目 录

第一章 绪论 .....	1
1.1 课题背景和研究意义 .....	1
1.2 研究现状 .....	2
1.2.1 应用类型 .....	2
1.2.2 实现技术 .....	3
1.3 本文的主要工作与创新点 .....	7
1.4 论文的章节安排 .....	7
第二章 基于机器视觉的增强现实技术 .....	9
2.1 图像识别技术 .....	9
2.2 摄像机姿态定位技术 .....	12
第三章 基于传感器的摄像机方向和姿态计算 .....	16
3.1 传感器原理介绍 .....	16
3.1.1 卫星定位 .....	16
3.1.2 电子罗盘 .....	16
3.1.3 重力加速度计 .....	17
3.2 摄像机方向和姿态的计算 .....	17
3.2.1 方向的计算 .....	18
3.2.2 姿态的计算 .....	20
第四章 基于传感器的图像平面投影定位 .....	25
4.1 三维空间到图像平面的投影模型 .....	25
4.2 图像平面投影坐标的计算 .....	26
4.2.1 图像平面投影横坐标 $u$ 的计算 .....	26
4.2.2 图像平面投影纵坐标 $v$ 的计算 .....	28



4.2.3 姿态左右倾斜对横纵坐标的补偿计算 .....	29
<b>第五章 基于传感器的三维虚拟路径叠加 .....</b>	<b>33</b>
5.1 虚拟路径的数据构造 .....	33
5.2 实景空间到 OPENGL 空间的变换关系 .....	34
5.2.1 GPS 坐标系到用户 OpenGL 坐标系的变换 .....	34
5.2.2 用户 OpenGL 坐标系到摄像机 OpenGL 坐标系的变换 .....	35
5.3 虚拟路径的实时生成 .....	37
5.3.1 虚拟路径段的提取 .....	37
5.3.2 虚拟路径段的渲染 .....	38
5.4 阴影渲染 .....	40
<b>第六章 基于传感器增强现实技术的 LBS 应用系统 .....</b>	<b>43</b>
6.1 ANDROID 操作系统及开发介绍 .....	43
6.1.1 Android 操作系统架构 .....	43
6.1.2 基于 Android 操作系统的开发 .....	44
6.2 用户交互设计 .....	46
6.2.1 兴趣点模式 .....	47
6.2.2 导航模式 .....	49
6.3 软件设计与实现 .....	51
6.3.1 系统总体架构 .....	51
6.3.2 系统底层模块 .....	53
6.3.3 用户交互模块 .....	56
<b>第七章 总结与展望 .....</b>	<b>63</b>
<b>致 谢 .....</b>	<b>66</b>
<b>参考文献 .....</b>	<b>67</b>
<b>攻硕期间取得的研究成果 .....</b>	<b>71</b>



## 第一章 绪论

### 1.1 课题背景和研究意义

增强现实（Augmented Reality, AR）是一种将虚拟物体和真实环境有机叠加的技术。相对于虚拟现实（Virtual Reality, VR）不同，增强现实主要以真实环境为基础，将虚拟物体准确叠加至真实环境中，当用户在真实环境中移动或真实环境发生改变时，虚拟物体可以实时地进行相应的变化，使虚拟物体和真实环境保持一种有机的结合。增强现实在医疗、教育、军事、工业、广告、游戏、旅游等领域正逐步得到广泛的应用。

20 世纪 60 年代，最早的增强现实模型是由被誉为计算机图形技术先驱的 Ivan Sutherland<sup>[1]</sup>和他的学生们开发出来的，当时被称为 Ultimate Display（终极显示）。该系统可以追踪人的头部来更新显示，使用户感觉身临其境。随后，从 70 年代到 80 年代，美国各大研究机构，包括美国空军基地阿姆斯特朗实验室、NASA 艾姆斯研究中心和北卡罗来纳大学，都有为数不多的人从事增强现实的相关研究工作。直到 1992 年，增强现实的词汇才由波音公司 Tom Caudell<sup>[2-4]</sup>提出，当时他采用增强现实技术来协助飞机组装。1999 年，华盛顿大学人机交互技术实验室的 Hirokazo Kato 开发出了基于机器视觉的增强现实开发包 ARToolKit<sup>[5-6]</sup>，并在当年的 SIGGRAPH 会议上展示，这推动了增强现实技术的普及，之后越来越多的增强现实开发工具和应用程序随之出现。

随着当今手持设备的发展，越来越多的传统技术和应用正在向手持设备转移，增强现实技术也不例外。所谓移动增强现实，主要是指增强现实技术在手持设备上的应用，除了要具备传统增强现实的虚实结合、实时交互和三维注册的特点<sup>[7]</sup>外，还需要具备较高的自由移动性，不会因为环境因素而只能固定在一个较小范围内活动。第一个移动增强现实系统是于 1997 年由哥伦比亚大学 Steven Feiner 等人开发的名为 Mobile Augmented Reality Systems 的系统<sup>[8]</sup>，用于导航。在 2000 年，一款将增强现实技术应用于 PC 平台经典游戏 Quake 的游戏，ARQuake<sup>[9-12]</sup>，由南澳大利亚大学 Wearable Computer Lab 开发出来，用户可以在在真实环境中参与 Quake 游戏的竞技。2003 年，新加坡国立大学 Mixed Reality Lab 开发的 Human Pacman<sup>[13-14]</sup>游戏同样是利用增强现实技术来实现经典游戏 Pacman。不过这些系

统都需要穿戴笔记本电脑、头戴式显示器等设备，携带不方便，不易推广。目前手机，特别是智能手机，其硬件处理性能飞速发展，并集成了越来越多的传感器，成为了一个增强现实技术应用很好的硬件平台。诺基亚研究院的 Mobile Augmented Reality Applications 项目<sup>[15]</sup>便是在配备了传感器和摄像机的手机上实现增强现实技术，并于 2006 年在 ISMAR06 上展示。2009 年，佐治亚理工学院的 Augmented Environments Lab 也利用高性能手持设备基于机器视觉开发了一款名为 ARhrrrr! 的游戏。

可以看到，增强现实技术，尤其是移动增强现实技术是一个炙手可热的新兴领域，同时随着移动互联网、物联网，甚至最近才提出的视联网的发展，该技术孕育着巨大的市场价值。作者在对移动增强现实技术的研究中，设计并实现了两种基于传感器的增强现实方法，并参与开发了基于传感器增强现实技术的地理位置服务应用系统，在此基础上，对未来的改进和方向进行了进一步的思考。

## 1.2 研究现状

### 1.2.1 应用类型

教学培训。增强现实技术可以使得教学培训具有更直观的表述，并且能够在真实环境、虚拟物体、老师和学生之间产生互动。例如需要给学生展示一个太阳系的运动体系，老师可以利用增强现实技术在真实环境中渲染出虚拟太阳系，学生和老师可以通过自然的肢体动作来和这个虚拟太阳系进行交互，这种方式能够使教学培训效果得到很大的提高。

游戏。游戏是增强现实技术应用中一个重要类型，玩家将得到那些基于虚拟现实的游戏所无法给予的体验。最主要的不同在于增强现实技术可将虚拟物体嵌入真实环境中，游戏过程是在虚拟物体和真实环境之间进行交互。这类应用具有代表性的是 ARQuake、Human Pacman 和 ARhrrrr!。

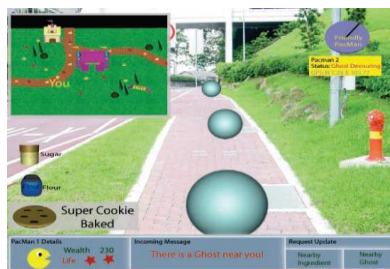


图 1-1 Human Pacman

信息检索。信息检索主要是指导航、社交、搜索等。传统的信息检索是通过用户输入目标对象名称，再由计算机反馈出相应的结果。而通过增强现实技术，当摄像机对准用户周围环境的某一处时，会自动显示出这附近的餐馆、商店等信息，或者获取摄像机中某个人的个人信息。这样的应用相对于传统方式更加自然，并且体现了移动性。这类应用具有代表性的是 Mobilizy 公司推出的 Wikitude，SPRXmobile 公司推出的 Layar。

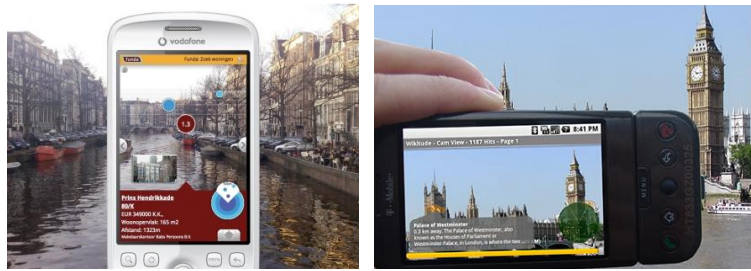


图 1-2 Layar（左）和 Wikitude（右）

内容创建与展示。采用增强现实技术对一个 3D 虚拟模型进行创建和展示，使我们的体验过程更具直观性。用户可以在真实环境中以自然方式从任意角度观察 3D 虚拟模型，并且对模型进行创建和修改。这与传统方式相比，人机交互形式更加趋近于自然行为。

协作支持。增强现实技术可以像虚拟现实技术一样协助人们完成某一作业，而与虚拟现实不同的是，增强现实可以使人们在现场同步完成作业，甚至可以多用户协作同时完成同一个作业。

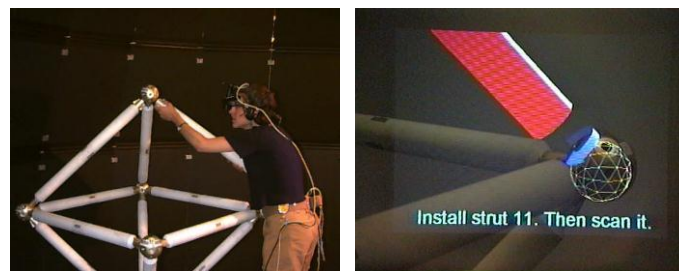


图 1-3 设备安装的协作支持

设备维护与检查。采用增强现实技术来对设备进行维护和检查，可以使工作人员更加直观和高效地发现设备问题，及时准确地了解问题原因，并立即采取相应的修复措施。

## 1.2.2 实现技术

完整的移动增强现实系统架构可以分为跟踪注册、图像渲染显示及无线网络通信三个部分，每一个部分都有相应的技术研究。如图 1-4 所示。

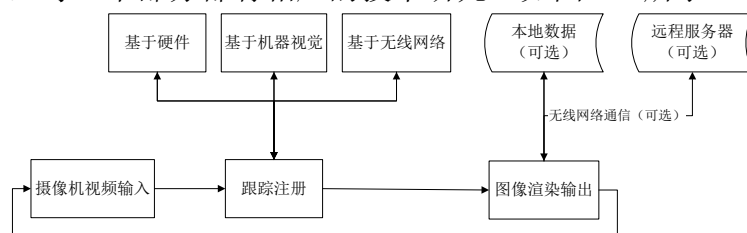


图 1-4 移动增强现实系统架构及运行流程

### 1.2.2.1 跟踪注册技术

跟踪注册技术是指对目标物体进行识别，计算摄像机相对于目标物体的姿态，确定虚拟物体在图像平面的渲染位置。这也是增强现实技术最为重要的部分。跟踪注册技术的实现可采用硬件、机器视觉、无线网络及以上三种混合的方式。

第一种为基于硬件的跟踪注册技术，一般可采用如下实现方法。

红外线发射管（Infra-red LED）通过发射极窄的波段来让特定的图像传感器检测，只有检测到该发射波段时才可以判别跟踪到目标对象，这样节省了大量的计算开销。可参考文献<sup>[16-17]</sup>。

射频识别技术（RFID）可用于增强现实的跟踪注册。射频识别作为一种电子标签，包含一个微处理器和天线，用于发射和接收信息。可参考文献<sup>[18-20]</sup>。

基于超宽带技术（Ultra Wideband）定位系统 Ubisense<sup>TM</sup>，可用于较大室内环境。目前该技术仅用于商用。可参考文献<sup>[21]</sup>。

电磁式跟踪系统，根据磁发射信号和磁感应信号之间的耦合关系确定被测对象的方位，但易受工作环境中磁场和金属物体的影响。可参考文献<sup>[22-23]</sup>。

超声波式跟踪系统，利用不同声源发出的超声波到达某一特定地点的时间差、相位差或者声压差进行定位跟踪，但有遮挡问题，易受环境噪声、温度、湿度等因素的影响。可参考文献<sup>[1, 24-27]</sup>。

第二种为基于机器视觉的跟踪注册技术，其所要求的设备条件简单，成本低廉，但计算开销大，并且稳定性不如传感器。机器视觉本身是一门交叉学科，其主要思想是对摄入的图像数据按照一定算法进行计算，检测出目标物体，获取摄像机相对于目标物体的姿态，并根据摄像机内部参数计算投影模型，最后渲染虚拟物体。

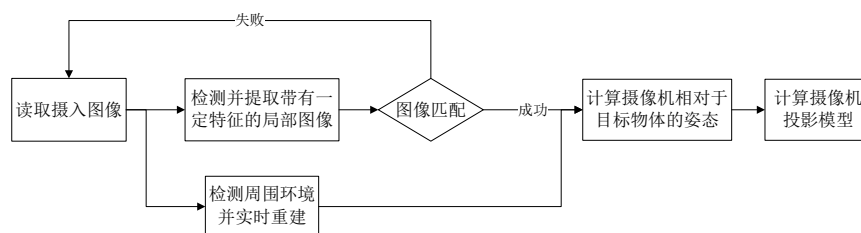


图 1-5 基于机器视觉的跟踪注册流程

系统对摄入图像进行实时检测，提取具有一定特征的局部图像，将其与预存的图像特征信息进行匹配，若匹配成功，则根据该局部图像的形状来计算摄像机的姿态，根据摄像机内部参数计算投影模型。这里涉及到图像识别技术和摄像机姿态定位技术。对于简单的目标物体，图像识别可以采用角点检测、边缘检测等，而对于复杂的目标物体，则可以采用几种比较流行的方法，如 SIFT (Scale-invariant feature transform) [28]、Ferns [29]、和 SURF (Speeded up Robust Features) [30-32]。其中 SIFT 对运算量要求较大，而 Fern 对内存需求较大，这两者都只是适用于台式 PC，而对这两种方法进行修改后可以应用于高性能手持设备上 [33][34]。SURF 在速度和准确性上都有良好的保证，可用于高性能手持设备。摄像机姿态定位一般是在目标物体上取若干已知点，计算这些已知点三维空间坐标与图像坐标之间的对应关系来获取物体坐标系到摄像机坐标系的平移旋转变换矩阵，同时利用摄像机内部参数来构造投影变换矩阵。

另外，SLAM (Simultaneous Localization And Mapping) [35] 方法可以实时地对周围环境进行检测、重建和定位，这种方法不需要预先存储周围环境的特征信息，适用范围更广，特别适用于未知环境，不过计算开销较大。PTAM (Parallel Tracking and Mapping) [36] 是基于该方法运行于 PC 平台的跟踪注册系统，经过改进后可在高性能手持设备上运行 [37]。

第三种为无线网络跟踪注册技术，无线网络可以用于检测和定位。

GSM 网络的手机可以通过对信号的三角测量法来获取手机当前的位置，但是准确度不高。

无线局域网，如 WIFI 等，其协议会提供基于无线接入点的 RSSI (Received Signal Strength Indication) 定位功能。基于 RSSI 的定位方法无需其它的传感器，而仅需已有的通信参数和可下载的无线地图。而这种方法的缺点则是精确度还不够高，并且需要多个无线接入点，以及还要预先构建无线地图 [38-39]。

以上三种跟踪注册方法也可以混合使用。基于机器视觉的方法对快速移动的场景无法准确跟踪和注册，对静止的或移动缓慢的场景有效，而采用传感器可以

弥补机器视觉的缺陷。可参考文献<sup>[40-41]</sup>。

### 1.2.2.2 显示技术

在显示方式上，一般分为视频透视式（Video see-through）和光学透视式（Optical see-through）。视频透视式（Video see-through）的优点是可利用 VR 头盔显示器，对跟踪注册系统的要求较低，但缺点是需要进行图像融合，并且需要使用两台以上摄像机拍摄外部图像。光学透视式（Optical see-through）的优点是对真实世界的感知准确，不易产生晕眩，减少头部负重，但缺点是需要重新研制头盔显示器，对跟踪注册系统的精度和实时性能要求高。

显示技术更多的是注重虚实融合。输出的图像需要将虚拟物体和真实环境同时输出，这要求两者能够准确地融合在一起，使用户感觉是真实存在的。要达到这一点，首先虚拟物体的建模要逼真，其次要对真实环境的光照信息进行检测，例如强度、方向、阴影等，并应用在虚拟物体上，然后虚实物体之间的遮挡关系要准确，最后达到无缝拼接的效果。但这会产生输出质量与实时性的平衡问题。2008 年牛津大学工程科学系 Active Vision Laboratory 的 Georg Klein 和 David Murray 所发表的论文《Compositing for Small Cameras》<sup>[42]</sup>就提出了一种虚实融合的方法。

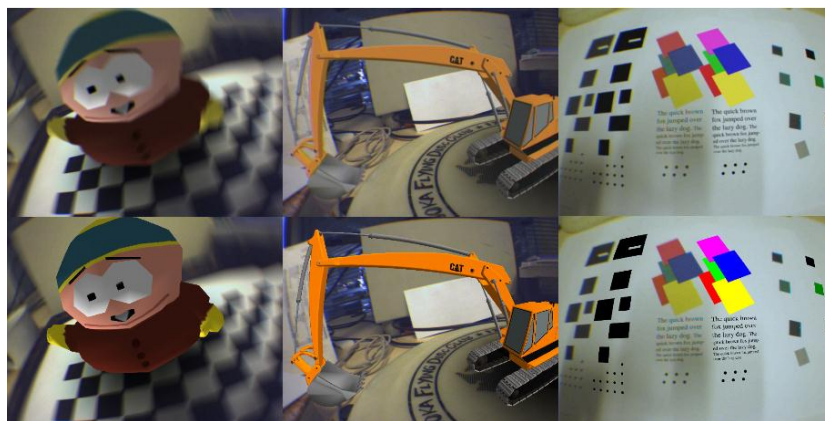


图 1-6 上图的虚实融合效果比下图好

### 1.2.2.3 无线网络通信技术

在一些移动增强现实的应用中，需要通过远程服务器来存储大量的数据信息，甚至对数据的处理都可以交给远程服务器来完成。所以，无线网络通信技术在移动增强现实应用中也至关重要。和有线网络相比，无线网络在延时、带宽、带宽波动和获取性都有一定局限性，这些对用户体验都有很大的影响。对于移动增强现实应用，无线网络需提供足够的数据传输率、低延时以及可靠的移动性。当前



的无线网络分为无线广域网（WWAN）、无线局域网（WLAN）、无线个人区域网络（WPAN）和无线社区区域网络（WMAN），对于不同的移动增强现实应用来说，无线网络的选择也是不同的，例如，在固定区域的可以选择无线局域网，而在户外移动范围很大的则可以选择无线广域网。

### 1.3 本文的主要工作与创新点

在本文中，首先研究基于机器视觉的增强现实技术，重点阐述图像识别技术和摄像机姿态定位技术。再提出应用于地理位置服务的两种基于传感器的增强现实方法，一种是图像平面的投影定位，一种是三维虚拟路径的叠加。先介绍各种传感器的原理，然后详细阐述这两种基于传感器的增强现实技术的实现方法。

接下来，作者设计并实现了一个基于传感器增强现实技术的地理位置服务应用系统，该系统运行于 Android 操作系统手持设备上。先介绍 Android 操作系统的基本知识，再介绍该应用系统的用户交互设计和系统架构及模块的设计和实现，分别阐述各个模块，对如何将基于传感器的增强现实技术嵌入至应用系统进行分析，使各种底层输入数据呈现为直观的可交互的图形，达到最佳用户体验。比较总结该应用系统相对于传统地理位置服务应用所带来的全新用户体验。

本文的主要创新点在于，针对手持设备的性能和地理位置服务的需求，避开基于机器视觉的增强现实技术，提出两种基于传感器的增强现实技术解决方案，在满足增强现实虚实结合、实时交互和三维注册的基本要求下，还因不受地理空间、自然环境的限制，自由移动，很好地体现了移动性。

### 1.4 论文的章节安排

第一章是全文的绪论，介绍了移动增强现实技术的背景，应用类型和相关的实现技术，并介绍了本文的结构和进行的工作。

第二章介绍基于机器视觉的增强现实技术，分别阐述图像识别和摄像机姿态定位的技术原理。

第三章介绍基于传感器的摄像机方向和姿态计算，首先介绍各种传感器的基本原理，再阐述如何利用传感器计算摄像机的方向角度和姿态角度。摄像机方向和姿态为后两章阐述的方法所用。

