華中科技大學

本科生毕业设计[论文]

基于飞行器的室内 SLAM 系统

院　　系　　光学与电子信息学院

专业班级　　集成 1202 班

姓　　名　　李小龙

学　　号　　U201214058

指导教师　　雷鑑铭

2016 年 6 月 6 日

学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包括任何其他个人或集体已经发表或撰写的成果作品。本人完全意识到本声明的法律后果由本人承担。

作者签名：　　年　月　日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保障、使用学位论文的规定，同意学校保留并向有关学位论文管理部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权省级优秀学士论文评选机构将本学位论文的全部或部分内容编入有关数据进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于 1、保密口，在　　年解密后适用本授权书

2、不保密口　。

（请在以上相应方框内打“√”）

作者签名：　　年　月　日

导师签名：　　年　月　日

摘 要

旋翼类飞行器是近几年以来，在研究领域或者商用的新宠，因为它有着出色的悬停性能，以及入手快、易操控性、可防护、易携带等特点。在自主侦查、探测方面，飞行器也有着极大的应用前景，一些基于飞行器的革命式的交叉创新也开始显露身手。旋翼类飞行器小巧的外形和出色的运动性能，所以比较适合在室内以及其他的相对而言比较狭窄、地形比较复杂的环境中运动。只是，在 SLAM 定位与实时地图重建今天仍处于算法优化、传感器升级的情况下，飞行器的室内导航技术还只是在少数的实验室、顶级的公司中完全实现，技术还处于初创期。但是很多优秀的算法开始逐步开源，如 ORB-SLAM，RGBD-SLAM，飞行器的技术愈加可靠起来，深度相机如 Kinect 的价格变得更加便宜，探索并搭建这样的一套基于飞行器的室内导航系统，意义非常重大，一方面对于飞行器的室内导航技术发展至关重要，另一方面可以实现特殊室内环境信息的无人化探索。本论文描述了搭建SLAM系统的过程,以及系统的软硬件组成,完成了来自顶尖实验室的SLAM算法的探索实践与简单优化，并完成了较大规模的技术整合，初步实现了基于飞行器的室内环境实时 3 维重建。对于之后飞行器向空中机器人的演变，有着重要的意义。

关键词：旋翼飞行器；SLAM；深度相机；匹配效率

Abstract

Multi-rotor copter is the new tendency these yearsBecause of its excellent hover performance, also its good features including being fast to learn the control skills, easy to use, safety and being convenient to be carried.In those situation like autonomous surveillance, unmanned exploration, the aerial vehicles have such great application potential, some revolutionary cross innovation which are based on aerial vehicles have been already appearing in this world. With its small shape and excellent motion ability, the multi-rotor aerial vehicle is fit for those environment which is narrower or more complex when compared to others. However, in this time, since SLAM technology is now under research and optimization, including those hardware, the indoor navigation for aerial robots has just been realized in those advanced Labs or companies, the technology is at its start-up. Meanwhile, we should also notice that so many excellent algorithms are joining the open-source tendency, like the ORB-SLAM, RGBD-SLAM, and those flight control unit is more and more reliable, depth camera like Kinect becomes cheaper, so it is both affordable and meaningful to build such system by ourselves. On the one hand, it's important to help those technologies about autonomous indoor navigation for aerial robots, on the other hand, it will realize the unmanned exploration for special environment with aerial vehicles. This paper describes the whole process to build such a SLAM system, also the composition of software and hardware. This paper finishes the practice and simple optimization of SLAM algorithms from those cutting-edge Labs, while also achieves integration of different technologies in one system. This project finishes the first stage of real-time 3D reconstruction of indoor environment, which matters much for the transition of aerial vehicles to unmanned flying robots.

Key Words： Multi-rotor copter ;SLAM; depth-camera; match efficiency

目　录

1 绪论

1.1 研究项目综述

计算机视觉领域，有两大分之，其中一个重要分支，3D reconstruction 因着 VR 的技术风潮，开始受到越来越多的重视。而对于机器人来说，则与研究已经超过 30 年的 SLAM（自主导航与地图重建）技术，近几年来，随着小型飞行器控制技术的飞速发展，以及 SLAM 算法效率的提升，在世界各大学的视觉研究所内已经实现了技术突破，技术积累已到达一个爆发期，TED 上面也多有介绍，又或者是谷歌的 3D 地图街景车。在国内则体现为价格可为一般消费者可以接受的室内扫地机器人。然而，技术不应该仅仅停留在此，它应该有更广阔的运用空间。我们注意到有这样一些情景，是不太适合人直接进去进行探索的，比如浓烟区，大峡谷，犯罪分子藏身的建筑物，这个时候利用飞行机器人灵活移动的特点，配合一定的视觉导航与避障，就可以很好地通过飞行器挂载的摄像头和传感器来进行探索。

研究目标为利用飞行器传输环境信息完成环境的数学重建，具体的研究内容为搭建空中视角的图像传输与实时重构的硬件系统，进行图像处理，验证并实现立体视觉算法。所采用的视觉平台为 Kinect V2.0、四旋翼飞行器、图像处理地面工作站，后期将把算法移植到普通的双目摄像头上。在一套完整的基于无人机图传系统，通信模块，以及地面站控制系统基础上加上航拍关键帧提取，对于获取到的图像数据，经过图像滤波、特征匹配、点云拼接与融合、纹理优化之后，做到一定程度的实时生成室内环境的 RGBD 空间中的 3D 图像信息。

无人机的具有灵活的空中视角，可以自由地完成地表平坦度、障碍物形状位置等信息的获取，探测手段具有扩展性； 利用直线提取取代传统的单点提取，并融合深度信息，更易于获取地表地形的空间特征； 获取到的三维场景经过透明化、特征筛选，可以简化模型，获取最核心的地形信息，便于在指导机器人运动上的控制，甚至是集群运动。

1.2 课题动机与目的

飞行器以其卓越的动力性能，得到了越来越广泛的重视和应用。除开军用的大

型直升机、民用航空的喷气式飞机，种类繁多的小型飞行器，有着相当广泛的应用场景。固定翼的飞行器通过弹射起飞，可以达到一定的飞行距离和时间，完成特定地区的侦查和测绘；小型直升机则可用于农药的喷洒，以及短距离物资运送；旋翼类飞行器则是近几年以来来，在研究领域、商用等比较热门的新宠，因为它有着出色的悬停性能，以及入手快、易操控性、可防护、易携带等特点，为人们所追捧。环观旋翼类飞行器的创新运用，目前已经开始逐渐应用的还只是无人机航拍、植保机喷洒农药，以及携带有高精密传感器的飞行器在电力巡线、桥梁检测方面的运用，但是飞行器应当具有更广泛、深入的运用，出现一些革命性的交叉式创新，特别是在自主侦查、探测方面。因为旋翼类飞行器小巧的外形和出色的运动性能，所以比较适合在室内以及其他的相对而言比较下载、地形比较复杂的环境中运动。只是，在 SLAM（simultaneous localization and mapping 地图即时构建与定位）今天仍处于算法优化、传感器升级的情况下，飞行器的室内导航技术还只是在少数的实验室、顶级的公司中完全实现，探索并搭建这样的一套室内导航系统，对于飞行器室内飞行意义非常大。

而另外一方面，随着图像处理技术的不断发展，人们越来越倾向于发掘环境的 3 维信息，基于双目、单目以及 RGBD 相机的 3 维重建技术，正在成为研究界的新宠。搭载在飞行器上的视觉测量平台，不仅可以用于机器人自身的定位与导航，同时也可以借助于飞行器灵活的运动特性，来帮助人们更好地探索环境，在那些不太适合人直接进入的场合，人们可以借助于这样一双会飞的“眼睛”看到并将环境信息转化为点云实现实时建模，人们可以用全局视角来观察环境，从而更好地进行复杂环境下的决策。

比较精密的光电传感器大都比较昂贵，激光雷达、带有深度信息的摄像头价格动辄上万，但是微软在 2010 年度推出的 Kinect for Windows1.0 极大地改变了这样的状况，人们可以通过相对低廉的花费，来获取到图像的深度信息，特别是之后推出的 Kinect for Windows2.0，图像定位与识别的算法进一步提高。Kinect 传感器应用于三维重建，比如可以用 Kinect 绕着物体转动一圈，即可以生成物体的高精度 3 维模型，生成的点云模型，可用于 3D 打印。所以 Kinect 的出现提供了我们一套性价比较高的 3 维图像处理系统。值得一提的是，Kinect 属于 rgb-d 相机的一种，也就是带有深度信息的摄像头，而经过开发者的不断努力，适用于 rgb-d 相机的 Linux 下的驱动以及库软件也不断地更新和完善，这样一款图像设备得以和强大的

linux 开发社区对接，从而产生了很多实用而创新的设计。

1.3 国内外研究现状

针对移动机器人的 SLAM 系统早已有之，30 年前美国等国家便已开展相关的研究。近几年的室内 SLAM 领域的研究，主要是在 SLAM 算法上的效率与精确度的提升，以及传感器硬件的小型化在特殊机器人上的运用，比如多轴飞行器。