第四章介绍基于传感器的图像平面投影定位，介绍三维空间到图像平面的基

本投影原理，阐述利用该投影原理来获取目标物体在摄像机图像平面投影坐标的计算方法。

第五章介绍基于传感器的三维虚拟路径叠加，先介绍虚拟路径的数据构造模型，再阐述实景空间到 OpenGL 空间的变换过程，然后描述虚拟路径实时生成的方法。为了增强虚实融合的效果，对虚拟路径增加了阴影渲染，因此还对阴影渲染的方法作了简单的描述。

第六章介绍基于传感器增强现实技术的 LBS 应用系统，首先介绍 Android 操作系统，其次对应用系统的用户交互设计进行阐述，再根据需求，设计和实现了系统架构及各个模块，对每一个模块的实现进行详细地描述，对如何将基于传感器的增强现实技术嵌入至实际的应用系统中进行了分析和阐述。

最后，在第七章，对全文的工作做出了系统全面的总结，并对今后需要进一步深入研究的方向和移动增强现实的发展前景做了展望。

## 第二章 基于机器视觉的增强现实技术

基于机器视觉的增强现实技术主要是通过对摄像机输出视频流中的图像进行检测，识别出目标物体，然后计算出摄像机相对于该物体的姿态，并根据这个摄像机姿态来确定虚拟物体在屏幕上的渲染位置，最后渲染虚拟物体。整个过程涉及到图像识别技术和摄像机姿态定位技术，本章节中主要对这两种技术进行阐述。

### 2.1 图像识别技术

图像识别技术的基本原理是将获取到的图像与模板图像数据进行对比，根据一定的相似度来判断是否匹配。整个过程涉及到图像的阈值化、轮廓提取、角点检测、图像归一化、模板匹配等技术。

图 2-1 是从摄像机获取的一帧图像，图像中有一个标志点，作为目标物体。现在需要从这一帧图像中，识别出这个标志点。



图 2-1 从摄像机获取的一帧包含标志点的图像

首先对图像进行阈值化。一张图像由若干像素组成，每一个像素具有 RGB 三色分量，分别为红绿蓝，来表示该像素的颜色值，其中每一个分量的取值范围为[0, 255]。同时也可以使用 YUV 来表示像素的颜色，Y 表示亮度，也就是图像的灰度信息，U 和 V 表示色差，每个分量的取值范围也是[0, 255]。图像的 RGB 分量值和 YUV 分量值之间是可以转换的，转换关系请参见式 2-1。将一张彩色图像中所有像素 RGB 的每一个分量值设为其对应 YUV 的 Y 分量值，就可以转换成灰度图像，这个 Y 分量值就是像素的灰度值。

$$\begin{bmatrix} Y & U & V \end{bmatrix} = \begin{bmatrix} R & G & B \end{bmatrix} \begin{bmatrix} 0.299 & -0.148 & 0.615 \\ 0.587 & -0.289 & -0.515 \\ 0.114 & 0.437 & -0.100 \end{bmatrix} \quad (2-1)$$

阈值化是将灰度图像中每一个像素的灰度值与一个阈值进行比较，小于该阈值则该像素的灰度值变为 0（或 255），为黑色（或白色），大于该阈值则该像素的灰度值变为 255（或 0），为白色（或黑色），通常将这个阈值设为 100。阈值化使得灰度图像分割成黑白二值图像，用于标志点的轮廓提取。图 2-2 是阈值化后的图像。



图 2-2 阈值化后的黑白二值图像

标志点的形状为矩形，因此，要对这个阈值化后的黑白二值图像进行轮廓跟踪标记，提取具有矩形形状的连通域。在进行轮廓跟踪时，可以从图像的第一个像素开始，按从左至右、从上至下的顺序开始检测像素的灰度值。在图 2-2 中，标志点最外层轮廓为白色，因此，在对像素灰度值进行检测时，第一个灰度值为 255 的像素一定为某一个连通域最左上方的边界像素，将该像素记为 A。像素 A 周围有八个相邻像素，像素 A 的右、右下、下和左下相邻像素中，至少有一个是灰度值为 255 的边界像素，记为像素 B。从像素 B 开始，按照顺时针方向检测相邻像素中的边界像素，直到这个边界像素等于像素 A，说明已经绕这个连通域转了一圈了。判断一个像素是否为边界像素时，若该像素上下左右的相邻像素的灰度值只要存在一个为 0，则该像素为边界像素。通过这种轮廓跟踪的方法可以将图像中所有轮廓为白色的连通域检测出来，并标记。

对已标记的每一个连通域的形状进行判别，提取具有矩形轮廓的连通域。按顺时针方向，顺序建立连通域轮廓上的所有像素索引及数量 pnum，设轮廓跟踪时检测到的该连通域的第一个像素索引为 p<sub>1</sub>，并计算出其中离 p<sub>1</sub> 距离最远的另一个像素，设其索引为 p<sub>2</sub>。顺时针方向沿着连通域轮廓，遍历从 p<sub>1</sub> 到 p<sub>2</sub> 所有的轮廓像素，进行角点检测，记录下这段轮廓中所有顶点的像素索引和顶点个数

$vnum_1$ 。再沿着同样的方向，遍历从  $p_2$  到  $p_1$  剩余的轮廓像素，进行角点检测，记录下这段轮廓中所有顶点的像素索引和顶点个数  $vnum_2$ 。根据矩形的几何拓扑结构和因摄像机的角度产生的矩形轮廓变形，若连通域为矩形，则  $p_1$  一定为一个顶点，而  $p_2$  则可能为  $p_1$  的对角顶点，或为其某一个相邻顶点。图 2-3 展示了三种可能的情况。

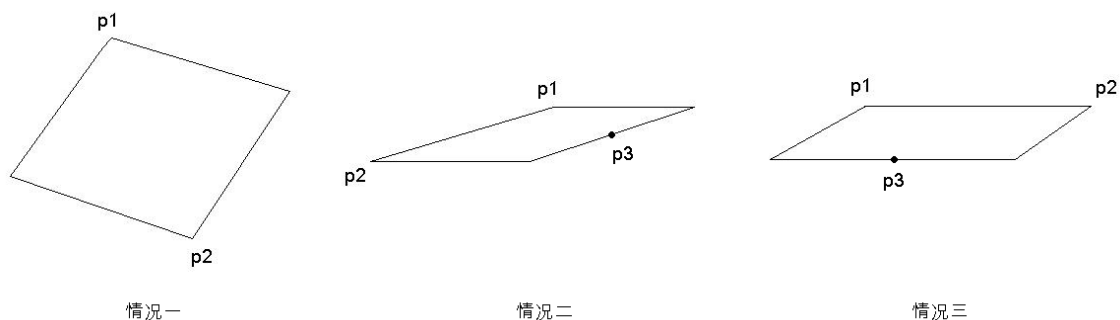


图 2-3 矩形轮廓的判定

在情况一中，当  $vnum_1$  和  $vnum_2$  都等于 1 时，该连通域为矩形。在情况二中， $p_2$  为  $p_1$  的左相邻顶点，此时  $vnum_1$  大于 1， $vnum_2$  等于 0。在该连通域轮廓中确定第三个像素索引  $p_3$ ， $p_3 = p_2/2$ ，再按照同样的遍历方法遍历  $p_1$  到  $p_3$  和  $p_3$  到  $p_2$  的轮廓像素，当  $vnum_1$  和  $vnum_2$  都等于 1 时，该连通域为矩形。情况三与情况二类似， $p_2$  为  $p_1$  的右相邻顶点，此时  $vnum_1$  等于 0， $vnum_2$  大于 1。仍然需要确定第三个像素索引  $p_3$ ， $p_3 = (p_2 + pnum - 1)/2$ ，再做  $p_2$  到  $p_3$  和  $p_3$  到  $p_1$  的遍历，当  $vnum_1$  和  $vnum_2$  都等于 1 时，该连通域为矩形。

提取到的矩形连通域被看作是可能的标志点，构成候选区，并记录下数量。如图 2-4 所示，由于摄像机的角度，使得候选区中矩形连通域的图像产生变形，而模板图像是一种标准化的数据信息，其图像平面角度和像素尺寸是固定的，因此需要对该矩形连通域中的图像进行归一化，并调整其分辨率，使得其图像数据与模板图像一致，为待匹配图像。

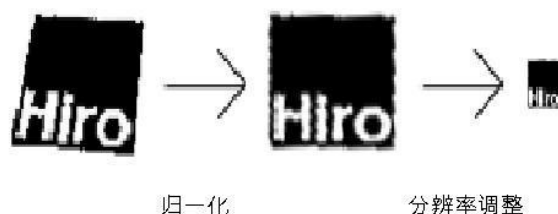


图 2-4 图像归一化和分辨率调整

最后，将待匹配图像与模板图像进行对比。由于两者的像素尺寸是相同的，可以通过计算两者所有对应像素灰度值的平方误差之和来获取一个误差值，误差值越小，相似度越高。公式如下：

$$err = \sum_{y=0}^{n-1} \sum_{x=0}^{m-1} [g_1(x, y) - g_2(x, y)]^2 \quad (2-2)$$

其中， $n$  和  $m$  分别表示图像的高和宽， $g_1(x, y)$  表示待匹配图像中像素  $(x, y)$  的灰度值， $g_2(x, y)$  表示模板图像中像素  $(x, y)$  的灰度值， $err$  表示误差值。可以设置一个误差阈值，若  $err$  小于这个阈值，这表示匹配成功，该待匹配图像为目标物体。

## 2.2 摄像机姿态定位技术

为了使得虚拟物体能够准确地渲染叠加在摄像机输出视频流中目标物体上，就需要对摄像机的姿态进行定位。定位主要是确定目标物体的物体坐标系到摄像机坐标系的平移旋转变换矩阵，以及摄像机坐标系到图像坐标系的投影变换矩阵。通过这两个变换矩阵就可以确定虚拟物体在图像平面上的位置，而该位置也是目标物体在图像平面上的位置，这样便实现了虚拟物体和目标物体的叠加。

图 2-5 为机器视觉中各个坐标系之间的变换关系，以标志点作为目标物体，标志点的一个角点  $A$  在图像平面上的投影点是  $A'$ 。

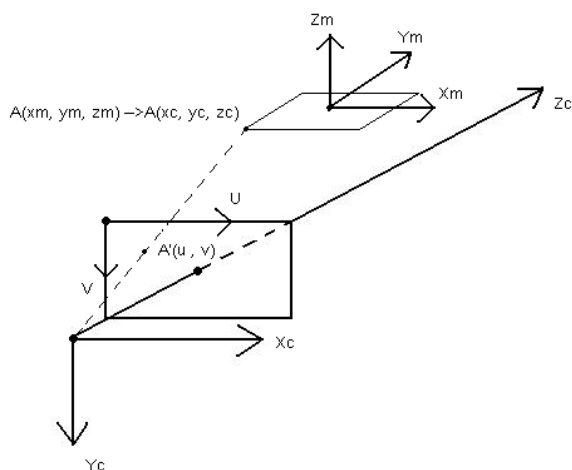


图 2-5 机器视觉中各个坐标系之间的变换关系

角点  $A$  和其它三个角点在标志点物体坐标系上的坐标是已知的，并且在进行

图像识别时可以获取这四个角点在输出视频流当前帧中图像坐标系上的坐标，因此，可以将这四个角点在物体坐标系上的坐标与图像坐标系上的坐标建立起如下对应关系：

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = M \begin{bmatrix} x_{mi} \\ y_{mi} \\ z_{mi} \\ 1 \end{bmatrix}, \quad i = 1, 2, 3, 4 \quad (2-3)$$

其中， $u$  和  $v$  表示角点的图像坐标， $x_m$ 、 $y_m$  和  $z_m$  表示角点在物体坐标系上的坐标，都是已知参数， $M$  表示一个坐标变换矩阵，是未知参数。

将这个变换矩阵  $M$  分解开来，如下：

$$M = M_1 M_2 \quad (2-4)$$

$$M_1 = \begin{bmatrix} a_x & 0 & u_0 \\ 0 & a_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$M_2 = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}$$

变换矩阵  $M$  包含两个部分， $M_1$  为摄像机内部参数，可通过摄像机标定获取，其中  $a_x=f/dx$ ， $a_y=f/dy$ ， $f$  为摄像机焦距， $dx$  和  $dy$  分别表示单个像素在图像平面横向和纵向上的物理尺寸，单位毫米。 $u_0$  和  $v_0$  为图像平面中心在图像坐标系上的坐标值。 $M_2$  为需要求取的  $3 \times 4$  平移旋转变换矩阵，前三列用于旋转变换，最后一列用于平移变换，通过该变换矩阵可以将虚拟物体从目标物体的物体坐标系变换到摄像机坐标系上。

事实上，这四个角点在标志点物体坐标系上的坐标  $z_m$  值是等于 0 的，因此，将式 2-3 带入式 2-4 可得：

$$\begin{aligned}
 \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} &= \begin{bmatrix} a_x & 0 & u_0 \\ 0 & a_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} x_{mi} \\ y_{mi} \\ z_{mi} \\ 1 \end{bmatrix} \\
 &= \begin{bmatrix} a_x & 0 & u_0 \\ 0 & a_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{bmatrix} \begin{bmatrix} x_{mi} \\ y_{mi} \\ 1 \end{bmatrix} \\
 &= H \begin{bmatrix} x_{mi} \\ y_{mi} \\ 1 \end{bmatrix} \\
 i &= 1, 2, 3, 4
 \end{aligned} \tag{2-5}$$

$H$  为  $3 \times 3$  的单应矩阵<sup>[43]</sup>, 表示一个在物体坐标系中  $Z=0$  的平面与其图像平面之间的对应关系, 可以利用四个共面不共线的已知点求取。标志点正好有四个共面不共线的已知点, 符合单应矩阵  $H$  的计算思想。现在  $M_2$  变成了一个  $3 \times 3$  的变换矩阵, 有 9 个未知数, 利用式 2-5 可以列出 12 个方程, 进行联立求解。方程如下:

$$\begin{aligned}
 u_i &= (a_x \times r_{11} + u_0 \times r_{31}) \times x_{mi} + (a_x \times r_{12} + u_0 \times r_{32}) \times y_{mi} + a_x \times t_1 + u_0 \times t_3 \\
 v_i &= (a_y \times r_{21} + v_0 \times r_{31}) \times x_{mi} + (a_y \times r_{22} + v_0 \times r_{32}) \times y_{mi} + a_y \times t_2 + v_0 \times t_3 \\
 1 &= r_{31} \times x_{mi} + r_{32} \times y_{mi} + t_3 \\
 i &= 1, 2, 3, 4
 \end{aligned} \tag{2-6}$$

经过联立求解后, 便可以获取  $M_2$  中用于旋转变换的前两列向量和用于平移变换的最后一列向量。根据旋转变换矩阵的正交性, 前两列向量的叉乘可以求出第三列向量。这样, 便获取到了平移旋转变换矩阵  $M_2$ 。

投影变换分为两种, 正交投影和透视投影, 这里采用透视投影。在图像平面上的透视投影效果是由摄像机的投影属性决定的, 将摄像机的视觉范围看成一个视锥模型, 如图 2-6 所示。

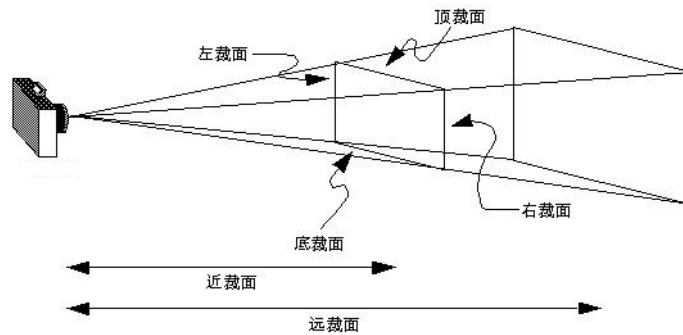


图 2-6 视锥模型



视锥模型有六个参数，分别为左截面、右截面、顶截面、底截面、近截面和远截面的位置，透视投影变换矩阵通过这六个参数来构造。在 OpenGL 中，可以将这个视锥的六个参数带入透视投影变换函数 `glFrustum()` 来设置该透视投影变换矩阵。透视投影变换矩阵的标准模型如下：

$$M_{frustum} = \begin{bmatrix} \frac{2n}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2n}{t-b} & 0 & 0 \\ \frac{r+l}{r-l} & \frac{t+b}{t-b} & \frac{n+f}{n-f} & -1 \\ 0 & 0 & \frac{2nf}{n-f} & 0 \end{bmatrix} \quad (2-7)$$

透视投影矩阵也可以通过摄像机的视觉广度、图像平面的宽高比和远近截面来构造透视投影变换矩阵，其实这种方法是间接地计算视锥模型。在 OpenGL 中，相对应的函数是 `gluPerspective()`。其标准模型如下：

$$M_{persp} = \begin{bmatrix} \cot \frac{fovy}{2} & 0 & 0 & 0 \\ \frac{w}{h} & \cot \frac{fovy}{2} & 0 & 0 \\ 0 & 0 & \frac{n+f}{n-f} & -1 \\ 0 & 0 & \frac{2nf}{n-f} & 0 \end{bmatrix} \quad (2-8)$$

以上两种透视投影变换矩阵的效果是一样的，其中各个元素可以利用摄像机内部参数来计算，这样便可以获得需要求取的透视投影变换矩阵。

图 2-7 展示了利用 ARToolKit 实现的太阳、地球及月亮运行的虚拟演示效果。ARToolKit 是一款基于机器视觉的增强现实技术的软件开发包，提供了诸如摄像机标定、图像模板生成、图像识别、摄像机姿态定位等功能。

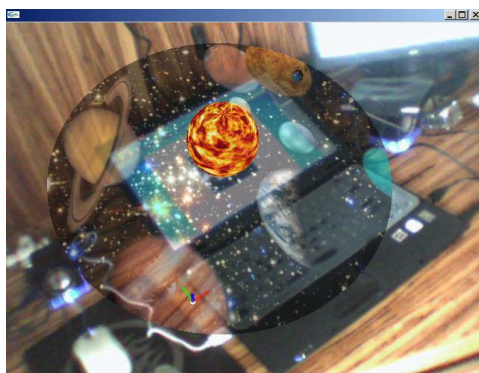


图 2-7 利用 ARToolKit 实现的演示效果

## 第三章 基于传感器的摄像机方向和姿态计算

### 3.1 传感器原理介绍

本文所涉及到的传感器包括卫星定位接收器、电子罗盘和重力加速度计，下面分别对其进行介绍。

#### 3.1.1 卫星定位

世界上有四大卫星定位系统，美国的全球卫星定位系统、俄罗斯的格罗纳斯导航系统、欧盟的伽利略导航系统和中国的北斗导航系统，目前全球应用最广的还是美国的全球卫星定位系统，本文也采用该系统。

全球定位系统，英文是 Global Positioning System，简称 GPS。原名被称为“导航星”（Navstar），是 1973 年美国国防部授权研制的海陆空三军共用的美国第二代卫星定位系统，并于 1994 年建成。GPS 是一个基于空间的卫星定位系统，能够在全球任意地方、全天候地提供可靠的地理位置信息，不受天气等因素影响。地理位置信息主要包括经度（longitude）、纬度（latitude）和海拔（altitude）三维坐标。

#### 3.1.2 电子罗盘

罗盘是用于定位方向的仪器，地球的两极是一个具有巨大的吸引力的磁场，磁针受同性相斥、异性相吸的自然法则，在地球磁场的作用下，会停止在一个指向南北方向的位置。

电子罗盘同样利用地球磁场，其中一个重要的元件，磁阻传感器，就用来监测磁场的方向。一般用两个相互垂直磁阻传感器，并且其组成的平面与地平面平行。方向向前的磁阻传感器用于检测地球磁场在该方向的分量值  $x$ ，方向向左的磁阻传感器用于检测地球磁场在该方向的分量值  $y$ ，然后利用公式  $\text{azimuth} = \arctan(y/x)$  来计算出实际的方位值。

在有的电子罗盘中会加入与前两个磁阻传感器相互垂直的第三个磁阻传感器，用于测量地球磁场在朝下的分量值  $z$ ，以及倾角传感器，用于测量罗盘倾斜

时的前后及左右角度变化。这是因为当罗盘不在水平位置而发生倾斜时，水平方向的两个磁阻传感器所监测到的磁场方向会产生误差，此时，利用朝下的磁阻传感器监测到的地球磁场分量值  $z$  和倾角传感器监测到的前后及左右倾斜角度  $\alpha$  和  $\beta$ ，来计算地球磁场  $x$  和  $y$  分量值的修正值  $x_1$  和  $y_1$ 。计算公式分别为  $x_1 = x \cos \alpha + y \sin \alpha \sin \beta - z \cos \beta \sin \alpha$  和  $y_1 = x \cos \beta + z \sin \beta$ 。