宾州州立大学学者，演示了基于关键帧的地图方法，要比滤波器方法在相同的运算代价上更精确。基于关键帧技术最具代表性的 SLAM 系统，可能是 PTAM。因为它第一次实现了将相机追踪和地图构建分开，进行并行计算，它在小型场合，如增强现实领域应用较好。PTAM 中的地图云点，会通过图像区块与 FAST 角点匹配。云点适合追踪但不适合位置识别。而实际上我们发现，PTAM 并不适合检测大的闭合回路，重定位基于低分辨率的关键帧小图像块，对视图不变性较差。

俄勒冈大学 ECE 学院的教授展示了大场景的单目 SLAM 系统，系统的前端使用 GPU 光流，同时用 FAST 特征匹配和运动 BA；后端用滑动窗口 BA。闭环检测。则是借助于 7 自由度约束的相似变换位姿图优化，能够校正尺度偏移。

德国慕尼黑大学基于四轴飞行器，在室内实现了基于 LSD-SLAM 的空中机器人系统，系统运行环境为 GPU，重建的实验室内环境可以通过无线实时地传输并显示在电脑上；

宾夕法尼亚大学工程与应用技术学院的机器人学教授 Vijay Kumar，在 2015 年 10 月的 TED 演讲上，展示了他们实验所开发的惊人的有关飞行器视觉的技术，其中就包括飞行器挂载摄像头等传感器进行室内环境重建，效果非常理想，同时也实现了飞行器搭载手机，通过手机的摄像头扫描物体，完成三维重建。

1.4 研究方法与思路

由于“基于飞行器的室内 SLAM 系统”，包含了比较多的软硬件支撑，比如稳定可靠的飞行器，Kinect 摄像头，强大的图像处理平台，以及众多的开源图像处理库，因而需要对硬件的使用具有比较丰富的经验，linux 基本使用与 linux 环境下进行 C++编程的能力要求较高，对开源硬件、开源库的上手效率要求比较高，另

外也涉及到图像处理变换中的各种数学模型和算法，对数学能力要求较高。因此，在准备这个项目的过程中，需要项目设计者能够及时补充并更新知识，总体的研究方法如下：

1、通过查阅国内外论文，了解国内外顶尖实验室的研究内容；

2、根据已有的知识来制定项目的初步硬件方案；

3、通过互联网资源来接触 ROS、机器视觉、SLAM 领域的牛人，认真地听取他们对于项目方案的评估；

4、积极寻找开源项目的支持，研究代码框架和编程要点；

5、进一步熟悉 linux、ROS 的使用，进行基本的 demo 编写；

6、实地地调试，消除 bug 并不断地优化代码；

研究思路如下：

1、多方调研，确定摄像头、开发平台的选型方案；

2、根据确定的方案，搭建 ubuntu 与 ROS 的环境；

3、利用开源的 demo 来测试系统库环境是否已安装好；

4、首先实现有线连接下，基于 Kinect 的室内环境 3D 重建；

5、调试 wifi 的视频传输方案，做到 kinect 数据的实时无线传输；

6、系统联合调试，对比并优化 SLAM 算法的效率。

2 SLAM 技术的发展

SLAM 技术即为实时的定位与地图生成，是设计智能移动机器人必须要克服的技术难题，因为如果要实现复杂环境下机器人自主的导航，必须要像人一样，有自己的双眼去认识周围环境，熟悉并形成周围环境的构成，对自己进行定位。

1985 年卡内基梅隆大学的 Hans P.Moravec Alberto Elfes 在 ICRA（International Conference on Robotics and Automation）会议上发表了《High Resolution Maps from Wide Angle Sonar》，利用超声波的反射测量机器人到周边的距离，初步实现了室内环境轮廓等二维信息的探测，同时期来自 MIT 人工智能实验室的 Rodney Brooks 发表了《Visual Map Making for a Mobile Robot》，提出了应用于智能机器人的视觉导航技术。

1987 《Consistent Integration and Propagation of Disparate Sensor Observations》作者 Hugh Durrant-Whyte 提出了基于矩阵计算的融合算法，来把多种传感器的不连续数据进行处理，利用相关性地图得到全局的三维信息；

1989 年在移动机器人重建和更新环境 3-D 信息的问题中，Ayache and Faugera 介绍了一种全新的思路，实现了基于扩展卡尔曼滤波的 SLAM 算法，同时把位置信息以及不确定性都包括进去，用 EKF 矩阵去描述并滤除传感器测量带来的误差。

1991 年 Philippe Moutarlier 第一个完整地实现了 SLAM 系统，但是算法的效率还比较低，获取到的有效的环境信息很有限；等到 1995 年，Hugh Durrant-Whyte 比较系统地归纳了解决 SLAM 问题可行的思路；

1997 年，Lu 和 Milios 实现了一致性的位姿估计，通过保留本地的关键帧并获取帧相关的空间信息，再基于最大相似性准则来合并所有的空间上的关系，生成完整可靠的地图如下：

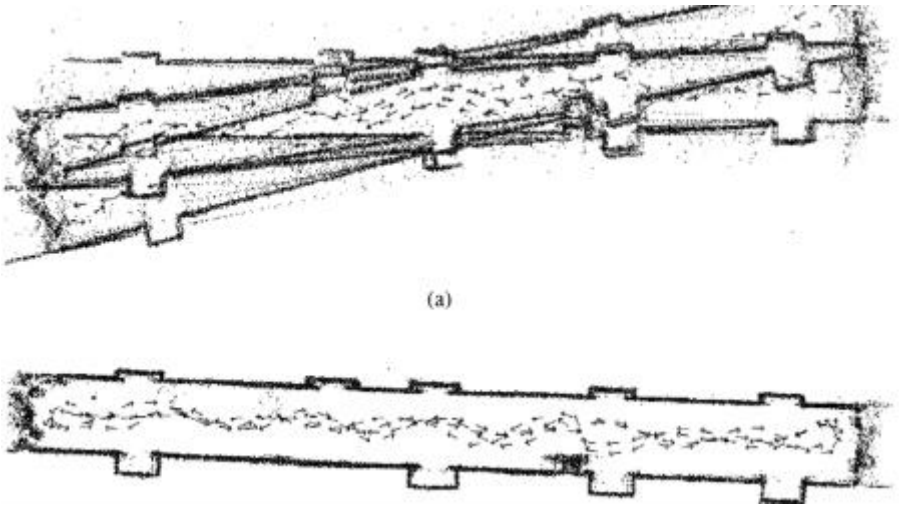


图 2-1 生成地图

1999 年，Jose A. Castellanos 等开发并实地测试了 SPmap（the symmetries and perturbations map）算法，通过引入对位置信息不确定性因素的一般相似性估计，他们利用 2-D 激光测距仪实现了与实际运动非常符合的实验结果；

同样是在 1999 年，Gutmann 和 Konolige 提出了闭环检测的方法，这一方法旨在通过新采集到的数据帧与之前的关键帧进行对比，找出可能的已经存在的关键帧匹配，这样可以极大地减少了累积误差。