### 3.1.3 重力加速度计

重力加速度计的本质是将运动或重力变化转换成电信号，利用了压电效应的原理。压电效应是指对于不存在对称中心的异极晶体加在晶体上的外力除了使晶体发生形变以外，还将改变晶体的极化状态，在晶体内部建立电场的现象。这是一种因机械力作用使介质发生极化的过程。重力加速度计内部采用了这种晶体，当产生外力作用时，会导致晶体变形，从而产生电压，计算该电压和所施加的外力之间的关系，便可将外力转换成电压输出。

当重力加速度计处于静止状态时，由于地球引力，有一个朝向地心的外力对其作用，则使得该重力加速度计测出当前朝向地心方向的加速度值约为 9.8，正好是地球的重力加速度值。重力加速度计可用于测量物体的加速度、倾斜角等，在本文中则使用重力加速度计来计算摄像机的姿态。

## 3.2 摄像机方向和姿态的计算

利用电子罗盘和重力加速度计的读数分别计算摄像机的方向及姿态数据。图 3-1 显示计算得到的摄像机方向和姿态的测试数据， $dir$  为方向角度， $inc$  为前后俯仰角度， $rol$  为左右倾斜角度。



```
dir:-871.9732 inc:-19.172016 rol:23.376394
```

图 3-1 经过处理后的方向和姿态数据

这里将阐述如何利用从传感器获取的数据来计算摄像机方向和姿态。由于摄像机是嵌在手持设备上的，因此摄像机与手持设备的方向和姿态是保持一致的。对手持设备定义一个坐标系，该坐标系与 OpenGL 所定义的坐标系一致，为右手坐标系，如图 3-2 所示。

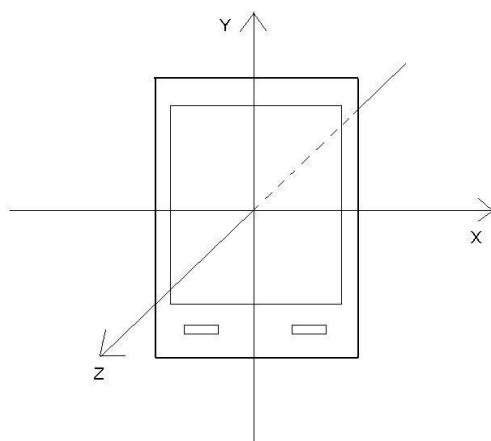


图 3-2 手持设备坐标系

该坐标系定义 Z 轴正方向是朝向屏幕内，X 轴正方向水平向右，Y 轴正方向垂直向上。

### 3.2.1 方向的计算

方向角度通过电子罗盘的读数来获取，设该读数为  $\omega_0$ 。当手持设备屏幕朝上水平放置时， $\omega_0$  表示地磁北极方向与 Y 轴的夹角，0=北、90=东、180=南、270=西。在实际应用中，手持设备是竖直放置，屏幕朝内，摄像机朝外。根据前面对电子罗盘的介绍，由于配备了倾角传感器，可以对因姿态倾斜而造成电子罗盘方向角度的误差进行补偿。经过实验，发现手持设备分别水平和竖直放置时，其电子罗盘的方向角度未发生改变。图 3-3 是手持设备中电子罗盘方向角度的示意图。

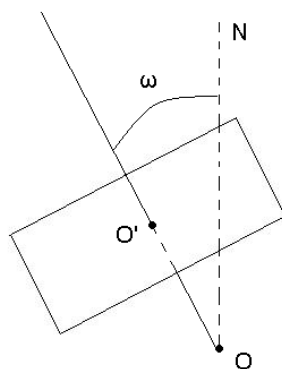


图 3-3 电子罗盘方向

根据对  $\omega_0$  的值进行测试，发现有三个问题需要解决。第一是  $\omega_0$  值的突变问题。 $\omega_0$  的读数范围是  $[0, 359]$ ，从这个范围可以看到，0 和 359 是临界值，当罗盘

从东向西旋转时,会发生数值 0 到 359 的突变,从西向东旋转时,会发生数值 359 到 0 的突变。突变导致在临界值获取的数据不稳定,需要对这个突变进行平滑处理,使其数值变化形式为-2、-1、0、1、2、…、359、360、361、…,呈线性递增或递减的变化。这就需要判断数据  $\omega_0$  是否发生了数值突变。因此需要一个变量 **rotCounter** 来表示累积旋转次数,若发生一次突变,则 **rotCounter** 增加或减少一次。通过对数据  $\omega_0$  前一值和后一值的差值来判断是否发生突变,若差值大于某个指定值,可设为 180,说明发生了 359 到 0 的突变, **rotCounter** 增加一次,若差值小于 -180,说明发生了 0 到 359 的突变, **rotCounter** 减少一次。

计算出旋转次数 **rotCounter** 后,就可以对这个方向角度进行处理。旋转的周期为 360,公式如下:

$$\omega_1 = \omega_0 + \text{rotCounter} \times 360 \quad (3-1)$$

此时,数据  $\omega_1$  的范围为  $[-\infty, +\infty]$ ,且呈线性递增或递减的变化,符合计算的要求。

第二个问题是由于电子罗盘敏感度较大,即使手持设备处于静止状态,数据  $\omega_0$  仍然会出现较大的抖动,导致数据  $\omega_1$  也产生较大的抖动。就需要**对数据  $\omega_1$  过滤,使其抖动幅度减小**。关于过滤的方法,作者测试过几种方法,第一种方法在读取电子罗盘数据时,间隔若干时间再读取一次,实验时采用间隔 0.5 秒读取一次,但测试结果发现,抖动仍然存在,只是每一次更新频率由间隔时间决定,时间过长,会出现严重的滞后,时间过短,则与直接读取电子罗盘数据无任何差别。第二种方法是将前一次读取的电子罗盘数据和当前读取的电子罗盘数据进行比较,若前一次比当前大,则当前的数据适当增加一固定值,若前一次比当前小,则当前的数据适当减少一固定值,这样做的目的是将前后读取的值的差值缩小,达到过滤的目的,但测试效果不理想,仍然存在较大抖动。第三种方法是将旋转周期 360 分成  $n$  块,当电子罗盘读数落入某一块的范围时,则只采用某一固定的值,数学表达式为  $\text{floor}(\text{vals}[0]/n) \times n$ ,  $\text{floor}(x)$  函数表示“下取整”。但采用这种方法时,若  $n$  的值过小时,计算出的数据存在严重的不连续性,若  $n$  值过大时,计算出的数据同样会产生较大的抖动幅度。第四种方法是在若干时间内计算一次这段时间内电子罗盘读数的平均值,来达到减少抖动幅度的目的,但是,由于输入的原始电子罗盘方向数据本身存在抖动,所以计算出来的平均值之间也存在同样的抖动幅度。以上的方法都不能够有效的解决抖动问题,作者则采用了第五种方法,公式如下:

$$\omega_2 = \omega_1 \times filterFactor + \omega_2 \times (1 - filterFactor) \quad (3-2)$$

其中 filterFactor 是过滤系数，将其设置为 0.05 最合适。

通过将数据  $\omega_1$  带入式 进行计算后得到的结果数据  $\omega_2$  则具有极小的抖动性，其测试结果满足要求。

第三个问题是当手持设备左右倾斜时，由于电子罗盘与地球磁场的相对方向产生改变，造成电子罗盘方向角度读数  $\omega_0$  发生改变，使得当前计算出的  $\omega_2$  也同样产生变化，但此时摄像机的朝向并未发生改变，因此需要对这种因姿态左右倾斜造成电子罗盘方向读数的偏差而进行补偿。利用重力加速度计来计算一个左右倾斜的角度，设该角度为  $\gamma$ ，根据手持设备左右倾斜时，方向角度  $\omega_2$  值的变化情况， $\gamma$  的取值范围应是  $[-90, +270]$ ，手持设备的姿态从 Y 轴水平朝右，X 轴垂直朝下，逆时针绕 Z 轴旋转 360 度时， $\gamma$  的值从 -90 度到 270 度变化。 $\gamma$  的具体计算方法将会在姿态计算中详细阐述。由于电子罗盘和重力加速度计都是实时产生数据，所以，这种补偿也是实时的。对电子罗盘方向角度进行补偿的公式如下：

$$\omega = \omega_2 + \gamma \quad (3-3)$$

这时，最后计算得到的  $\omega$  就是需要的方向角度，取值范围是  $[-\infty, +\infty]$ 。

### 3.2.2 姿态的计算

手持设备的姿态分为两种，一是左右倾斜，一是前后俯仰。这些角度的计算需要通过重力加速度计来实现。重力加速度计产生三个数据，分别设为  $V_x$ 、 $V_y$  和  $V_z$ 。在手持设备坐标系中， $V_x = A_x - G_x$ ，表示加速度在 X 轴上的分量与重力加速度在 X 轴上的分量的差值； $V_y = A_y - G_y$ ，表示加速度在 Y 轴上的分量与重力加速度在 Y 轴上的分量的差值； $V_z = A_z - G_z$ ，表示加速度在 Z 轴上的分量与重力加速度在 Z 轴上的分量的差值。当手持设备处于静止状态时， $A_x$ 、 $A_y$  和  $A_z$  均为 0。可以通过  $V_x$ 、 $V_y$  和  $V_z$  来计算姿态。

#### 3.2.2.1 左右倾斜的计算

在计算手持设备左右倾斜角度时，只需用到  $V_x$  和  $V_y$ 。

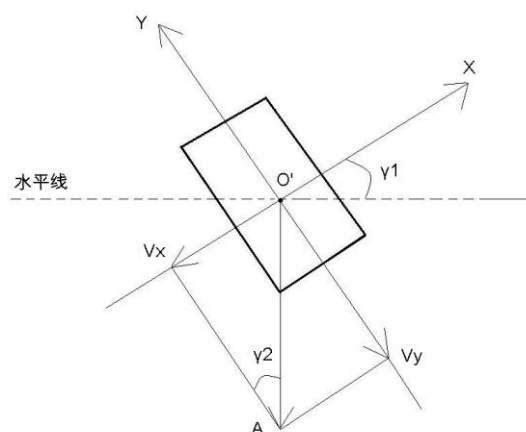


图 3-4 左右倾斜角度

如图 3-4 所示，要求取的倾斜角度是  $\gamma_1$ ，根据几何定理，可以得出  $\gamma_1 = \gamma_2$ ，因此，求取  $\gamma_1$  转换成了求取  $\gamma_2$ 。又根据三角函数定理，可得公式如下：

$$\gamma_1 = \gamma_2 = \arctan \frac{V_x}{V_y} \quad (3-4)$$

反正切函数的取值范围是在  $[-90, +90]$ 。当手持设备的姿态从 Y 轴水平朝右，X 轴垂直朝下，绕 Z 轴逆时针旋转 180 度至 Y 轴水平朝左，X 轴垂直朝上， $\gamma_1$  的变化过程是由 -90 度递增至 90 度，为了表述方便，如图 3-5 所示，称姿态属于该变化范围内的为倾斜姿态 1，然后手持设备又从该姿态绕 Z 轴继续逆时针旋转 180 度至 Y 轴水平朝右，X 轴垂直朝下， $\gamma_1$  的变化过程同样是由 -90 度递增至 90 度，称为倾斜姿态 2。

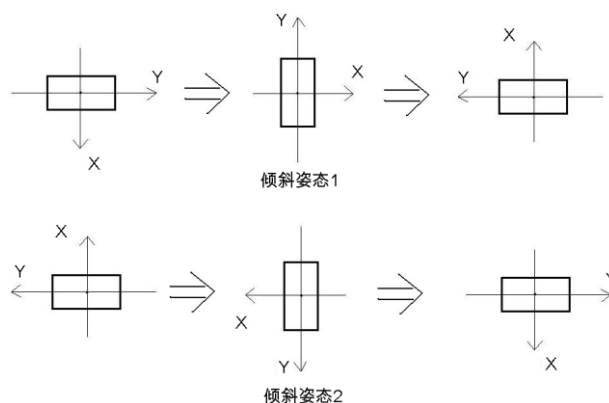


图 3-5 左右倾斜的两种姿态

为了对电子罗盘的读数进行补偿，需要对倾斜姿态 2 的  $\gamma_1$  值进行修正，使其变化由 90 度递增至 270 度。在计算  $\gamma_1$  时进行判断，若  $V_y$  的值大于 0，手持设备

处于倾斜姿态 1，就直接采用式 3-4 计算，若  $V_y$  的值小于 0，处于倾斜姿态 2，则采用以下公式计算：

$$\gamma_1 = \gamma_2 = 180 + \arctan \frac{V_x}{V_y} \quad (3-5)$$

和电子罗盘一样，重力加速度计的敏感度也相当高，数据  $V_x$ 、 $V_y$  和  $V_z$  存在较大的抖动，导致计算出的  $\gamma_1$  也产生较大的抖动，需要对  $\gamma_1$  进行过滤。过滤方法和电子罗盘一样，如下：

$$\gamma = \gamma_1 \times \text{filterFactor} + \gamma \times (1 - \text{filterFactor}) \quad (3-6)$$

这时，最后计算得到的  $\gamma$  就是需要的左右倾斜角度，取值范围是  $[-90, +270]$ 。

### 3.2.2.2 前后俯仰的计算

计算手持设备前后俯仰角度时，需要分别考虑手持设备竖直和横向两种状态。若为竖直状态，则通过  $V_y$  和  $V_z$  来计算前后俯仰角度，若为横向状态，则通过  $V_x$  和  $V_z$  来计算前后俯仰角度。

如图 3-6 所示，竖直状态表示手持设备的左右倾斜角度  $\gamma$  处于  $(-45, +45)$  或  $(135, 225)$  之间，横向状态表示手持设备的左右倾斜角度  $\gamma$  处于  $[45, 135]$  或  $[225, 270]$  和  $[-90, -45]$  之间。

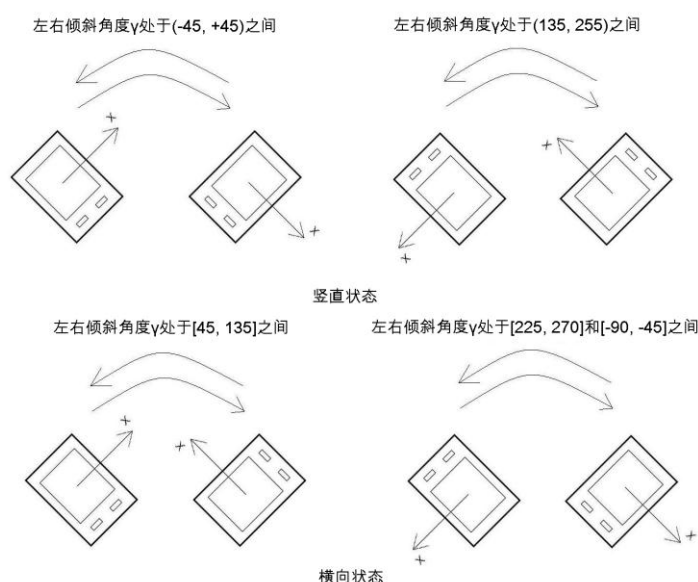


图 3-6 手持设备竖直和横向两种状态



先讨论手持设备为竖直状态时的前后俯仰角度的计算方法，如图 3-7 所示。

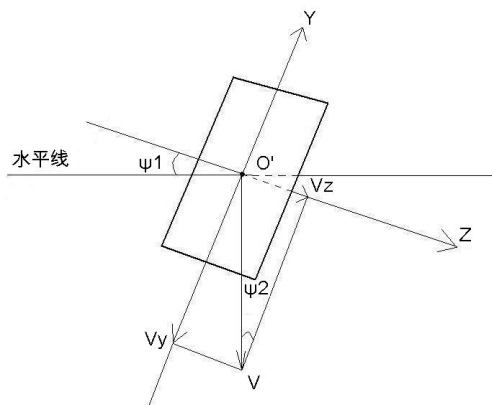


图 3-7 竖直状态的前后俯仰角度

$\psi_1$  为要求取的俯仰角度。根据几何定理， $\psi_1 = \psi_2$ ，因此，求取  $\psi_1$  转换为求取  $\psi_2$ 。当  $\gamma$  处于  $(-45, +45)$  之间时，采用如下公式计算：

$$\psi_1 = \psi_2 = \arctan \frac{V_z}{V_y} \quad (3-7)$$

当  $\gamma$  处于  $(135, 225)$  之间时，则采用如下公式：

$$\psi_1 = \psi_2 = -\arctan \frac{V_z}{V_y} \quad (3-8)$$

现在讨论手持设备为横向状态时的前后俯仰角度的计算方法，如图 3-8 所示。

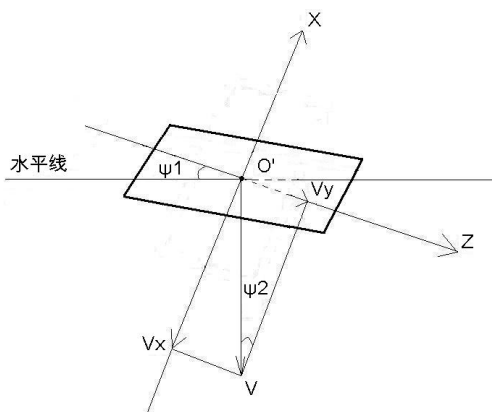


图 3-8 横向状态的前后俯仰角度

和计算竖直状态时的前后俯仰角度类似，当  $\gamma$  处于  $[45, 135]$  之间时，公式如下：

$$\psi_1 = \psi_2 = \arctan \frac{V_z}{V_x} \quad (3-9)$$

当  $\gamma$  处于[225, 270]和[-90, -45]之间时，公式如下：

$$\psi_1 = \psi_2 = -\arctan \frac{V_z}{V_x} \quad (3-10)$$

对  $\psi_1$  的抖动进行过滤，公式如下：

$$\psi = \psi_1 \times filterFactor + \psi \times (1 - filterFactor) \quad (3-11)$$

这时，最后计算得到的  $\psi$  就是需要的前后俯仰角度，取值范围是[-90, +90]。  
当摄像机仰视时， $\psi$  处于[-90, 0)范围，当摄像机俯视时， $\psi$  处于(0, 90]范围。

## 第四章 基于传感器的图像平面投影定位

### 4.1 三维空间到图像平面的投影模型

在物理学中，利用了小孔成像来解释投影成像模型。小孔成像是将带有小孔的遮挡板放在屏幕和物体之间。物体受到光照后产生漫射，一部分光线由物体射出，穿过中间的小孔投射到后面的屏幕上产生倒像。如图 4-1 所示。

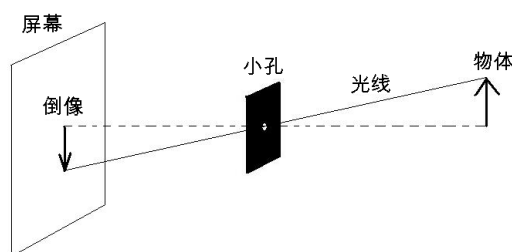


图 4-1 小孔成像

现在，将屏幕移至遮挡板和物体之间，此时小孔为投影中心，物体发出的光线都将聚焦于投影中心，并投影在屏幕上，其本质与小孔成像是一样的。而且此时，在屏幕上形成的影像是正像。如图 4-2 所示。

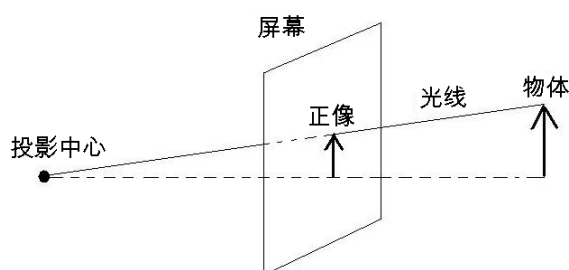


图 4-2 投影屏幕在投影中心和物体之间形成的投影模型

这就是需要利用的基本原理。现在将这个模型用另外一种图表示。在图 4-3 中，矩形方框表示图像平面，O 点为摄像机光心，射线  $OO'$  为摄像机光轴， $O'$  为摄像机光轴和图像平面的交点，位于图像平面中心。A、B、C 为周围的任意兴趣点， $A'$ 、 $B'$ 、 $C'$  分别为各兴趣点在图像平面的投影。兴趣点 A 和 C 由于同处于射线 OA 上，则在图像平面上的投影位置重合。