2001 年 Guivant 和 Nebot 对 SLAM 算法的实时性进行了优化，使得实时生成相对于环境信息的轨迹图成为可能，如下图所示：

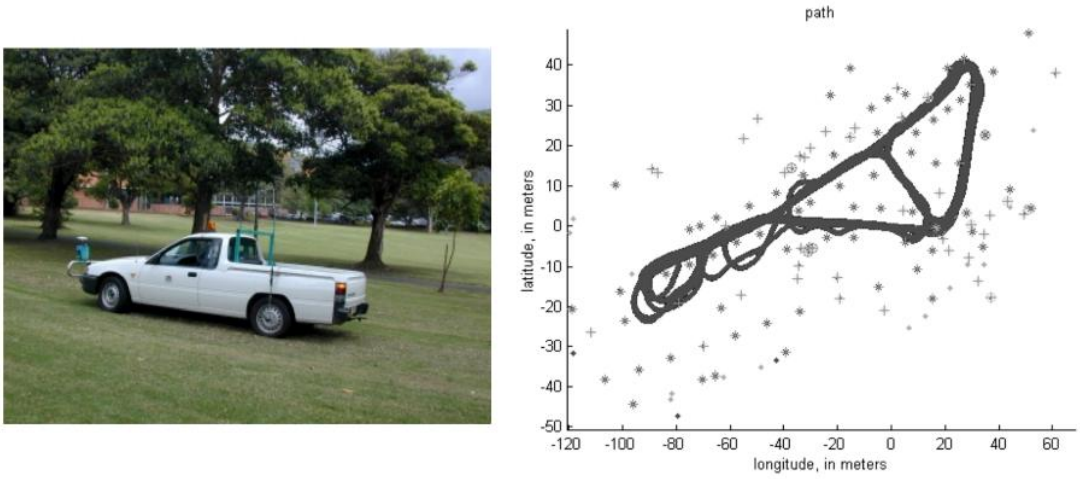


图 2-2 2001 年轨迹生成

2002 年的 ICRA 上 Paul Newsman 进一步提出了利用激光测量进行的实时 SLAM，Montemerlo 和 Thrun 开发了 FastSLAM 算法，极大地加快了地图重建的效

率，目前所生成的地图仍为 2 维的。

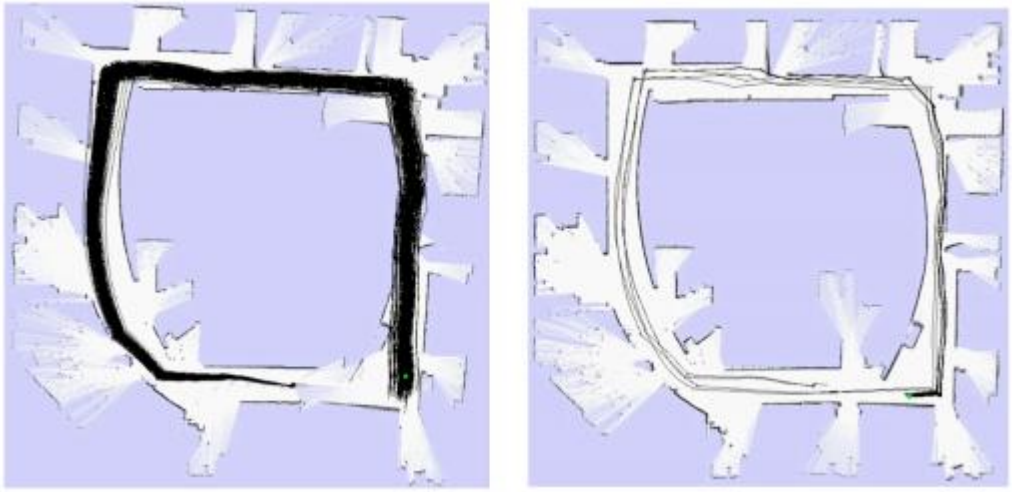


图 2-3 2 维重建效果

2004 年 Ryan 和 Hanu 利用姿态和图像完成了基于视觉的 SLAM，融合了“零漂移”的相机测量以及导航传感器的数据，再经过闭环检测来去除推算的静态误差。

2006 年，Dellaert 和 kaess 提出了 SAM（Smoothing and Mapping）方式，建立测量的雅各比矩阵和最小的平方计算的方程，在对机器人运动中的图像建模计算，获取到机器人位姿与路标性点，从而完成地图的重建；

2007 年 klein 和 Murray 两位一起提出了 PTAM 算法（Parallel Tracking and Mapping）；

2011 年 Necombe 等利用 Kinect 完成了 3 维信息的精密获取，来自微软的研究小组和多伦多大学等的研究者一起，完成了实时动态的对物体表面的 3 维重建；隔年 Extension of KinectFusion 取得了成功，重建效果比较理想