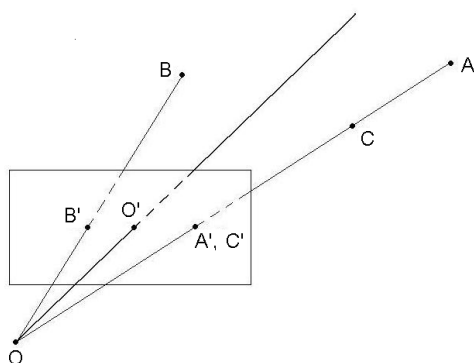


图 4-3 图像平面投影原理

由此可得出结论，兴趣点和摄像机光心的连线与图像平面的交点即为该兴趣点在图像平面上的投影位置。以下需要做的工作就是依据此原理，研究计算投影位置的方法。

## 4.2 图像平面投影坐标的计算

根据投影模型，不难发现，图像平面的宽和高分别与摄像机水平和垂直视角成一定比例。设以图像平面中心为原点的图像坐标系为  $C_{xy}$ ，以图像平面左上角为原点的图像坐标系为  $C_{uv}$ 。在  $C_{xy}$  下，兴趣点在图像平面上的投影点的坐标为  $(x, y)$ ，将投影点分别投射到  $C_{xy}$  的  $X$  轴和  $Y$  轴上，形成了两个投射点。将这两个投射点分别与摄像机光心进行连线，这两条直线会与摄像机的光轴形成两个夹角，分别为水平和垂直夹角。那么，该投影点坐标的  $x$  和  $y$  分别与这两个夹角形成的比例是等于宽高与水平垂直视角的比例。通过这个关系，只要获取兴趣点和摄像机光心的连线与摄像机光轴的水平垂直夹角，并已知图像平面的宽高和摄像机水平垂直视角，即可计算出  $C_{xy}$  下兴趣点在图像平面上的投影坐标  $(x, y)$ 。由于在程序实现中一般是使用  $C_{uv}$ ，因此，再将其转换成  $C_{uv}$  下的横纵坐标  $(u, v)$ 。下面分别讨论横纵坐标的计算方法。

### 4.2.1 图像平面投影横坐标 $u$ 的计算

为了表述方便，在图 4-4 中，兴趣点  $A$  的投影点  $A'$  正好位于图像平面  $C_{xy}$  下的  $X$  轴上，因此投影点  $A'$  也是其在  $X$  轴上的投射点。 $LngA$ 、 $LatA$ 、 $AltA$ 、 $LngO$ 、 $LatO$  和  $AltO$  分别为兴趣点  $A$  和手持设备当前的经纬度海拔信息； $\omega$  表示摄像机的方向角度； $\theta$  表示兴趣点  $A$  相对于点  $O$  偏离正北方向的角度； $\beta$  表示兴趣点  $A$

和点 O 连线与光轴 OO' 的水平夹角； $\alpha$  表示摄像机的水平视角；W 表示图像平面宽度。

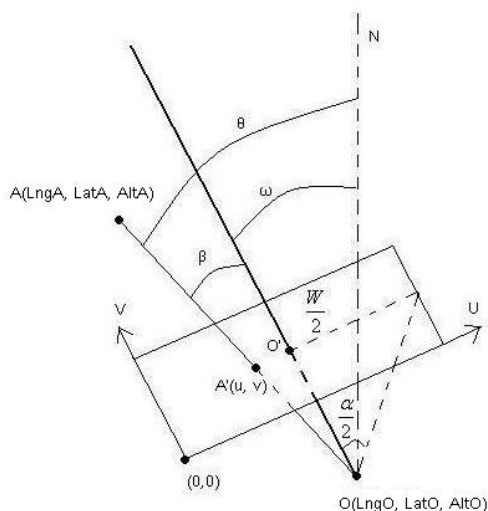


图 4-4 图像平面投影横坐标计算模型

先计算  $\theta$  的值，根据三角函数定理可得如下公式：

$$\theta = \arctan_4 \frac{\text{LngA} - \text{LngO}}{\text{LatA} - \text{LatO}} \quad (\text{four-quadrant inverse tangent}) \quad (4-1)$$

这里的反正切函数为 four-quadrant inverse tangent，其  $\theta$  取值范围是 $[-180, 180]$ ，为了后面的计算方便，需要将此范围转换成 $[0, 359]$ ，则当  $\theta$  的值处于 $[-180, 0)$ 的范围时， $\theta$  自加 360。公式如下：

$$\theta = \theta + 360, \quad -180 \leq \theta < 0 \quad (4-2)$$

再计算  $\beta$ ，公式如下：

$$\beta = \theta - \omega \quad (4-3)$$

$\omega$  的取值范围是 $[-\infty, +\infty]$ ，而  $\theta$  的取值范围是 $[0, 359]$ ，而  $\beta$  的取值范围根据测试，其绝对值的取值范围必须限定在 $[0, 180]$ ，因此，在计算式 4-3 之前，需要将  $\theta$  的取值范围扩大到 $[-\infty, +\infty]$ ，并且要与  $\omega$  的变化保持同步，使得两者差值  $\beta$  的绝对值取值范围首先保持在 $[0, 359]$ 。公式如下：

$$\theta = \theta + \text{rotCounter} \times 360 \quad (4-4)$$

现在利用式 4-3 计算得到的  $\beta$  绝对值取值范围在 $[0, 359]$ 。为了使其绝对值范

围缩小至 $[0, 180]$ ，其公式如下：

$$\beta = \begin{cases} \beta + 360, & \beta < -180 \\ \beta - 360, & \beta > 180 \end{cases} \quad (4-5)$$

水平夹角  $\beta$  与线段  $A'O'$  的长度存在固定的比例关系，该比例和  $(W/2)/(\alpha/2)$  相等。而线段  $A'O'$  的长度也等于点  $A'$  在  $C_{xy}$  下的坐标  $x$ ，则可得出公式：

$$x = \beta \times \frac{\frac{W}{2}}{\frac{\alpha}{2}} \quad (4-6)$$

转换成在  $C_{uv}$  下的横坐标：

$$u = x + \frac{W}{2} \quad (4-7)$$

此时，已得到计算图像平面投影横坐标的完整公式：

$$u = \beta \times \frac{\frac{W}{2}}{\frac{\alpha}{2}} + \frac{W}{2} \quad (4-8)$$

根据设备的不同， $W$  和  $\alpha$  的值也不同，作者采用的设备平台，其  $W$  的值为 320， $\alpha$  根据测试，其值设定在 30 度为佳。

#### 4.2.2 图像平面投影纵坐标 $v$ 的计算

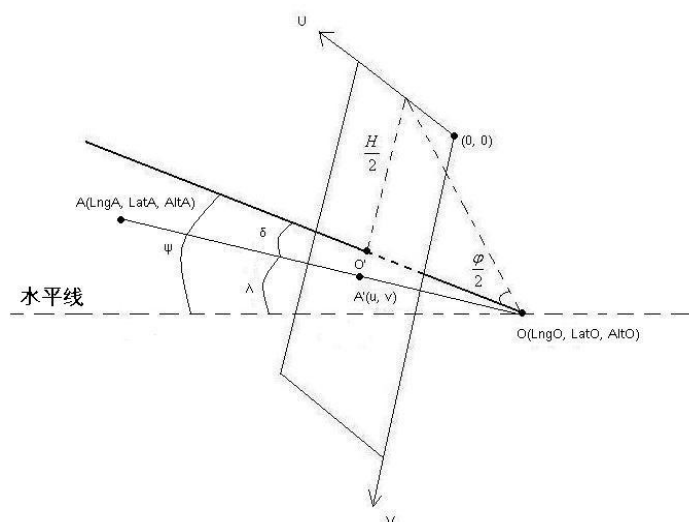


图 4-5 图像平面投影纵坐标计算模型

图像平面投影的纵坐标计算原理和横坐标相似，为了表述方便，如图 4-5 所示，兴趣点 A 的投影点 A' 正好位于图像平面  $C_{xy}$  下的 Y 轴上，因此投影点 A' 也是其在 Y 轴上的投射点。LngA、LatA、AltA、LngO、LatO 和 AltO 分别为兴趣点 A 和手持设备当前的经纬度海拔信息； $\psi$  表示摄像机的前后俯仰角度； $\lambda$  表示兴趣点 A 相对于手持设备点 O 偏离水平面的夹角； $\delta$  表示兴趣点 A 和点 O 连线与光轴  $OO'$  的垂直夹角； $\varphi$  表示摄像机的垂直视角；H 表示图像平面高度。

先求取  $\lambda$  的角度。当前已知兴趣点 A 的海拔 AltA 和手持设备的海拔 AltO，OA 的距离可以通过 GPS 两地距离公式计算获得。根据三角函数定理，可得如下公式：

$$\lambda = \arcsin \frac{AltA - AltO}{Dist_{OA}} \quad (4-9)$$

再计算  $\delta$ ，注意  $\psi$  的取值范围，当摄像机仰视时， $\psi$  处于  $[-90, 0)$  范围，当摄像机俯视时， $\psi$  处于  $(0, 90]$  范围。公式如下：

$$\delta = -\Psi - \lambda \quad (4-10)$$

垂直夹角  $\delta$  与线段  $A'O'$  的长度存在固定的比例关系，该比例和  $(H/2)/(\varphi/2)$  相等。而线段  $A'O'$  的长度也等于点 A' 在  $C_{xy}$  下的坐标 y，则可得出公式：

$$y = \delta \times \frac{\frac{H}{2}}{\frac{\varphi}{2}} \quad (4-11)$$

转换成在  $C_{uv}$  下的纵坐标：

$$v = y + \frac{H}{2} \quad (4-12)$$

计算图像平面投影纵坐标的完整公式为：

$$v = \delta \times \frac{\frac{H}{2}}{\frac{\varphi}{2}} + \frac{H}{2} \quad (4-13)$$

根据设备的不同，H 和  $\varphi$  的值也不同，作者采用的设备平台，其 H 的值为 480， $\varphi$  根据测试，其值设定在 50 度为佳。

### 4.2.3 姿态左右倾斜对横纵坐标的补偿计算

当手持设备进行左右倾斜的姿态变化时，若投影在图像平面上的兴趣点仍然保持原来的坐标位置上，则会使得兴趣点在图像平面上与目标物体发生位置偏离。因此需要对投影点的坐标随手持设备左右倾斜的姿态变化进行实时调整。如图 4-6 所示。

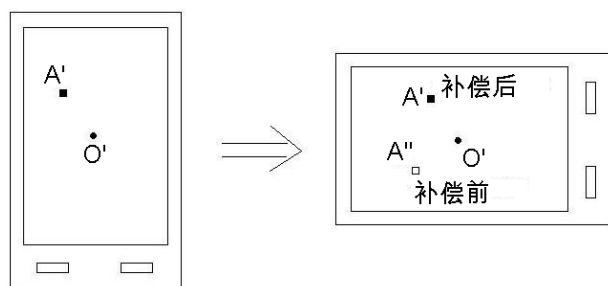


图 4-6 补偿前后的投影点位置

根据实验发现，当手持设备的左右倾斜角度为  $\gamma$  时，投影点可以绕以图像平面中心为圆心，投影点到图像平面中心的距离为半径，反向于左右倾斜的方向旋转  $\gamma$ ，以此达到补偿的目的。如图 4-7 所示。

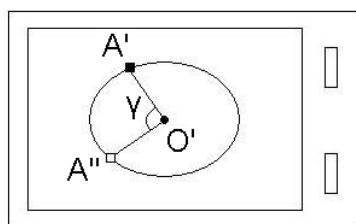


图 4-7 投影点位置的补偿原理

可以利用极坐标系来计算投影位置的变化。先对投影点进行  $C_{uv}$  到  $C_{xy}$  的坐标变换，再进行  $C_{xy}$  到以图像平面中心为极点的极坐标系的坐标变换，然后根据  $\gamma$  来改变投影点的角坐标，最后还原成  $C_{uv}$ 。

$C_{uv}$  到极坐标系的坐标变换如图 4-8 所示。

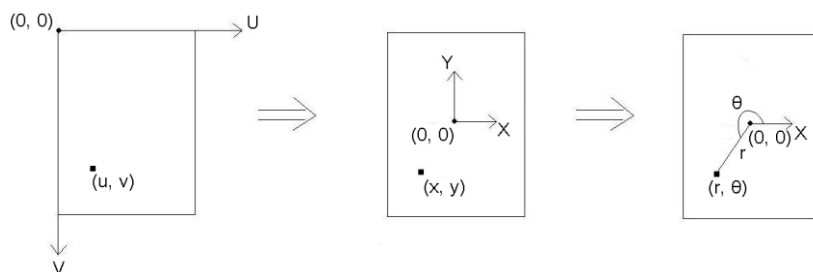


图 4-8  $C_{uv}$  到极坐标系的坐标变换

$C_{uv}$  到  $C_{xy}$  的坐标变换公式如下：



$$x = u - \frac{W}{2} \quad (4-14)$$

$$y = -v + \frac{H}{2} \quad (4-15)$$

$C_{xy}$  到极坐标系的坐标变换过程中, 首先计算投影点到极点的距离, 即半径坐标  $r$ , 公式如下:

$$r = \sqrt{x^2 + y^2} \quad (4-16)$$

再计算投影点与极点连线和极轴的角度, 即角坐标  $\theta$ , 公式如下:

$$\theta = \arctan\left(\frac{y}{x}\right) \text{ (four-quadrant inverse tangent)} \quad (4-17)$$

这里的反正切函数为 four-quadrant inverse tangent, 此时角坐标  $\theta$  的取值范围是  $[-180, +180]$ , 而在极坐标系当中, 角坐标  $\theta$  的取值范围应为  $[0, 359]$ 。因此, 当  $\theta$  处于  $[-180, 0)$  范围时,  $\theta$  自加 360。公式如下:

$$\theta = \theta + 360, \quad -180 \leq \theta < 0 \quad (4-18)$$

坐标变换完成, 现在根据  $\gamma$  来改变投影点的角坐标  $\theta$ 。 $\gamma$  与投影点绕极点反向于左右倾斜方向旋转的角度是相等的, 则  $\theta$  与  $\gamma$  的差值就是投影点新的角坐标  $\theta'$ 。公式如下:

$$\theta' = \theta - \gamma \quad (4-19)$$

投影点的半径坐标  $r$  不变, 此时已经获取了投影点在极坐标系下的新坐标  $(r, \theta')$ , 将其变换回  $C_{xy}$  下的坐标, 公式如下:

$$x' = r \times \cos \theta' \quad (4-20)$$

$$y' = r \times \sin \theta' \quad (4-21)$$

再变换回  $C_{uv}$  下的坐标, 公式如下:

$$u' = x' + \frac{W}{2} \quad (4-22)$$

$$v' = -(y' - \frac{H}{2}) \quad (4-23)$$

坐标 $(u', v')$ 便是补偿后的兴趣点在图像平面上的投影位置。

图 4-9 展示了本章所阐述的基于传感器的图像平面投影定位所得到的实验结果。可以看到，无论手持设备处于何种姿态，目标物体都能够在摄像机的输出视频流中被准确地对准，同时能够显示实时距离。

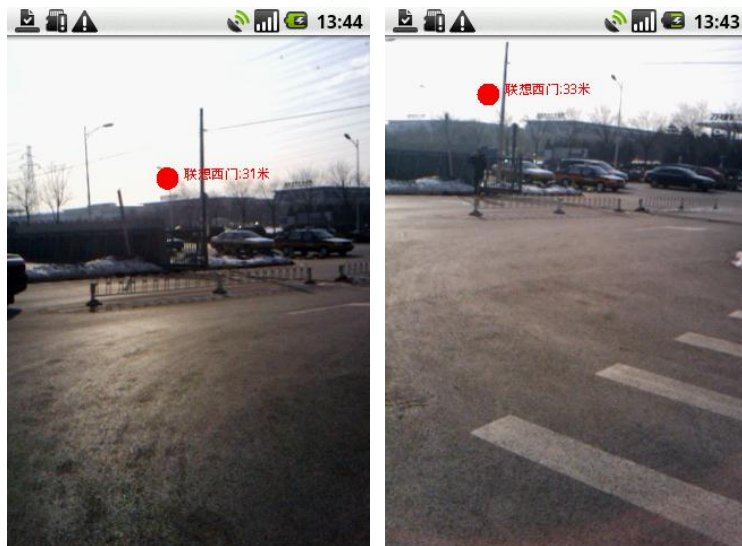


图 4-9 基于传感器的图像平面投影定位实验结果

## 第五章 基于传感器的三维虚拟路径叠加

在基于地图的位置导航应用中，当用户获取了一条从用户当前位置到目的地的导航路径时，会在地图上标示出来，如图 5-1 所示。



图 5-1 基于地图的导航路径显示

若采用增强现实技术,则可将三维虚拟导航路径叠加到真实道路上,从而增强用户的导航体验。基于地图的导航路径显示适用于远距离的观察,而采用增强现实技术来实施导航更有助于用户对当前位置周围环境近距离的观察。

### 5.1 虚拟路径的数据构造

空间中任意形状的线条可由若干个点依次相连构成。构成线条的点数越多，该线条的形状越细腻。如图 5-2 所示。

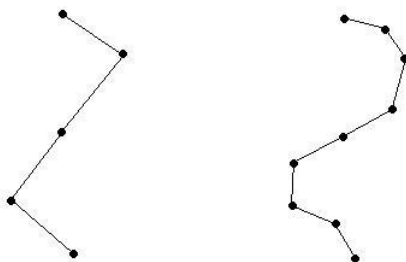


图 5-2 线条和点的关系

将真实道路分解为若干个点,然后将这些点由虚拟的线段依次相连,就可以构成一条虚拟路径,且这条虚拟路径能够完全地和真实道路重合。利用卫星定位

可以采集到真实道路上任意一点的经纬度及海拔位置信息，并且根据需要来决定被采集点的数量和密度。表 5-1 为一条真实道路上 8 个采集点的位置信息。

表 5-1 真实道路采集点的位置信息

采集点	经度	纬度	海拔
1	116.29382908344269	40.054001212120056	44.0
2	116.29333019256592	40.05365788936615	43.0
3	116.29286348819733	40.05337357521057	41.0
4	116.29228949546814	40.05313754081726	40.0
5	116.29183351993561	40.05293369293213	38.0
6	116.29172623157501	40.052729845047	36.0
7	116.29188179969788	40.052273869514465	38.0
8	116.29200518131256	40.05170524120331	43.0

## 5.2 实景空间到 OpenGL 空间的变换关系

虚拟路径是通过 OpenGL 图形库进行渲染的。为了能够将虚拟路径叠加到摄像机输出视频流中的真实道路上，需要解决实景空间到 OpenGL 空间的变换。变换过程由 GPS 坐标系变换到以用户为原点的用户 OpenGL 坐标系，再变换到以摄像机为原点的摄像机 OpenGL 坐标系，最后经过投影变换将虚拟物体投影到图像平面上。

### 5.2.1 GPS 坐标系到用户 OpenGL 坐标系的变换

GPS 坐标系中，经纬度为角度，海拔为米制单位，而 OpenGL 坐标系使用逻辑长度单位，要使两者进行变换，必须对两者的度量单位建立对应关系。OpenGL 使用的逻辑长度是一种相对长度，可以设定为任意物理长度单位，这里设定 OpenGL 中 1 个单位长度等效于 1 米。现在需确立经纬度与米制单位之间的换算关系，先通过 GPS 两地距离公式来计算两采集点之间的米制距离，再通过平面两点距离公式计算这两采集点的角度距离，这两个值之间的比例关系就是米制单位与经纬度之间的对应关系。为了使计算结果更加准确，将表 5-1 中的 8 个采集点依次两两计算，得到了 7 组结果，如表 5-2 所示。

表 5-2 米制单位与经纬度比例关系统计结果

采集点	GPS 两地距离公式	平面两点距离公式	比例关系
1 和 2	57.164058305375185	0.00060560929672036107	1:0.00001059423202
2 和 3	50.824639210451501	0.00054648651101796828	1:0.00001075239332
3 和 4	55.520440149962468	0.00062062862324020480	1:0.00001117838082
4 和 5	44.994721726913433	0.00049946736283797816	1:0.00001110057677
5 和 6	24.464549158327682	0.00023035787937909961	1:0.00000941598710
6 和 7	52.461351073086959	0.00048178327816452825	1:0.00000918358503
7 和 8	64.166567390032156	0.00058186010268088660	1:0.00000906796368

将这 7 组结果取平均值得 1:0.00001018473125，这就是 OpenGL 和经纬度之间长度单位的比例关系。

GPS 坐标系属于左手坐标系，纬度轴始终朝向正北方向，用户 OpenGL 坐标系属于右手坐标系，Z 轴始终朝向正南方，如图 5-3 所示。

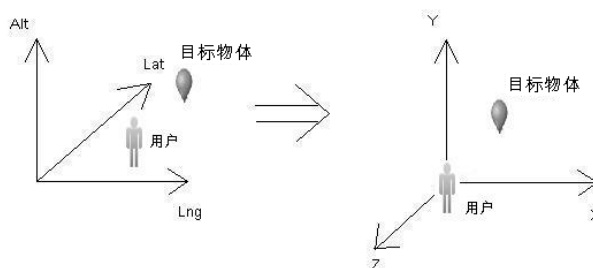


图 5-3 GPS 坐标系到用户 OpenGL 坐标系的变换

根据坐标系间的变换规则，可得出下面的变换矩阵：

$$M_1 = \begin{bmatrix} \frac{1}{k} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{1}{k} & 0 \\ -\frac{LngO}{k} & -AltO & \frac{LatO}{k} & 1 \end{bmatrix} \quad (5-1)$$