图 2-4 3 维地图生成

2013 年出现的基于点云片拼接到姿态图的 SLAM 优化，将 SLAM 技术引入了新的时代。

下图反映出了 1997 年到 2014 年度之间，在线的位子姿估计的取得的进展。

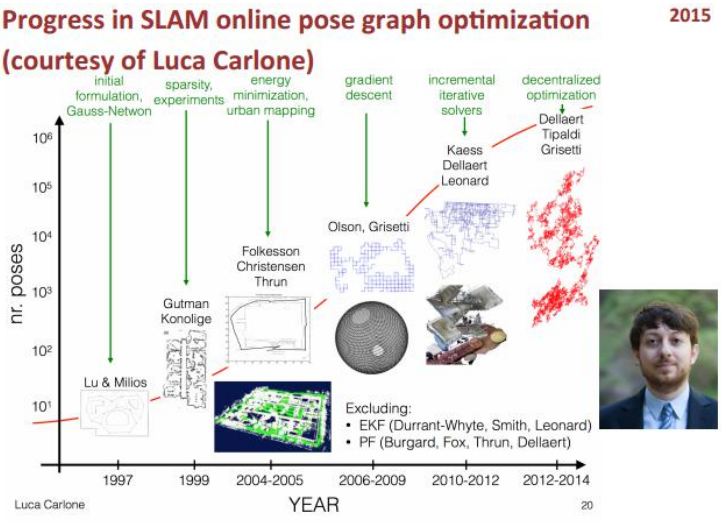


图 2-5 生成地图

3 三维点云算法测试

3.1 TSDF 的概念与测试

TSDF, Truncated Signed Distance Function，对于现在所处理的三维问题来说，是在我们定义的的范围内，建立一个 3D 网格坐标系，并且将这个均匀地划分为网格。例如我们已有的 3D 空间，在该坐标系中，我们在这个 3D grid 当中有 100 * 100 * 100 个单元格，每个格子有一个对应的 TSDF 的值。

首先，我们会获得当前时刻相机对应于参考世界的旋转和平移，以及相机获取到的当前帧的 RGB 图像及深度图像。3D 网格位于参考世界中，目标点在网格坐标系中有一个代表其位置的空间点，该空间点在当前相机图像上的映射点为像素点位置，而在深度图中，则可以找到其对应的深度，可以得到改像素点对应的三维点。通过将该三维点映射到参考世界中，计算得到它到参考世界的原点的距离。需要说明的是，3D 网格中的点的坐标有正有负。网格模型如下图所示：