$k = 0.00001018473125$

其中，LngO、AltO 和 LatO 为用户的经纬度及海拔。

### 5.2.2 用户 OpenGL 坐标系到摄像机 OpenGL 坐标系的变换

摄像机可以具有任意姿态，使得目标物体的位置相对于摄像机姿态在实时改变，需要将这些目标物体变换到摄像机坐标系下。摄像机坐标系仍然使用 OpenGL 坐标系，且原点与用户 OpenGL 坐标系的原点一样，由于摄像机具有任意姿态，

所以，相对于用户 OpenGL 坐标系，摄像机 OpenGL 坐标系在三个轴上的旋转自由度是没有限制的，如图 5-4 所示。

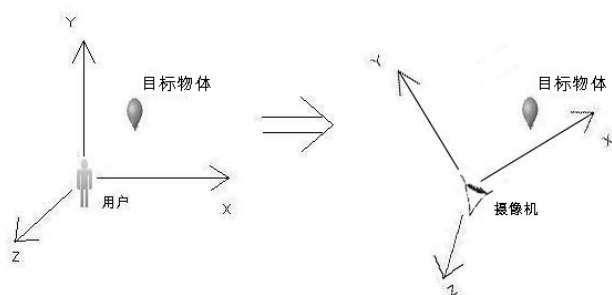


图 5-4 用户 OpenGL 坐标系到摄像机 OpenGL 坐标系的变换

这样的坐标系变换可以使用一个旋转变换矩阵来处理，表示如下：

$$M_2 = \begin{bmatrix} \cos \gamma & \sin \gamma & 0 & 0 \\ -\sin \gamma & \cos \gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \psi & \sin \psi & 0 \\ 0 & -\sin \psi & \cos \psi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \omega & 0 & -\sin \omega & 0 \\ 0 & 1 & 0 & 0 \\ \sin \omega & 0 & \cos \omega & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5-2)$$

$$= \begin{bmatrix} \cos \gamma \cos \omega + \sin \gamma \sin \psi \sin \omega & \sin \gamma \cos \psi & -\cos \gamma \sin \omega + \sin \gamma \sin \psi \cos \omega & 0 \\ -\sin \gamma \cos \omega + \cos \gamma \sin \psi \sin \omega & \cos \gamma \cos \psi & \sin \gamma \sin \omega + \cos \gamma \sin \psi \cos \omega & 0 \\ \cos \gamma \sin \omega & -\sin \psi & \cos \gamma \cos \omega & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

其中， $\psi$  为摄像机前后俯仰角度， $\omega$  为摄像机方向角度， $\gamma$  为摄像机左右倾斜角度。

最终虚拟物体要在图像平面上显示出来，利用式 2-8 来构造投影变换矩阵。这里只讨论采用类似于人眼观察的标准镜头的摄像机设备，其视角广度在 45 度至 50 度之间，因此 fovy 可设置为 50 度。宽  $w$  和高  $h$  可由图像平面的当前属性确定。近截面  $n$  可设置为 1，远截面  $f$  可设置为 1000。

下图展示了利用上述方法在 PC 平台上进行的虚拟路径叠加模拟实验。将地板看成一条道路，在输出视频流中叠加一条虚拟路径，可见虚拟路径能够准确地叠加在目标物体上。

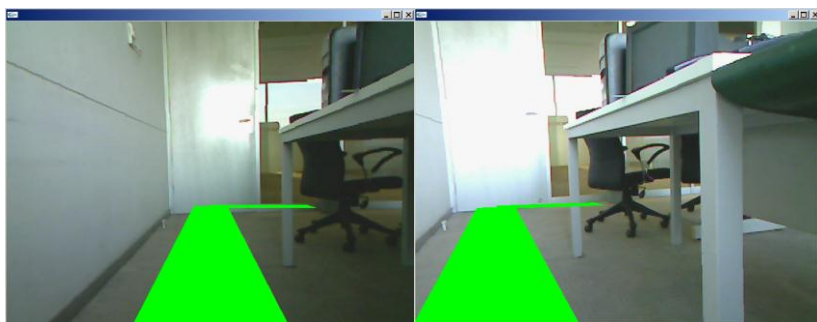


图 5-5 虚拟路径叠加模拟

### 5.3 虚拟路径的实时生成

将各个采集点相连，在摄像机 OpenGL 坐标系下进行渲染，形成一条虚拟路径。用户的移动会使其与各个采集点的相对位置发生改变，使得构成的虚拟路径在摄像机 OpenGL 坐标系下产生平移，模拟出用户与虚拟路径之间真实的相对运动。如图 5-6 所示。

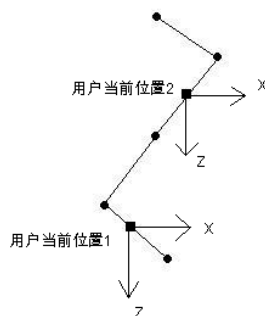


图 5-6 用户与虚拟路径之间的相对位置

#### 5.3.1 虚拟路径段的提取

由于虚拟路径的长度较长，弯道较多，以及周围环境遮挡的问题，若将虚拟路径完整渲染出来，会使得渲染图像杂乱、叠加效果差，因此每次仅渲染一部分路径段。经测试，提取离用户当前位置最近的一个采集点及其后最多两个采集点所构成的虚拟路径段效果最佳。

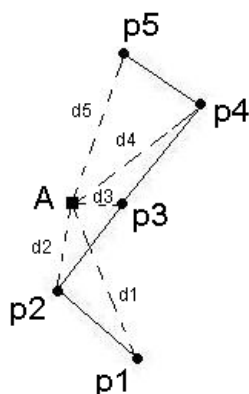


图 5-7 用户与各采集点的距离

如图 5-7 所示，一条真实道路上有采集点  $p_1$ 、 $p_2$ 、 $p_3$ 、 $p_4$  和  $p_5$ ，已知用户当前位置  $A$  和这 5 个采集点的经纬度及海拔，利用 GPS 两地距离公式依次求取  $A$  到各采集点的距离  $d_1$ 、 $d_2$ 、 $d_3$ 、 $d_4$  和  $d_5$ ，找出其中最小值  $d_3$ ，其对应的  $p_3$  为最近采集点， $p_3$ 、 $p_4$  和  $p_5$  构成提取的虚拟路径段。对这个虚拟路径段进行坐标系变换，使其变换到摄像机 OpenGL 坐标系下，准备渲染。

### 5.3.2 虚拟路径段的渲染

用矩形图元来表示两个采集点之间连接的路段，通过依次渲染这些矩形图元形成虚拟路径段。

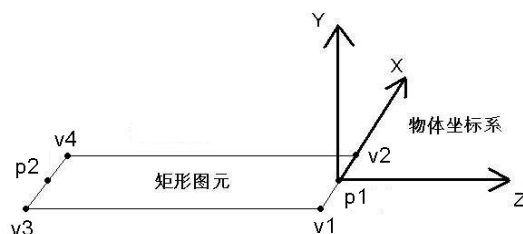


图 5-8 矩形图元的物体坐标系

如图 5-8 所示，在 OpenGL 中定义顶点  $v_1$ 、 $v_2$ 、 $v_3$  和  $v_4$  来构建这个矩形图元，其中边  $v_1v_2$  和边  $v_3v_4$  均为 2 个逻辑长度。两个采集点  $p_1$  和  $p_2$  分别位于矩形图元两边的中心，其中  $p_1$  同时位于其物体坐标系的原点位置。对矩形图元的操作都是基于该物体坐标系。

对每一个矩形图元进行渲染时分两个步骤，步骤一，根据  $p_1$  和  $p_2$  在 OpenGL 空间的距离来确定该矩形图元的长度。在物体坐标系下， $v_1$  和  $v_2$  的坐标是确定的，利用三维空间两点距离公式计算出  $p_1$  和  $p_2$  的距离，确定  $v_3$  和  $v_4$  的坐标，使边  $v_1v_3$



和边  $v_2v_4$  的长度等于其距离。

步骤二，根据  $p_1$  和  $p_2$  之间的连线与正北方向的偏角来确定矩形图元绕 Y 轴的旋转角度，以及与地平面的偏角来确定矩形图元绕 X 轴的旋转角度，使矩形图元的姿态与  $p_1$  和  $p_2$  的连线保持一致。

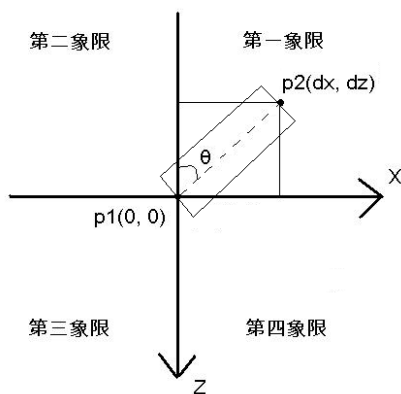


图 5-9 矩形图元绕 Y 轴的旋转角度

如图 5-9 所示，建立一个以  $p_1$  为原点的 XZ 坐标系， $p_2$  在 XZ 坐标系下的坐标(dx, dy)为两者在用户 OpenGL 坐标系下的坐标差值， $\theta$  为矩形图元绕 Y 轴的旋转角度。XZ 坐标系的 Z 轴方向朝下，与 OpenGL 右手坐标系相对应，为了表述方便，象限仍然采用笛卡尔坐标系的规定。根据  $p_2$  所在坐标轴及象限的不同， $\theta$  的计算方法也有所不同，公式如下：

$$\theta = \begin{cases} 90, & dx > 0 \text{ 且 } dz = 0 \\ -90, & dx < 0 \text{ 且 } dz = 0 \\ \arctan \frac{dx}{dz}, & dz < 0 \\ \arctan \frac{dx}{dz} - 180, & dx < 0 \text{ 且 } dz > 0 \\ \arctan \frac{dx}{dz} + 180, & dx \geq 0 \text{ 且 } dz > 0 \end{cases} \quad (5-3)$$

计算矩形图元绕 X 轴的旋转角度时，与计算  $\theta$  类似，如图 5-10 所示。

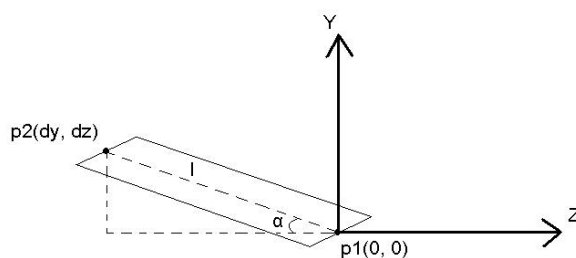


图 5-10 矩形图元绕 X 轴的旋转角度

建立一个以  $p_1$  为原点的 YZ 坐标系， $p_2$  在 YZ 坐标系下的坐标  $(dy, dz)$  为两者在用户 OpenGL 坐标系下的坐标差值， $l$  为  $p_1$  和  $p_2$  在 OpenGL 空间的距离， $\alpha$  为矩形图元绕 X 轴的旋转角度。 $\alpha$  的计算公式如下：

$$\alpha = \arcsin \frac{dy}{l} \quad (5-4)$$

最后利用  $\theta$  和  $\alpha$  对矩形图元进行相应的旋转。图 5-11 展示了利用表 5-1 所列出的 8 个采集点，通过上述基于传感器的三维虚拟路径叠加所得到的实验结果。可以看到，虚拟路径能够根据用户当前位置和摄像机的姿态准确地叠加在真实道路上。

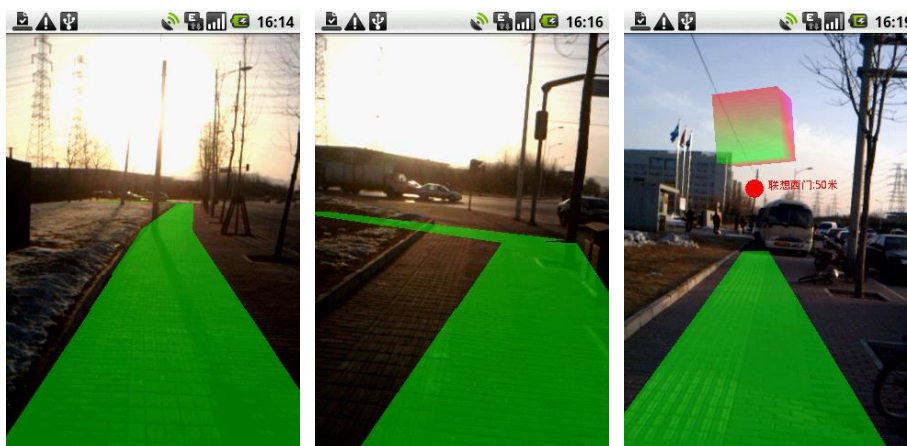


图 5-11 基于传感器的三维虚拟路径叠加实验结果

## 5.4 阴影渲染

在增强现实技术中，虚拟物体与真实环境的融合度越高，其显示效果越好，阴影可以作为主要手段来增强虚实融合度。

阴影在物理学上被定义为光线被不透明物体阻挡而产生的黑暗范围。在计算

机图形学上，已经有很多算法来处理对阴影的渲染，常用的算法有平面阴影（Planar Shadows）<sup>[44]</sup>、阴影贴图（Shadow Mapping）<sup>[45]</sup>以及阴影锥（Shadow Volumes）<sup>[46]</sup>。针对目前的软硬件开发平台和应用需求，采用平面阴影算法。

平面阴影算法最早由 Blinn 在 1988 年发表的论文《Me and My (Fake) Shadow》<sup>[44]</sup>中提出，这是阴影生成最快的算法，但只适合在平面上进行投影。该算法的基本思想是通过光源的位置与投影平面的方程构造一个阴影投射变换矩阵（Shadow Caster Matrix），遮光物各顶点坐标与该阴影投射变换矩阵相乘，求出遮光物在投影平面上的位置，并渲染形成阴影效果。

如图 5-12 所示，L 为一个点光源，V 为遮光物的一个点，S 为点 V 在投影平面  $n \cdot P + D = 0$  上的阴影投射点。

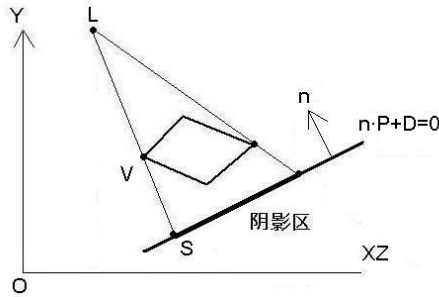


图 5-12 平面  $n \cdot P + D = 0$  上的投影

$n$  为投影平面的法向量(A, B, C),  $P$  为投影平面上任意一点(x, y, z),  $P_0$  为投影平面上一确定的点( $x_0, y_0, z_0$ )。平面的点法式方程如下：

$$\begin{aligned} n \cdot P + D &= 0 \\ D &= -n \cdot P_0 \end{aligned} \quad (5-5)$$

已知点光源 L 的坐标( $L_x, L_y, L_z$ )和点 V 的坐标( $V_x, V_y, V_z$ )，需要求取阴影投射点 S 的坐标( $S_x, S_y, S_z$ )，公式如下：

$$S = L - \frac{D + n \cdot L}{n \cdot (V - L)} (V - L) \quad (5-6)$$

将式 5-6 转换成一个阴影投射变换矩阵：

$$M_{shadow} = \begin{bmatrix} n \cdot L + D - L_x A & -L_y A & -L_z A & -A \\ -L_x B & n \cdot L + D - L_y B & -L_z B & -B \\ -L_x C & -L_y C & n \cdot L + D - L_z C & -C \\ -L_x D & -L_y D & -L_z D & n \cdot L \end{bmatrix} \quad (5-7)$$

根据该阴影投射变换矩阵，只要已知投影平面的点法式方程和点光源的坐标就可以构造出该变换矩阵，而在三维空间中，平面可由不共线的三点确定。设已知三维空间中不共线三点  $p_1$ 、 $p_2$ 、 $p_3$ ，如图 5-13 所示。

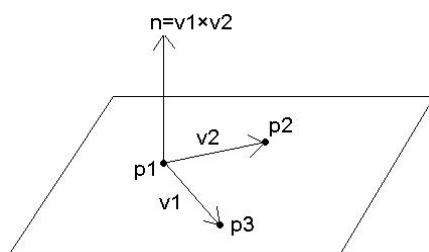


图 5-13 平面法向量的计算

根据向量减法的性质， $p_3-p_1$  可求得向量  $v_1$ ， $p_2-p_1$  可求得向量  $v_2$ 。根据向量外积的性质，对向量  $v_1$  和向量  $v_2$  做外积运算，可以求取同时垂直于这两个向量的第三个向量，该向量就是平面的法向量  $n$ 。由于法向量是单位向量，所以，需要对这个法向量  $n$  进行正规化。再根据式 5-5，利用法向量和一个已知点求取  $D$ 。此时就得到了该平面的点法式方程，最后与点光源的坐标构造阴影投射变换矩阵。

在对遮光物进行阴影渲染时，首先利用这个阴影投射变换矩阵来对坐标系进行变换，使得渲染的物体都被“压扁”在这个投影平面上。

图 5-14 展示了对虚拟路径及其它虚拟物体增加阴影渲染后的实验结果。可以看到，阴影渲染加深了虚拟路径及其它虚拟物体与真实环境之间的位置关系，对虚实融合度带来了很大的提高。



图 5-14 阴影渲染效果

## 第六章 基于传感器增强现实技术的 LBS 应用系统

最后，本文结合以上的讨论，利用传感器增强现实技术来实现一种运行在 Android 操作系统手持设备上的地理位置服务（LBS）应用系统。首先对 Android 操作系统进行介绍，再阐述该应用系统中用户交互设计思想，最后讨论该应用系统的架构及各个模块的设计与实现。

### 6.1 Android 操作系统及开发介绍

Android 是 2007 年 Google 公司推出的一款基于 Linux 内核的开放操作系统平台，主要是用于手持设备，如手机、平板电脑等。目前，许多移动终端设备商都相继推出了各自基于 Android 的产品。作为 Android 最大的特点：开放性，任何移动终端设备商都可以将 Android 进行修改，从内核到用户界面，使其能够在各自的硬件平台上运行。

#### 6.1.1 Android 操作系统架构

Android 操作系统架构分为应用程序、应用程序框架、系统库、运行时和 Linux 内核五个部分，如图 6-1 所示：

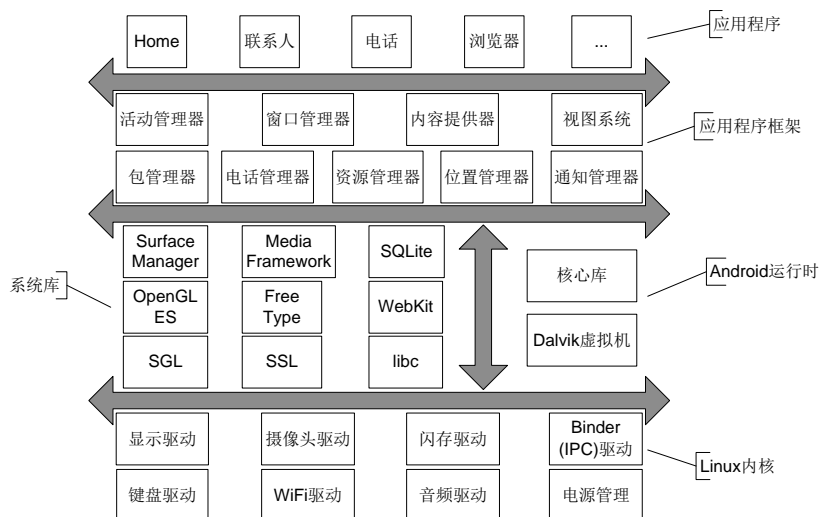


图 6-1 Android 操作系统架构

应用程序：应用程序都包含在这一层。在 Android 中，即使预装的应用程序，

也可以卸载或替换成普通开发人员编写的应用程序，相互是平等的。

**应用程序框架：**这一层是 Android 应用程序开发的基础，包含了 Android 几乎所有的功能。由于 Android 本身具有开放性，这些模块也是可以由开发人员修改、添加和替换的。

**系统库：**这一层是对应用程序框架的支撑，包含了各种可以调用的接口，来实现各种特定的功能。

**Android 运行时：**应用程序在 Android 上运行时，是通过 Android 运行时来运行的。Android 运行时包括两个部分，核心库和 Dalvik 虚拟机。核心库包含两个部分，一部分是应用程序需要调用的功能函数，另一部分是 Android 本身的核心库，如 android.os、android.net、android.media 等。Dalvik 虚拟机是一种基于寄存器的 Java 虚拟机，专门针对手持设备而设计的。与标准的 Java 虚拟机不同，一个标准的 Java 虚拟机要同时执行多个应用程序，而在 Android 上，每一个应用程序都有一个自己的 Dalvik 虚拟机来执行，所以，应用程序都有一个自己的进程。

**Linux 内核：**由于采用了 Linux 内核，才使得 Android 的开放性得到保障。Linux 内核中包含了一些核心系统服务，如安全、内存管理、进程管理、网络栈以及驱动模型。这一层其实是给其上的软件层和其下的硬件层建立了一个抽象层，使开发人员不需要关心硬件的具体细节。作为移动终端设备商，则需要对这一层进行改动，使其能够在自己的设备上部署运行。