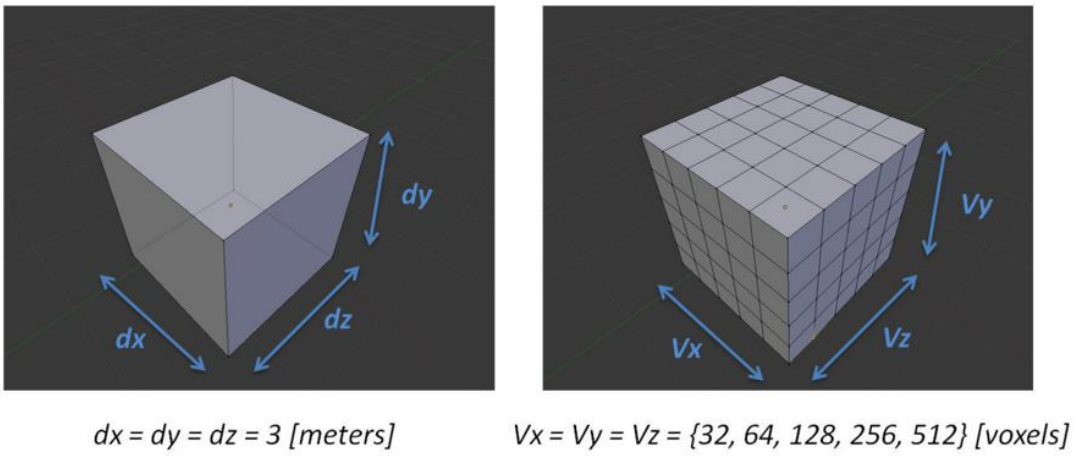


图 3-1 网格模型图

一般而言正负跨越的地方，就是三维曲面的表面，下图为来自 PCL 点云库官网的示例图片。

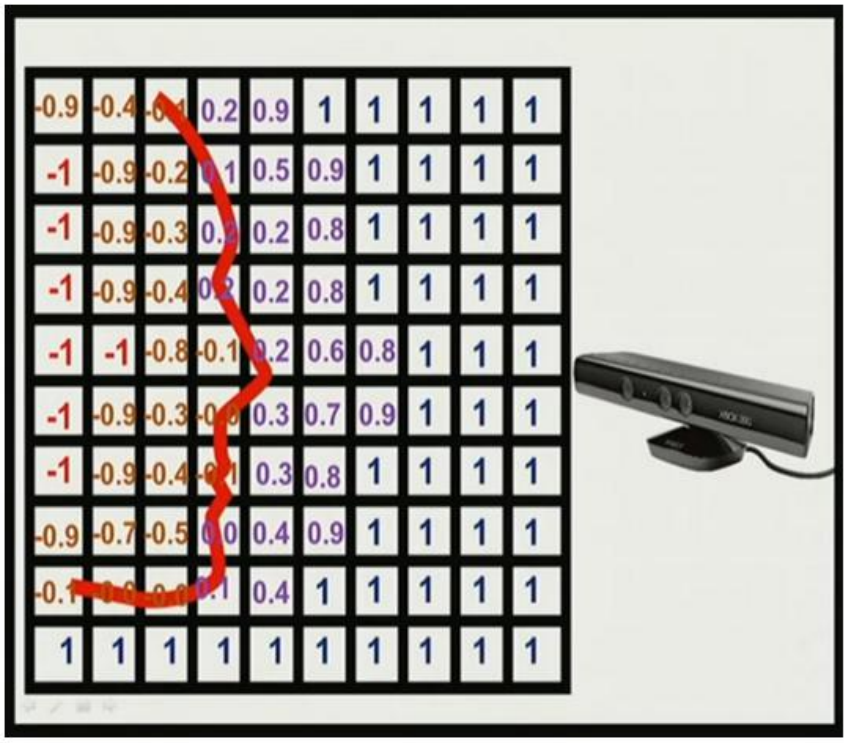


图 3-2 正负边界标示图

3.2 相机模型与坐标变化

常见的相机模型有五大类，总体上可以分为线性模型与非线性模型，这里采用比较常用的针孔模型，针孔模型也是相机的线性模型。

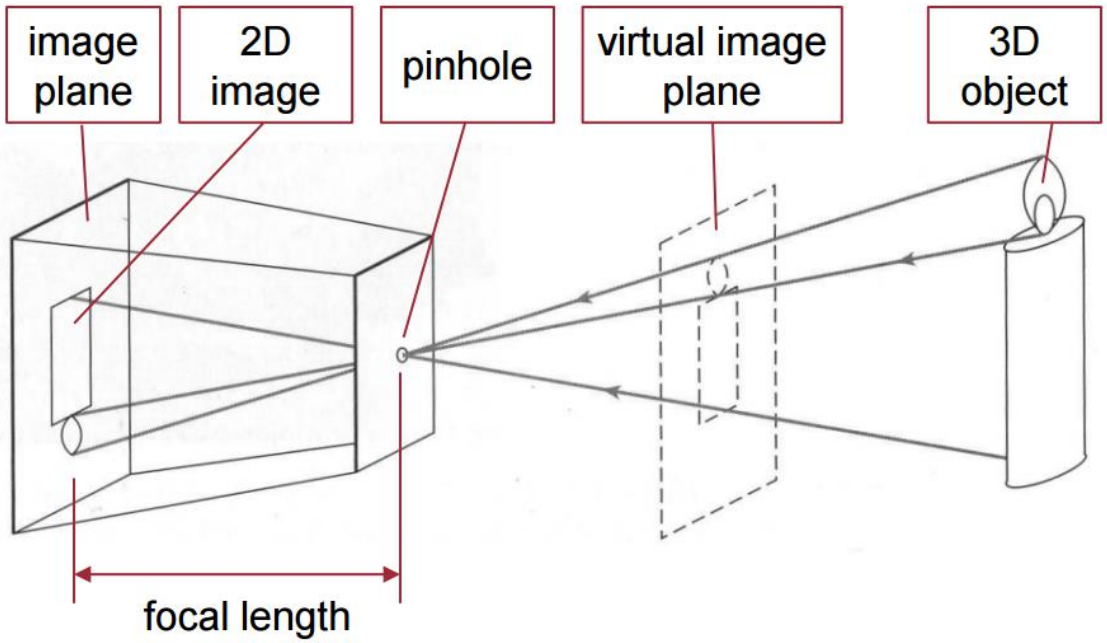


图 3-3 针孔相机模型

如果这个世界可以用一个点云模型来描述，那么点云有 $X=\{x_1,x_2,...,x_n\}$ 来表示，每一个元素都由 6 个分量：r，g，b，x，y，z 组成。其中 r、g、b 表示空间点的颜色子空间，而 x、y、z 则表示空间点的位置子空间。