### 6.1.2 基于 Android 操作系统的开发

在 Android 操作系统上进行开发时，主要使用 Java 程序语言。在搭建开发环境时，需要安装用于 Java 程序开发的 JDK 和 Eclipse 开发工具，同时需要安装针对 Android 应用程序开发的 Android SDK 和 Eclipse 插件 ADT(Android Development Tool)。Android 还提供 NDK(Native Development Kit)，是对 Java 中的 JNI(Java Native Interface)技术的封装。利用 Android NDK 来编译 C/C++ 函数库，让 Java 程序调用这些函数库，以此来获取更快的运行速度。所以，还可以选择安装 Android NDK 和 Eclipse 插件 CDT(C/C++ Development Tooling)。

由于是嵌入式开发，需要进行交叉编译。开发过程中，先在 PC 平台上进行编码，然后通过交叉编译，将编译好的可执行文件安装至 Android 手持设备上运行和调试。

一个 Android 应用程序由不同的组件构成，分别是 Activity、Service、

BroadcastReceiver 和 ContentProvider，这些组件之间靠 Intent 来实现通信。应用程序不需要将所有的这些组件都包含进来，但至少必须有一个 Activity 组件，因为这相当于应用程序的启动入口。这些组件需要在 AndroidManifest.xml 配置文件中声明。

Activity 可以看作是应用程序的一个界面，而应用程序会有很多个界面，所以，一个应用程序可以包含多个 Activity，而作为启动入口的 Activity 会在 AndroidManifest.xml 中声明。一个应用程序的生命周期是被 Activity 的生命周期所影响。Activity 在程序实现中是一个系统组件类，其中有 onCreate()、onStart()、onResume()、onPause()、onStop()、onRestart 和 onDestroy()这七个成员函数需要实现，这些成员函数对应着 Activity 生命周期中的各个状态。图 6-2 展示了一个 Activity 的生命周期的各个状态。

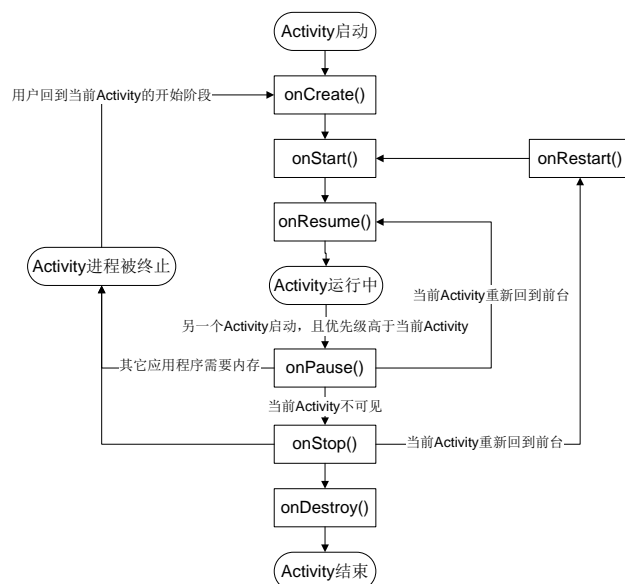


图 6-2 Activity 的生命周期

Android 应用程序工程一般包含 src 程序代码目录、gen 目录、Android API 目录、jni 目录、libs 目录、res 资源目录和 AndroidManifest.xml 配置文件，如图 6-3 所示。

**src 程序代码目录：**该目录中包含了 Android 应用程序所需的全部 Java 程序代码文件，都是按照 Java 程序中对包的结构来部署。

**gen 目录：**该目录中只有一个 R.java 的文件，是由 ADT 自动生成的。R.java 中定义了一个 R 类，其中定义了 res 资源目录中存放的所有资源所对应的成员变量，如用户界面布局、位图、字符串等。在程序中若要使用 res 资源目录中的

资源，则只需通过 R 类来调用对应的成员变量。

**Android API 目录：**该目录中存放的是 Android 操作系统中所有系统库和核心库的对象和接口，使得应用程序能够从该目录中调用所需要的对象和接口。

**jni 目录：**该目录中存放了用 C/C++编写的函数代码，利用 Android NDK 可以将其编译成 C/C++函数库，供 src 程序代码目录中的 Java 程序调用。

**libs 目录：**C/C++编写的函数代码通过 Android NDK 编译成的函数库就存放在该目录中，函数库是以 Linux 操作系统下 C/C++库文件.so 为后缀名。

**res 资源目录：**该目录中默认的有 drawable、layout 和 values 三个子目录。在 drawable 目录中存放一些应用程序需要用到到位图文件。layout 目录主要负责用户界面的布局，里面存放的都是一些 XML 格式的用户界面布局文件，类似于编写 Web 文件使用的 HTML 文件。values 目录中主要存放一些 XML 格式的参数描述文件，如定义字符串、颜色、格式等资源 and 值。

**AndroidManifest.xml 配置文件：**该文件是系统的文件，在里面设置了应用程序中各个组件的属性，以及对应各个组件的实现类。同时还定义了该应用程序需要启动的硬件设备及其属性，如 GPS 接收器、摄像机等。在该配置文件中还有一个很重要的内容是 Intent 过滤器，用于设定某个 Activity 是否是程序入口、如何启动、何时启动等。

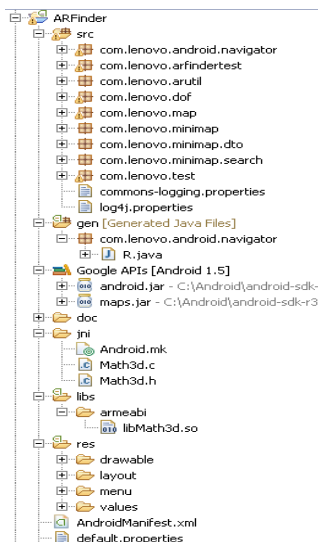


图 6-3 Android 应用程序工程结构

## 6.2 用户交互设计

本应用系统将增强现实技术与地理位置服务相互结合，对传统的地理位置服



务是一种创新，带给用户全新的交互体验。

应用系统分为两种模式，分别为兴趣点模式和导航模式，其中兴趣点模式为默认模式。在运行流程设计上，当系统启动时，首先进行卫星定位，此时处于定位状态，若定位超过 3 分钟，则询问用户重新定位还是取消，若选择取消，系统将停止在实景界面不作任何操作，仅有摄像机输出视频流，没有兴趣点。若定位成功，进入默认的兴趣点模式，同时在后台通过网络端搜索远程服务器中的距离用户三公里范围内的所有类别兴趣点。兴趣点类别包括“全部”、“餐厅”、“购物”、“公交”和“加油站”，用户可以选择其它类别重新搜索兴趣点。当搜索成功后，在输出视频流上只显示最多 20 个搜索到的兴趣点，可以通过列表的形式浏览所有兴趣点。用户可以选择某个兴趣点，查看其简略信息和详细信息，也可以进入针对该兴趣点的导航模式。图 6-4 展示了应用系统的用户交互流程。

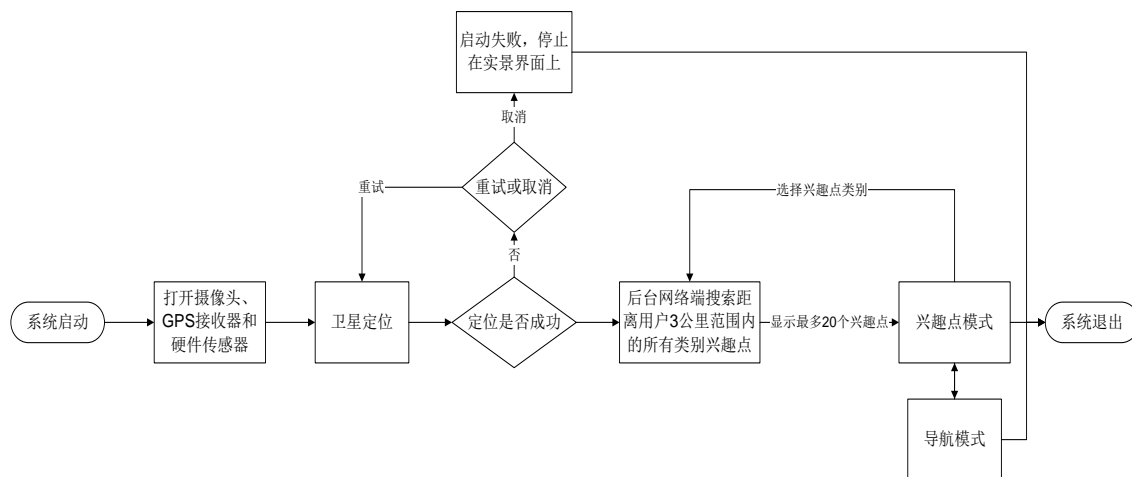


图 6-4 应用系统的用户交互流程

### 6.2.1 兴趣点模式

兴趣点模式是应用系统中的默认模式，在系统启动时首先进入该模式。在兴趣点模式下，用户可以在摄像机输出视频流上观察搜索到的兴趣点，并且这些兴趣点准确地叠加在目标实体上。兴趣点的显示方式采用气泡图形，根据兴趣点与用户距离的近中远，分别采用三种不同的气泡图标。定义近距离范围是 0 到 200 米，中距离范围是 200 到 1000 米，远距离范围是 1000 到 3000 米。随着距离从近到远，气泡图标也从大到小、渐进透明，如图 6-5 所示。



图 6-5 兴趣点气泡图标

根据目标实体的位置分布，在屏幕上所显示的部分兴趣点气泡会发生重叠的情况。为了提示用户某个区域的兴趣点气泡重叠了，在兴趣点气泡重叠区域右上角显示重叠兴趣点气泡的数量。

为了增强透视效果，当手持设备做前后俯仰姿态变化时，兴趣点在屏幕纵向上的摆动幅度也会根据兴趣点与用户距离的近中远来决定，离用户越远，幅度越大，如图 6-6 所示。

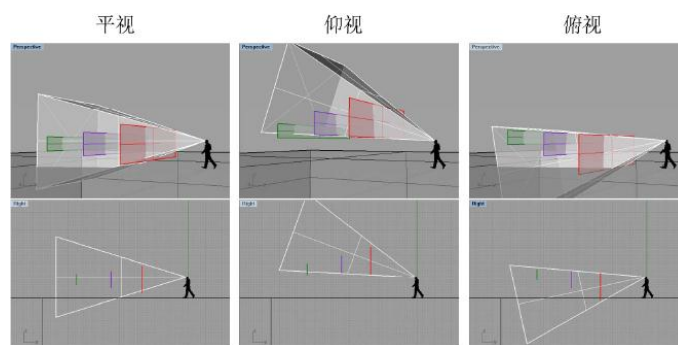


图 6-6 兴趣点在纵坐标上的摆动幅度

用户通过触摸兴趣点气泡来进行兴趣点的选取操作，被选取的兴趣点气泡会由原来的蓝色变成黄色产生高亮显示。若所触摸兴趣点气泡是重叠的，则会使重叠的气泡散开，再分别触摸选取相应的兴趣点气泡。触摸兴趣点气泡后会在屏幕上方弹出一个兴趣点信息卡来查看该兴趣点的简略信息，简略信息包括兴趣点名称、地址、与用户之间的实时距离和电话，在该兴趣点信息卡的右边设计一个导航按钮用于启动针对该兴趣点的导航模式。用户触摸兴趣点信息卡会显示该兴趣点的详细信息，除了简略信息所包含的内容以外，还增加了对该兴趣点的详细介绍。

由于采用手指触摸兴趣点气泡的方式会造成用户一定的不便，特别是在长时间的情况下。为了将用户手指解脱出来，在用户交互设计上，在屏幕中心叠加一块对焦区，当有兴趣点气泡落入该对焦区时，会自动选取该兴趣点气泡。在实际测试中，这种兴趣点气泡自动选取方式在很大程度上增强了用户体验的舒适度。图 6-7 展示了兴趣点模式下的用户交互流程。

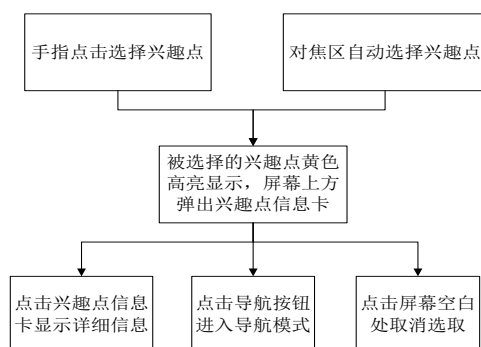


图 6-7 兴趣点模式的用户交互流程

用户在观察兴趣点的同时，也需要判别当前的方向，所以，在屏幕顶部添加一个方向标尺，来告知用户当前的方向。在屏幕底部设置按钮，触摸按钮可以列表形式查看所有兴趣点，包含了每个兴趣点的简略信息和导航按钮。兴趣点模式、详细信息和所有兴趣点列表的用户界面设计图请参见图 6-8。



图 6-8 兴趣点模式、详细信息和所有兴趣点列表的用户界面设计图

### 6.2.2 导航模式

用户可以通过三种方式进入导航模式，一是兴趣点信息卡上的导航按钮；二是兴趣点详细信息中的导航按钮；三是所有兴趣点列表中的导航按钮。在导航模式中，仍然采用类似于兴趣点模式的二维图标显示方式，在摄像机输出视频流中叠加一个导航标注点，标注当前导航段的目标点区域和导航路径终点区域。因此导航标注点的图标也分为导航段目标点和导航路径终点，如图 6-9 所示。



图 6-9 左图为导航段目标点图标，右图为导航路径终点图标

用户可以根据该导航标注点在摄像机输出视频流中的位置朝真实环境中所对应的位置方向移动，来实现导航。

一条导航路径由若干条导航段组成，每一条导航段都有一个目标点，最后一个导航段的目标点即为终点。当用户选择进入针对某兴趣点的导航模式时，系统会搜索一条从用户当前位置到该兴趣点的导航路径，搜索成功后，会显示一块导航路径面板，上面会列出每一条导航段的行走方式，以及导航路径终点的信息，并且第一条导航段会高亮显示。在导航路径面板上，用户可以选择确认导航、取消导航或重新搜索。

在进行导航时，系统会首先检测用户当前处于导航路径中的哪一条导航段，并将导航标注点叠加在摄像机输出视频流中该导航段的目标点位置上。用户可以通过实景朝该目标点位置移动，当用户进入该目标点 50 米半径区域时，系统会震动手持设备来提示用户，并且自动切换至下一个导航段，直到用户到达导航路径终点区域。在用户界面设计上，屏幕顶部添加了一个与兴趣点模式一样的方向标尺，来告知用户当前的方向。同时，在屏幕中偏上的位置还设计了一个导航信息卡，用于显示当前导航标注点的方向、与用户的实时距离和导航终点到达提示。其中导航标注点的方向是指若导航标注点位于屏幕范围内，则在导航信息卡中显示直行箭头；若导航标注点超出屏幕范围左侧，则在导航信息卡中显示左转箭头；导航标注点超出屏幕范围右侧，则在导航信息卡中显示右转箭头。触摸导航信息卡则显示导航路径面板，并且当前的导航段会高亮显示。这样的设计是为了增强用户的体验效果。图 6-10 展示了导航模式下的用户交互流程。

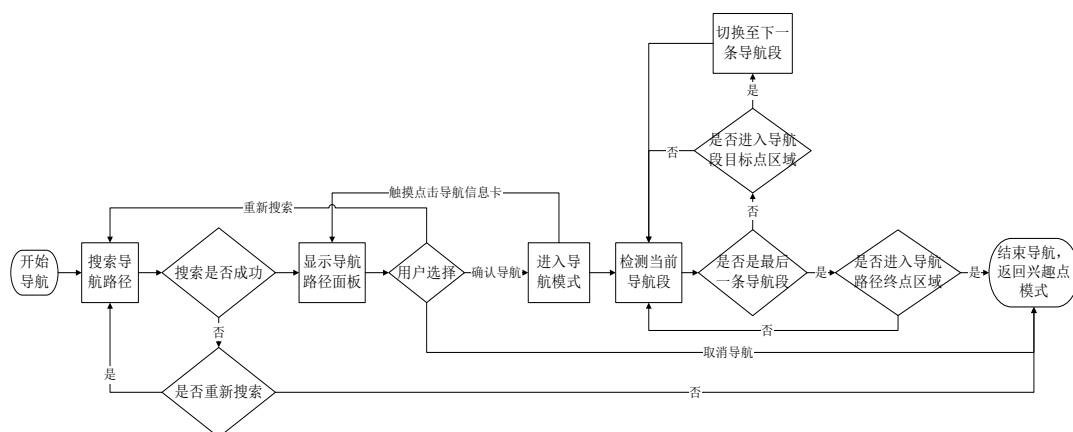


图 6-10 导航模式的用户交互流程

导航模式的用户界面设计图请参见图 6-11。

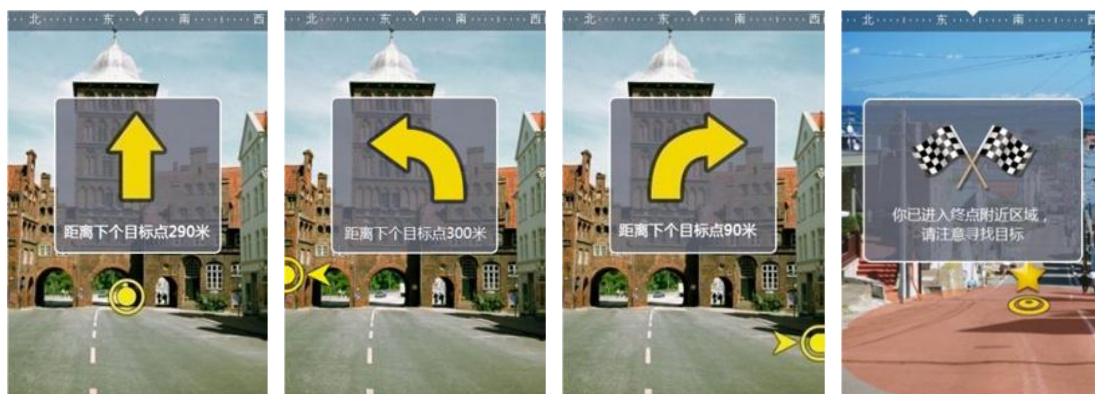


图 6-11 导航模式的用户界面设计图

## 6.3 软件设计与实现

### 6.3.1 系统总体架构

根据用户交互设计，在系统最上层设计一个用户交互模块，直接面向用户，负责用户界面布局、交互事件触发及运行逻辑，这一层称为用户层，作为系统的启动入口。为了系统架构的可读性和可扩展性，在用户层之下搭建一个接口封装模块，称为封装层。这个封装层将用户层所需要的功能封装起来，对其透明，封装层对用户层提供统一的接口供其调用。在封装层之下，便是被封装的功能层，分别为增强现实模块、兴趣点模块、导航路径模块、经纬度坐标偏转模块和搜索模块。增强现实模块主要提供基于传感器增强现实技术的方法实现，同时还提供一些其它常用的功能，如 **GPS 两地距离计算**、**获取摄像机方向和姿态**、**获取用户当前经纬度及海拔**、**获取屏幕高宽数据**等。兴趣点模块主要负责与兴趣点相关的功能实现，如兴趣点的数据结构定义、兴趣点的缓冲存储、远程服务器返回周围兴趣点搜索结果的解析等。导航路径模块主要负责与路径相关的功能实现，如导航路径的数据结构定义、导航路径的缓冲存储、远程服务器返回导航路径搜索结果的解析等。经纬度坐标偏转模块用于将真实经纬度坐标转换成偏转经纬度坐标。基于国家安全的考虑，在中国大陆地理位置服务应用中，经纬度坐标不能采用真实经纬度坐标，而是经过转换后的偏转经纬度坐标。经纬度坐标偏转模块用于将 **GPS** 接收器采集到的真实经纬度坐标转换成偏转经纬度坐标，并且这种转换是在远程服务器上进行的。偏转经纬度坐标之间的相对位置并没有发生改变，不影响相关的计算结果。搜索模块用于通过对远程服务器的访问进行周围兴趣点和导航

路径的搜索。

在功能层之下构建一个网络通信模块，称为网络层，用于对远程服务器进行网络访问操作，如经纬度坐标偏转模块需要通过网络通信模块向远程服务器发送经纬度坐标转换的请求，搜索模块需要通过网络通信模块向远程服务器发送周围兴趣点或导航路径的搜索请求。同时网络通信模块需要接收返回结果并提交给上层模块。

用户层以下的统称为系统底层。应用系统的模块架构如图 6-12 所示。

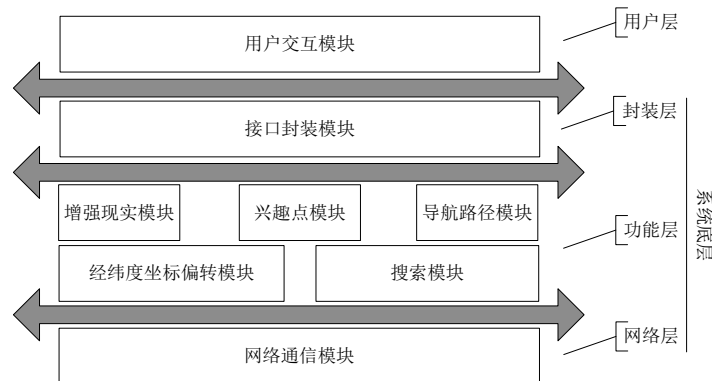


图 6-12 应用系统模块架构示意图

在软件实现上，将用户层中的用户交互模块定义成 Activity 系统组件类的继承类 CameraActivity，并将其作为系统启动类，在 AndroidManifest.xml 配置文件中声明。CameraActivity 除了负责用户界面布局、交互事件触发及运行逻辑外，还需负责系统初始化的操作。封装层用两个类来实现，分别为 ServiceProxy 和 MinimapService。ServiceProxy 继承了 Service 系统组件类，Service 组件相当于一个没有用户界面的 Activity 组件，可以在后台长时间运行，并且可以与 Activity 组件异步运行。由于有这个特性，封装层中的任何操作都不会造成用户层在运行时的滞后。ServiceProxy 对用户层提供统一的调用接口，其本身则调用实现封装层具体功能的 MinimapService。MinimapService 负责初始化功能层中增强现实模块、兴趣点模块和导航路径模块的实现类对象，调用各自的成员函数。同时，在 MinimapService 中直接实现了功能层中经纬度坐标偏转模块和搜索模块的功能。ServiceProxy 与 MinimapService 之间属于聚合关系。功能层中的增强现实模块、兴趣点模块和导航路径模块分别由类 ARUtil、AroundModule 和 RouteModule 实现。网络层中的网络通信模块由类 HttpClientUtil 实现，其中各个成员函数采用静态类型，可供 MinimapService、ARUtil、AroundModule 和 RouteModule 直接调用。应用系统的软件总体架构 UML 图如图 6-13 所示。

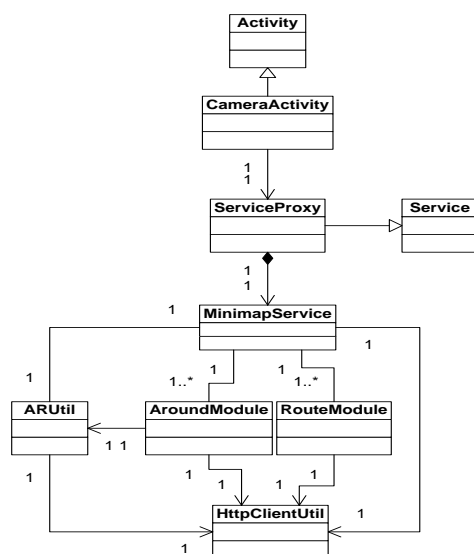


图 6-13 应用系统软件总体架构 UML 图

### 6.3.2 系统底层模块

系统底层包括封装层、功能层和网络层。在封装层中，ServiceProxy 的主要作用是将其下的模块与用户层之间进行封装隔离。ServiceProxy 本身没有任何具体功能的实现，而是通过调用 MinimapService 来完成任务。MinimapService 对象的初始化是在 ServiceProxy 的成员函数 onCreate()中完成的。MinimapService 实现了对功能层中增强现实模块、兴趣点模块和导航路径模块的实现类对象的初始化，封装各自的调用接口。在 MinimapService 中也直接实现了经纬度坐标偏转模块和搜索模块的功能。这两种功能都是在远程服务器上完成的，而对远程服务器进行网络访问操作是靠网络层实现的，因此 MinimapService 中的这两个模块会先根据远程服务器提供的经纬度坐标转换、周围兴趣点搜索和导航路径搜索的网络 API 规则，构建相应的 URI（统一资源定位符）和数据参数，再调用网络层的相关函数进行对远程服务器的访问请求，获取返回的请求结果，然后对该结果进行解析，构成可用的数据信息。对周围兴趣点搜索的实现流程如图 6-14 所示。

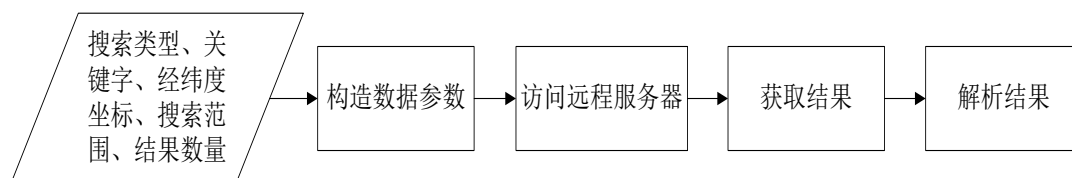


图 6-14 周围兴趣点搜索的实现流程



在功能层中, ARUtil 是增强现实模块, 实现了基于传感器增强现实技术的方法, 同时还实现了如 GPS 两地距离计算、获取摄像机方向和姿态、获取用户当前经纬度及海拔、获取屏幕高宽数据等功能, 供外部调用。ARUtil 还负责对 GPS 接收器、电子罗盘和重力加速度计进行初始化操作, 这些操作是在 ARUtil 的构造函数中完成的。初始化这些硬件设备的实现过程是通过创建这些硬件设备的监听器, 并对其进行系统注册而完成。例如对 GPS 接收器的初始化工作, 首先创建一个 GPS 接收器的监听器 gpsListener, 该监听器实现了 LocationListener 系统接口。同时设定经纬度坐标更新的间隔时间, 然后对这个监听器进行系统注册。此时 GPS 接收器初始化完成, 并可以开始工作了。

在 LocationListener 系统接口中, 有一个成员函数 onLocationChanged(), 系统在该成员函数中获取用户当前真实经纬度及海拔。由于需要采用偏转经纬度坐标, 因此, 在 onLocationChanged() 中, 当获取了真实经纬度坐标后, 还需要调用 MinimapService 中的经纬度坐标偏转模块将其转换成偏转经纬度坐标。在经纬度坐标偏转模块中分别实现了 coordGps2Coord20() 和 coord202CoordDeflect() 两个函数来负责将真实经纬度坐标转换成偏转经纬度坐标。

电子罗盘与重力加速度计的监听器创建和 GPS 接收器的监听器创建类似, 所实现的系统接口为 SensorEventListener, 在其成员函数 onSensorChanged() 中可以获取电子罗盘和重力加速度计的读数, 并实现了本文第三章阐述的方法计算出摄像机的方向和姿态。

ARUtil 的成员函数 getScreenXY() 用于获取某个已知经纬度及海拔的点在屏幕上的位置, 采用本文第四章阐述的方法, 其传入参数为经纬度及海拔。这样, 系统上层模块只要获取了经纬度及海拔, 就可以调用该成员函数获取其在屏幕上的投影坐标。

AroundModule 是兴趣点模块, 主要负责与兴趣点相关的功能实现, 如兴趣点的数据结构定义、兴趣点的缓冲存储、远程服务器返回周围兴趣点搜索结果的解析等。在 AroundModule 中定义了一个嵌套类 Around, 用于表示单个兴趣点的数据结构, 包括兴趣点名称、地址、电话、偏转经纬度坐标等信息。Around 中同时定义了一些成员函数供外部调用来获取兴趣点的这些相关数据。Around 的属性如图 6-15 所示。



Around
-dataLength
-poiId
-name
-address
-phoneNumber
-coordDeflectX
-coordDeflectY
-staticDistance
-cityCode

图 6-15 Around 的属性

根据用户交互设计，兴趣点在屏幕纵向上的摆动幅度根据兴趣点与用户距离的近中远来决定，离用户越远，幅度越大。因此，在 **Around** 中实现了一个成员函数 **getScreenY()**来调用 **ARUtil** 中的 **getScreenXY()**，获取在屏幕上的投影纵坐标，并对其结果根据用户交互设计进行处理，将结果提供给上层模块。其实现方法是比较兴趣点与用户距离的近中远范围，根据范围的不同，在原投影纵坐标的基础上加上或减去当前摄像机前后俯仰角度与摆幅系数的乘积，以此来控制其摆幅。其实现逻辑如下：

```
IF 距离在 0 到 200 米范围内 THEN
    投影纵坐标保持不变;
ELSEIF 距离在 200 到 1000 米范围内 THEN
    投影纵坐标 - 前后俯仰角度 × 3;
ELSE
    投影纵坐标 - 前后俯仰角度 × 5;
ENDIF
```

**RouteModule** 是导航路径模块，主要负责与路径相关的功能实现，如导航路径的数据结构定义、导航路径的缓冲存储、远程服务器返回导航路径搜索结果的解析等。在 **RouteModule** 中定义了一个嵌套类 **Route**，表示一条导航路径的数据结构，主要包括导航路径总长度、导航段个数、导航段数据列表等信息。**Route** 的属性如图 6-16 所示。

Route
-dataLength
-routeLength
-navigationSize
-navigations

图 6-16 Route 的属性

一条导航路径又是由若干条导航段组成，因此在 **Route** 中又定义了一个嵌套类 **Navigation** 来表示一条导航段的数据结构，主要包括导航段名称、导航段长度、构成导航段的经纬度坐标数据个数、偏转经纬度坐标数据列表等信息。**Navigation** 的属性如图 6-17 所示。

Navigation
-dataLength
-routeName
-navigationLength
-coordDeflectSize
-coordDeflects

图 6-17 Navigation 的属性

在 **Route** 和 **Navigation** 中都定义了一些成员函数供外部调用来获取导航路径和导航段的相关数据。

**HttpClientUtil** 是网络层中的网络通信模块，实现了对远程服务器的网络访问操作。数据包的提交方式分为 **get** 和 **post** 两种，**get** 方式是直接将数据参数直接添加在 **URI** 中进行提交，传送数据量小；**post** 方式是将数据参数按照变量和值相对应的方式存储，再一并提交至 **URI**，传送数据量大。在应用系统中，需要进行网络访问的是经纬度坐标转换、周围兴趣点搜索和导航路径搜索，其中远程服务器提供的经纬度坐标转换的网络 **API** 规则是通过 **get** 方式提交的，而周围兴趣点搜索和导航路径搜索的网络 **API** 规则是通过 **post** 方式提交的。因此，在 **HttpClientUtil** 中分别实现了这两种提交方式的成员函数，分别为 **getStringResultForHttpGet()** 和 **getStringResultForHttpPost()**。在这两个成员函数中，除了提交访问请求外，还需要接收请求结果。从远程服务器上返回的请求结果是数据流类型，需要将其解析成可用的字符串类型。

### 6.3.3 用户交互模块

应用系统分为兴趣点模式和导航模式，在 **CameraActivity** 中分别实现了这两种模式。**CameraActivity** 作为系统启动入口，在系统启动时需要负责系统初始化的操作，包括各种数据对象和硬件设备。

系统初始化工作完成后，需要检测卫星定位是否成功。所以在 **CameraActivity** 中实现一个 **SensorListener** 系统接口，其中的成员函数 **onSensorChanged()** 用于循环检测卫星定位是否成功，以及摄像机的方向和姿态。同时在 **CameraActivity** 中

定义一个布尔类型的公有成员变量 `isGpsAvailable`，初始值为假，表示正在定位状态中，为真时表示成功。`isGpsAvailable` 通过 `ServiceProxy` 对象调用 `ARUtil` 中返回当前卫星定位状态的成员函数来赋值，并在 `onSensorChanged()` 中循环检测。

由于兴趣点模式是默认模式，系统启动且卫星定位成功后会首先进入该模式，所以，`CameraActivity` 也会进行兴趣点模式的初始化，例如加载相应的用户界面布局 XML 文件、初始化各种用户界面的 `View` 系统类的继承类对象、设置各种监听器、创建对话框等。这些初始化操作都是在 `onCreate()` 中实现的。

在 `CameraActivity` 中加载的用户界面布局 XML 文件为 `camera.xml`，其中包括了兴趣点模式和导航模式的用户界面设置。由于系统最先进入的是兴趣点模式，所以，在 `camera.xml` 中默认将导航模式的用户界面设置成隐藏。在 Android 操作系统上，用户界面布局可由 XML 文件来静态定义，也可通过实现多个 `View` 系统类的继承类来动态定义。同时，用户界面布局 XML 文件中每个用户界面都有对应的 `id` 标识，`View` 系统类的继承类对象可以通过这个 `id` 标识进行绑定和初始化，通过这个对象动态地对相应的用户界面进行操作。针对用户界面的操作包括触摸、显示隐藏、图形绘制等。在兴趣点模式下的用户界面主要包括摄像机输出视频流、兴趣点气泡、方向标尺、兴趣点信息卡和按钮。这些用户界面在 `camera.xml` 文件中要定义，同时根据需要，定义了相应的 `View` 系统类的继承类对象与其绑定和初始化。这些用户界面中，最重要的是摄像机输出视频流和兴趣点气泡，其在 `camera.xml` 中对应的布局设置如下：

```
<com.android.arui.CameraView
    android:id="@+id/camera_panel"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" />
<com.android.arui.AroundView
    android:layout_gravity="top"
    android:layout_marginTop="81dip"
    android:id="@+id/around_panel"
    android:layout_width="fill_parent"
    android:layout_height="310dip" />
```

在该设置中使用了两个自定义类 `CameraView` 和 `AroundView`，都是 `View` 系统类的继承类，分别实现了摄像机输出视频流的显示和兴趣点气泡的渲染及相应操作。这两个类的对象在 `CameraActivity` 中进行初始化和调用。

CameraView 对摄像机输出视频流进行了初始化, 定义视频图像格式、视频方向、长宽等属性, 并启动摄像机, 在屏幕上显示输出视频流。CameraView 继承了 SurfaceView 系统类并实现其方法, SurfaceView 系统类是 View 系统类的继承类。

AroundView 实现了对兴趣点气泡的渲染及相应操作。该类是 ImageView 系统类的继承类, ImageView 系统类是 View 系统类的继承类。其中有一个非常重要成员函数 onDraw(), 该成员函数需要进行覆盖重新实现。onDraw() 主要负责具体的图形渲染, 当 AroundView 的对象在 CameraActivity 中初始化后, onDraw() 开始循环执行, 兴趣点气泡渲染在 onDraw() 中实现。在渲染兴趣点气泡之前, 需要进行周围兴趣点的搜索, 搜索操作在 onDraw() 中最先执行。周围兴趣点搜索是通过 ServiceProxy 调用 MinimapService 中相关的周围兴趣点搜索函数, 在后台进行对远程服务器的搜索请求访问, 并将返回的结果存储在类型为 Around 的 List 容器 arounds 中。由于 onDraw() 是循环执行的, 而当用户没有提交新的搜索请求时, 周围兴趣点搜索操作是不会循环执行的, 所以, 需要在 AroundView 中定义一个布尔类型的私有成员变量 isSearchAroundEnabled 来控制周围兴趣点的搜索操作。AroundView 中周围兴趣点搜索的实现逻辑如下:

```
isSearchAroundEnabled 初始值为真;
IF isSearchAroundEnabled 为真 THEN
    WHILE 搜索结果不为空
        执行搜索操作;
    ENDWHILE
    赋 isSearchAroundEnabled 为假;
ENDIF
返回搜索结果;
```

由于 List 容器 arounds 存储了所有的结果, 甚至有上千个兴趣点的信息, 而根据用户交互设计, 在屏幕上最多显示 20 个, 所以, 将 arounds 中的兴趣点离用户距离由近至远的顺序提取出前 20 个, 存储于另一个 Around 类型的缓冲器 displayBuffer 数组中。若搜索的结果未满 20 个兴趣点, 则按实际的兴趣点数量存储。displayBuffer 数组中的兴趣点就是需要渲染在屏幕上的兴趣点。

在屏幕上渲染兴趣点气泡时, 采用 Rect 系统类的对象来表示一个兴趣点在屏幕上的矩形渲染区域。计算渲染区域在屏幕上的位置时, 首先通过存储于 displayBuffer 数组的 Around 类型兴趣点信息获取对应兴趣点在屏幕上的投影坐标, 再以该坐标为基点初始化一个 Rect 系统类的对象, 这样就获取了一个兴趣点

的渲染区域。同时定义一个 **Drawable** 系统类的对象来表示兴趣点的气泡图标，由于兴趣点气泡图标会根据兴趣点离用户的距离以及是否选取而采用 **res** 资源目录中不同的位图文件，因此 **Drawable** 系统类的对象会根据这些条件进行初始化。最后调用该 **Drawable** 系统类对象的成员函数 **draw()**即可进行渲染。兴趣点气泡渲染的实现逻辑如下：

**WHILE** 遍历所有在 **displayBuffer** 数组中的兴趣点

判断该兴趣点是否被选取；

根据距离和选取状态配置该兴趣点的位图；

计算该兴趣点的渲染区域；

将以上渲染信息加入至一个容器中；

**ENDWHILE**

对容器中的所有数据进行渲染；

关于实现兴趣点气泡的选取操作，在 **ImageView** 系统类中有一个触摸事件监听成员函数 **onTouchEvent()**，可以通过该成员函数获取当前用户触摸的屏幕坐标，并将该屏幕坐标与兴趣点气泡渲染区域进行比较，若触摸的屏幕坐标落入该渲染区域内，则表示选取了该兴趣点气泡，反之亦然。

兴趣点模式中，相关的软件架构 UML 图如图 6-18 所示。

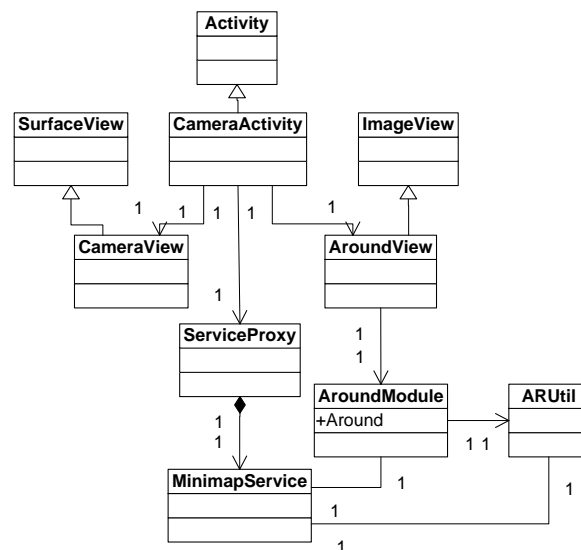


图 6-18 兴趣点模式软件架构 UML 图

兴趣点模式向导航模式切换时，根据用户交互设计，通过三种方式进入。在 **CameraActivity** 初始化阶段中，已经对这三种方式设置了监听器，事件触发后统

一调用成员函数 `enterNavigation()` 进入导航模式。在导航模式下的用户界面主要包括摄像机输出视频流、导航标注点、方向标尺和导航信息卡，用户界面布局 XML 文件为 `navigation.xml`。同时定义了 View 系统类的继承类 `NavigationView` 来实现导航模式的渲染和运行逻辑，其对象是在 `CameraActivity` 中初始化。`navigation.xml` 中关于 `NavigationView` 的布局设置如下：

```
<com.android.arui.NavigationView
    android:id="@+id/navigation"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" />
```

进入导航模式时，兴趣点模式中相关的用户界面会全部隐藏，显示导航模式中相关的用户界面，同时调用 `NavigationView` 对象的成员函数 `beginNavigate()` 进行导航路径搜索。导航路径搜索与周围兴趣点搜索一样，通过 `ServiceProxy` 调用 `MinimapService` 中相关的导航路径搜索成员函数，在后台进行对远程服务器的搜索请求访问，返回的导航路径由 `Route` 表示。在导航过程中，系统仅与当前的导航段进行交互，因此并没有在 `NavigationView` 中定义一个 `Route` 类型的成员变量用于存放导航路径的搜索结果，而是在 `MinimapService` 中封装两个接口 `setCurrentNavigation()` 和 `getCurrentNavigation()`，分别用于设置和获取当前的导航段，并通过 `ServiceProxy` 统一对外暴露这两个接口。同时在 `NavigationView` 中定义一个整型成员变量 `navIndex` 来记录当前导航段的索引，该索引的最大值为导航路径中导航段的总数。`NavigationView` 中导航路径搜索的实线逻辑如下：

获取目的地的经纬度坐标；

获取用户当前位置到目的地的所有路径数据，并存放至一个容器中；

IF 容器为空 THEN

    返回；

ENDIF

取出容器中的第一条路径数据，并返回结果；

`NavigationView` 中也有一个成员函数 `onDraw()` 需要覆盖重新实现，在该成员函数中主要实现导航标注点的渲染、导航信息卡的提示及相关运行逻辑，也是导航模式的关键实现部分。

一条导航段是由若干个经纬度坐标数据构成的，叠加在摄像机输出视频流上的导航标注点所对应的是导航段中最后一个经纬度坐标数据，即该导航段的目标点。因此，在每次渲染时，首先通过 `ServiceProxy` 的 `getCurrentNavigation()` 获取

当前的导航段,并通过 Navigation 中的成员函数 `getCoordDeflects()` 获取存放在 List 容器中构成当前导航段的所有偏转经纬度坐标,由 List 容器来获取最后一个经纬度坐标数据。之后利用 ServiceProxy 封装的接口 `getScreenXY()` 来获取导航标注点在屏幕上的投影坐标。导航标注点的渲染与兴趣点气泡相同,通过 Rect 系统类的对象来表示导航标注点在屏幕上的渲染区域,不再赘述。

在 `onDraw()` 中需要实时检测用户与导航段目标点的距离,当用户进入导航段目标点 50 米半径区域时,会震动手持设备提醒用户,并自动进入下一条导航段。用户与导航段目标点的距离数据由 ARUtil 的成员函数 `getDistance()` 负责计算,并由 MinimapService 和 ServiceProxy 封装提供对外接口。当该距离小于 50 时,调用 Android 操作系统提供的 Vibrator 系统类中的成员函数 `vibrate()` 来触发手持设备的震动,并使成员变量 `navIndex` 自加一,将其带入 ServiceProxy 的 `setCurrentNavigation()` 中来设置下一条导航段。若 `navIndex` 已经等于导航路径中导航段的总数时,表示当前的导航段为最后一条,即将到达导航路径的终点了。导航标注点渲染的实现逻辑如下:

获取当前导航段目标点的经纬度坐标数据;

计算其屏幕投影坐标;

IF 用户离导航段目标点距离小于 50 THEN

`navIndex` 自加 1;

    将当前导航段设置成下一个导航段;

    计算渲染区域;

    渲染导航标注点;

ELSE

    计算渲染区域;

    渲染导航标注点;

ENDIF

根据用户交互设计,导航信息卡需要实时地显示当前导航标注点的方向。因此,在 `onDraw()` 中,还需要实时地对导航标注点在屏幕上的位置进行检测,并在导航信息卡上动态加载 res 资源目录中相应的位图文件进行渲染。

导航模式中,相关的软件架构 UML 图如图 6-19 所示。

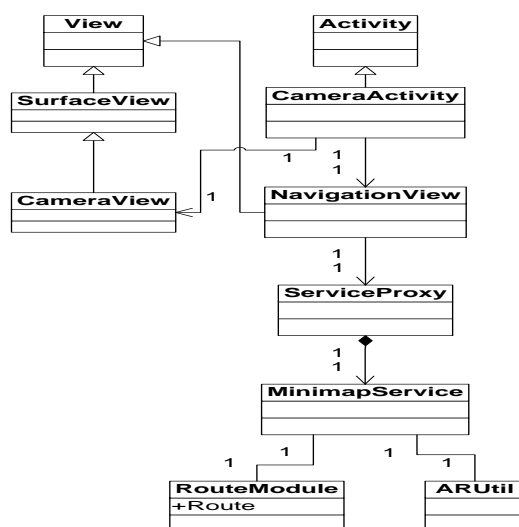


图 6-19 导航模式软件架构 UML 图

图 6-20 展示了应用系统中兴趣点模式和导航模式的实现效果。可以看到，整个实现效果完全满足了用户交互设计，利用基于传感器的增强现实技术，使虚拟物体能够很好地与目标物体对准。在整个测试过程中，系统运行稳定，交互结果准确，用户体验自然。

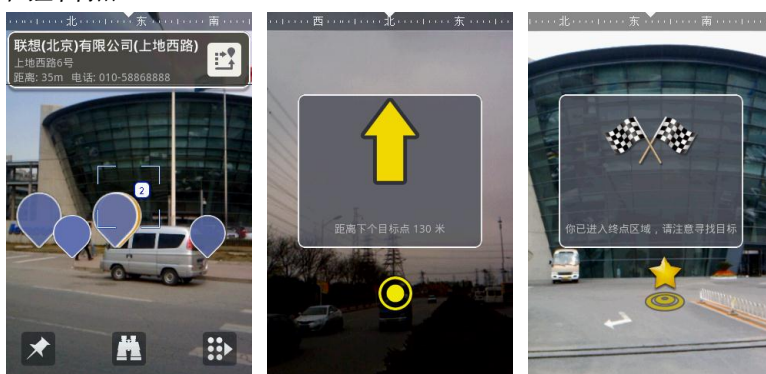


图 6-20 左图为兴趣点模式的实现效果，中图和右图为导航模式的实现效果



## 第七章 总结与展望

本文首先对移动增强现实技术的应用背景进行了全面的介绍，同时阐述了基于机器视觉的增强现实技术实现原理，再提出了两个具有高度移动性的基于传感器的增强现实技术，主要应用于地理位置服务，一种是利用传感器计算任意目标物体在摄像机图像平面的投影坐标来定位虚拟物体的渲染位置，另一种是利用传感器将三维虚拟路径叠加在摄像机图像平面中的真实道路上。最后利用基于传感器的增强现实技术，设计并实现了一种地理位置服务应用系统。

基于机器视觉的增强现实技术需要通过对复杂的图像数据进行计算来获取虚拟物体的渲染位置，运行开销大，并且容易受到光照、距离等因素影响其效果，受限于周围环境，移动性差，不适合大多数手持设备。本文提出的两种基于传感器的增强现实技术利用卫星定位来获取用户与目标物体之间的相对位置，利用电子罗盘和重力加速度计来获取摄像机的方向和姿态，并通过摄像机投影原理，计算出目标物体在图像平面上的位置，渲染虚拟物体。该解决方案除了具备虚实结合、实时交互和三维注册的特点外，还因不受地理空间、自然环境的限制，具备高度的移动性。其运行开销低、速度快，十分适合在手持设备上应用。

目前，移动互联网已经逐步形成一个新的趋势，蕴藏着巨大的创新点和商机。网络运营商、移动终端设备商和软件开发商都已经加入了这个新的市场中，从终端设备到软件产品，都在极力挖掘移动互联网的潜力。随着移动互联网的发展，地理位置服务应用模式也在快速扩展，从早期的地图导航定位到现在的社交网络交互，利用地理位置服务的需求越来越大，人们更希望能够通过直观的位置来将虚拟世界具体化。本文提出的两种基于传感器的增强现实技术是顺应了移动互联网的发展趋势，将增强现实技术应用到地理位置服务的领域中，是对这类应用模式的一种探索。所设计并实现的应用系统将这种探索具体化，其在用户体验方式上的创新，是具有研究价值的。

通过移动增强现实技术来创造更多的实际应用，特别是要保证其移动性，还需要从技术实现和应用模式两方面进行深入研究。在技术实现上，基于传感器的增强现实技术也可能会因为周围环境而干扰这些硬件设备，例如周围高大障碍物较多时会使得卫星定位误差较大，磁场较大的环境也会干扰电子罗盘的数据。另外，在对体积小或距离短的目标物体进行定位时，会因为硬件数据的精度问题，

使得定位不准确。这些不足正好与基于机器视觉的增强现实技术形成了互补关系，因此，可将两种解决方案结合起来，例如对体积大或距离远的目标物体可以采用传感器，而对体积小或距离近的目标物体，在传感器定位的基础上利用机器视觉对其进行补偿。这一类结合方式已经在一些国内外公开发表的论文中提及，从目前的趋势来看，移动增强现实技术将会越来越多地采用传感器和机器视觉相结合的方式。这种结合方式不仅能够保证移动性，而且其虚实叠加的准确度比起单一采用一种解决方案更准确。

但是在这种结合方式中，由于加入了机器视觉，在进行图像识别时，需要有模板图像进行匹配，在目标物体的种类很多的情况下，与之对应的模板图像也会增多，这些模板图像的采集和存储将会是一个新的问题。当然这也和具体的应用模式相关，在某些应用需求下，仅需通过对目标物体的轮廓拓扑结构就可以进行判别，在这种情况下甚至不需要模板图像。另外，图像数据的计算对于大多数手持设备来说，开销较大，因此，可以建立一个远程服务器，将手持设备上获取的图像数据通过网络传输到这个远程服务器上计算，并把结果返回至手持设备上。但这种方案对网络通信质量要求较高，可能因为网络通信质量差而无法保证实时性。

对用户当前位置进行定位，除了卫星定位外，也可以利用射频识别（RFID）或蓝牙定位技术。但这两种方案只适合局部区域，如社区、室内等。RFID 是一种非接触式的自动识别技术，它利用射频信号自动识别目标物体并获取相关数据信息，可用于各种极端环境。利用 RFID 进行位置定位，需要预先在一定的区域内安装多个 RFID 接收器，手持设备上需要配置一个 RFID 发射器，通过信号的发送与接收，来确定用户当前的位置。蓝牙是一种支持设备短距离通信的无线电技术，一般在 10 米内。与 RFID 类似，也需要预先搭建环境，在一定的区域内安装多个蓝牙发射器，手持设备上配置一个蓝牙接收器，当用户接近某一个蓝牙发射器时，手持设备上的蓝牙接收器会接收到信号，由此确定用户的当前位置。

在应用模式上，移动增强现实技术主要可以在信息检索、游戏等应用类型中进行扩展。信息检索的概念比较广，导航、社交、搜索等都可以属于信息检索。利用移动增强现实技术，用户可以通过摄像机对准感兴趣的目标物体来获取相关的信息，同时通过输出视频流的图像来查阅信息，这种信息可能是三维虚拟物体，也可能是简单的文字，但都会叠加在目标物体上，用户可以与这些信息进行交互，这种人机交互方式本身就是一种创新。将这种交互方式逐步带入到导航、社交、搜索等领域中，必将创造出一种新的用户体验模式。在游戏方面，引入移动增强

现实技术，用户身处的是真实的环境，而非虚拟环境，这是与传统游戏最大的不同。用户与叠加在输出视频流上的虚拟物体进行交互，这种交互可能完全是一种全身参与，例如需要在真实的环境中移动、手持设备需要作出不同的姿态等。游戏也可以支持多人参与，形成类似于传统网络游戏模式，例如采用上述的 **RFID** 定位技术，可以在一个区域内搭建游戏环境，每个参与者利用配有 **RFID** 发射器的手持设备，相互之间进行协同或对抗互动。

移动增强现实技术无论在技术实现的改进，还是应用模式的创新，一直在持续当中。伴随移动互联网的发展，移动增强现实技术的潜力必将被越来越多的人认识到，各大企业和研究机构已经投入了大量的资金和人力在这一领域展开研发。也许在不久的将来，移动增强现实技术会普及到我们的日常生活中，成为重要的人机交互方式。

## 致 谢

在论文即将完成之际，我首先要感谢我的导师陈雷霆老师，感谢陈老师在我攻读硕士期间在学术上给予我的指导，以及在为人处事上给予我的教导。陈老师无论在对治学的态度和对学生的关心上，始终尽职尽责，为我在今后的工作与学习中树立了榜样。一日为师，终生为父，在此，谨向培养和帮助我的陈老师致以由衷的敬意。

感谢蔡洪斌老师、曹跃老师、何晓曦老师、邱航老师和电子科技大学数字媒体技术研究所的其他老师。我在研究所工作与学习期间，他们给予了极大的帮助和支持，感谢他们为我所做的一切。

感谢联想研究院（北京）的马文超博士对本课题的指导，以及梁小霞和郜远对项目的支持和帮助。

同时，非常感谢我的父母。感谢他们在我求学生涯中，给予我在物质与精神上的全力支持，在今后的工作与学习中，我将以更优异的成绩来报答他们。

三年的硕士研究生生活即将结束，新的开始即将到来。这三年的经验对我来说无比宝贵，再一次衷心感谢在我硕士研究生学习期间所有帮助和支持我的人，祝他们一生平安、幸福美满。

## 参考文献

- [1] Ivan Sutherland. A head-mounted three dimensional display. In Proc. FJCC 1968. Washington D.C., USA: Thompson Books, 1968, 757–764
- [2] Thomas P. Caudell, David W. Mizell. Augmented Reality: An Application of Heads-Up Display Technology to Manual Manufacturing Processes. In Proc. Hawaii International Conference on System Sciences 1992. Los Alamitos, CA, USA: IEEE Computer Society Press, 1992, 659-669
- [3] Thomas P. Caudell. Introduction to Augmented Reality. In Proc. SPIE, 1994, 2351: 272-281
- [4] Adam L. Janin, David W. Mizell, Thomas P. Caudell. Calibration of Head-Mounted Displays for Augmented Reality Applications. In Proc. IEEE VRAIS 1993, Seattle, WA, USA, 18-22 September 1993, 246-255
- [5] Hirokazu Kato and Mark Billinghurst. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In Proc. IWAR 1999, San Francisco, CA, USA, 20-21 October 1999, 85–94
- [6] Hirokazu Kato, Mark Billinghurst, Suzanne Weghorst, et al. A mixed reality 3D conferencing application. Human Interface Technology Laboratory, University of Washington, Technical Report R-99-1, 1999
- [7] R. Azuma. A survey of augmented reality. Teleoperators and Virtual Environments, 1997, 6(4): 355-385
- [8] S. Feiner, B. MacIntyre, T. Hollerer, et al. A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. In Proc. ISWC 1997, Cambridge, MA, USA, 13-14 October 1997, 74–81
- [9] B. Thomas, B. Close, J. Donoghue, et al. ARQuake: An outdoor/indoor first person augmented reality application. In Proc. ISWC 2000, Atlanta, GA, USA, 18-21 October 2000, 139–146
- [10] B. Thomas, B. Close, J. Donoghue, et al. First Person Indoor\Outdoor Augmented Reality Application: ARQuake. Personal and Ubiquitous Computing, 2002, 6(1): 75–86
- [11] W. Piekarski, B.H. Thomas. ARQuake – Modifications and hardware for outdoor augmented reality gaming (LINUX03). In Proc. the 4th Australian Linux Conference, Perth, Australia, 22–25 January 2003

- [12] W. Piekarski, R. Smith, B. Thomas. Designing backpacks for high fidelity mobile outdoor augmented reality. In Procs. ISMAR 2004, Arlington, VA, USA, 2–5 November 2004, 280-281
- [13] A. Cheok, S. Fong, K. Goh<sup>1</sup>, et al. Human Pacman: A Mobile Entertainment System with Ubiquitous Computing and Tangible Interaction over a Wide Outdoor Area. In Proc. Mobile HCI 2003, Lecture Notes in Computer Science, 2003, 2795: 209-223
- [14] A. D. CHEOK, K. H. GOH, W. LIU, et al. Human Pacman: A mobile, wide-area entertainment system based on physical, social, and ubiquitous computing. *Personal Ubiquitous Computing*, 2004, 8(2): 71-81.
- [15] Markus Kähäri, David J. Murphy. MARA-sensor based augmented reality system for mobile imaging. In Proc. ISMAR 2006. New York, NY, USA: ACM Press, 2006
- [16] A. Olwal. LightSense: Enabling Spatially Aware Handheld Interaction Devices. In Procs. ISMAR 2006, Santa Barbara, CA, USA, 22-25 October 2006, 119-122
- [17] Alex Olwal, Anders Henrysson. LUMAR: A Hybrid Spatial Display System for 2D and 3D Handheld Augmented Reality. *ICAT*, 2007, 0: 63-70
- [18] O. Rashid, W. Bamford, P. Coulton, et al. PAC-LAN: Mixed-Reality Gaming with RFID-Enabled Mobile Phones. *ACM Computers in Entertainment*, 2006, 4(4)
- [19] D. Haehnel, W. Burgard, D. Fox, et al. Mapping and localization with RFID technology. In Proc. IEEE International Conference on Robotics and Automation, 2004, 1: 1015-1020
- [20] N. Proctor. Off base or On Target? Pros and Cons of wireless and location aware applications in the museums. In Procs. ICHIM 2005, Paris, France, 21-23 September 2005, 6-28
- [21] P. Steggles, S. Gschwind. The Ubisense Smart Space Platform. In Procs. Pervasive 2005, 2005, 191
- [22] 徐彤, 王涌天. 用于虚拟现实的六自由度电磁跟踪系统. *北京理工大学学报*, 2000, 20(5): 544-549
- [23] 金剑华, 陈一民. 增强现实中基于视觉与磁力跟踪器的三维注册方法. *计算机应用*, 2006, 26(6): 1485-1489
- [24] Mike Bajura, Henry Fuchs, Ryutarou Ohbuchi. Merging Virtual Reality with the Real World: Seeing Ultrasound Imagery within the Patient. In Procs. SIGGRAPH 1992, Computer Graphics, 1992, 26(2): 203-210
- [25] Andrei State, David T. Chen, Chris Tector, et al. Case Study: Observing a Volume Rendered Fetus within a Pregnant Patient. In Procs. IEEE Visualization 1994, Washington D.C., USA,

- 17-21 October 1994, 364-368
- [26] Andrei State, Gentaro Hirota, David T. Chen, et al. Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking. In Procs. SIGGRAPH 1996, New Orleans, LA, USA, 4-9 August 1996, 429-438
- [27] Andrei State, Mark A. Livingston, Gentaro Hirota, et al. Techniques for Augmented-Reality Systems: Realizing Ultrasound-Guided Needle Biopsies. In Procs. SIGGRAPH 1996, New Orleans, LA, USA, 4-9 August 1996, 439-446
- [28] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004, 60(2): 91-110
- [29] M. Ozuysal, P. Fua, V. Lepetit. Fast keypoint recognition in ten lines of code. In *Proc. CVPR 2007*. Washington D.C., USA: IEEE Press, 2007, 1-8
- [30] H. Bay, T. Tuytelaars, L. Van Gool. SURF: Speeded up Robust Features. In *Proc. ECCV 2006*, Graz, Austria, 7-13 May 2006, 404-417
- [31] H. Bay, B. Fasel, L. V. Gool. Interactive Museum Guide: Fast and Robust Recognition of Museum Objects. In *Procs. IMV 2006*, Graz, Austria, 13 May 2006
- [32] G. Takacs, V. Chandrasekhar, B. Girod, et al. Feature Tracking for Mobile Augmented Reality Using Video Coder Motion Vectors. In *Procs. ISMAR 2007*, Nara, Japan, 13-16 November 2007, 141-144
- [33] G. Fritz, C. Seifert, L. Paletta. Urban Object Recognition from Informative Local Features. In *Procs. ICRA 2005*, Barcelona, Spain, 18-22 April 2005, 132-138
- [34] G. Fritz. A Mobile Vision System for Urban Detection with Informative Local Descriptors. In *Procs. ICVS 2006*, New York, NY, USA, 5-7 January 2006, 30
- [35] A.J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Procs. ICCV 2003*, Nice, France, 14-17 October 2003, 1403-1410
- [36] G. Klein, D. Murray. Parallel tracking and mapping for small AR workspaces. In *Procs. ISMAR 2007*, Nara, Japan, 13-16 November 2007, 1-10
- [37] G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *Procs. ISMAR 2009*, Orlando, FL, USA, 19-22 October 2009, 83-86
- [38] A. Peternier, F. Vexo, D. Thalmann. Wearable Mixed Reality System in Less Than 1 Pound. In *Procs. EGVE 2006*, Lisbon, Portugal , 8-10 May 2006
- [39] A. L. Liu, H. Hile, H. Kautz, et al. Indoor Wayfinding: Developing a Functional Interface for Individuals with Cognitive Impairments. In *Procs. ASSETS06*. New York, NY, USA: ACM

Press, 2006, 95-102

- [40] J. Newman, G. Schall, I. Barakonyi, et al. Wide-Area Tracking Tools for Augmented Reality. In Procs. Pervasive 2006, Dublin, Ireland, 7-10 May 2006, 143-146
- [41] S. DiVerdi, T. Höllerer. GroundCam: A Tracking Modality for Mobile Mixed Reality. In Procs. IEEE Virtual Reality 2007, Charlotte, NC, USA, 10-14 March 2007, 75-82
- [42] G. Klein, D. Murray. Compositing for Small Cameras. In Procs. ISMAR 2008, Cambridge, UK, 15-18 September 2008, 57-60
- [43] Z. Y. Zhang. A flexible new technique for camera calibration. IEEE Trans. Pattern Analysis and Machine Intelligence, 2000, 22(11): 1330-1334
- [44] Jim Blinn. Me and My (Fake) Shadow. IEEE Computer Graphics and Applications, 1988, 8(1): 82-86
- [45] Lance Williams. Casting Curved Shadows on Curved Surfaces. In Procs. SIGGRAPH 1978, 1978, 12(3): 270-274
- [46] Franklin C Crow. Shadow Algorithms for Computer Graphics. In Procs. SIGGRAPH 1977, 1977, 11(2): 242-248



## 攻硕期间取得的研究成果

- [1] 周一舟. 增强现实应用中的三维注册方法. 中国, 发明专利, 2010.4, 已受理
- [2] 周一舟. 一种基于真实景象的导航方法. 中国, 发明专利, 2010.4, 已受理