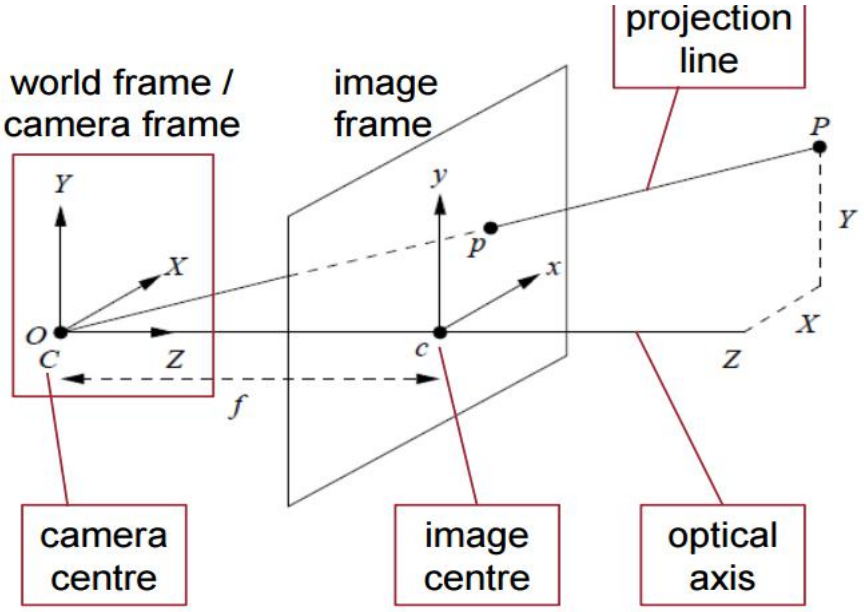


图 3-4 生成地图

相应于这样的点云表示约定，并联系相机的针孔模型，我们可以得到一个空间点[x,y,z]和它在图像中的像素坐标[u,v,d](d 指深度数据) 的对应关系：

$$\begin{aligned} u &= \frac{x \bullet f_x}{z} + c_x \\ v &= \frac{y \bullet f_y}{z} + c_y \end{aligned} \tag{3-1}$$

同样，逆推也可以得到相应的从已知的(u,v,d)得到(x,y,z)的表示方式。如下：

$$\begin{aligned} z &= d / s \\ x &= (u - c_x) \cdot z / f_x \\ y &= (v - c_y) \cdot z / f_y \end{aligned} \tag{3-2}$$

相机内参矩阵的组成 f_x, f_y, c_x, c_y

对于每一个相机而言，这些参数基本都是固定不变的，而且参数的具体值可以通过标定来获取。那么，此时就可以用简单的矩阵模型来描述每个点的空间位置与像素坐标来表示。

$$s \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = C \cdot \left(R \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} + t \right) \tag{3-3}$$

前一小节中说明了旋转、位移对坐标的影响，上式中的 R 代表旋转矩阵，t 则代表位移矢量。s 代表 scaling factor，表示深度图中的数据与实际距离的比例，通常计为 1000。

3.3 PCL 点云库与 3 维拼接

3.3.1 PCL（Point Cloud Library）点云库

PCL（Point Cloud Library），是一个大型跨平台开源 C++编程库，在吸收了前人点云相关研究基础。起初是 ROS（Robot Operating System）下的开源项目，主要应用于机器人研究应用领域，随着各个算法模块的积累，于 2011 年独立出来，正式联合 3D 信息处理的行业佼佼者，组建起来了强大的开发维护团队，以多所知名大学、研究所和相关硬件、软件公司为主。截止目前，它的发展非常迅速，不断有新的合作伙伴加入。它实现了很多功能，比如大量点云相关的算法，还有高效的数据结构，涉及到一系列的点云获取、滤波、分割，还有配准、检索、特征提取、识别、追踪以及曲面重建、可视化等。支持多种操作系统平台，可在 Windows、Linux、Android、Mac OS X、部分嵌入式实时系统上运行。如果说 OpenCV 是 2D 信息获取与处理的种子库，那么 PCL 就在 3D 信息获取与处理上具有同等地位，PCL 也可以免费进行商业和学术应用。本次毕设项目编写的 C++代码均采用了它的 1.7 版本中的 API 接口与函数。

图 3-5 合作开发联盟

3.3.2 点云生成

为了弄明白 SLAM 算法工作的原理，我们首先用同一位置、同一角度、同一相机拍摄的同一场景的 RGB 图与深度图来完成 2D 图像到 3D 图像的转换。

图 3-6 RGB 图

图 3-7 深度图

Linux 下使用 pcl_viewer 软件查看点云截图如下，其中 pcl_viewer 为ＰＣＬ点云库中的软件，

图 3-8 单帧拼接效果

3.3.3 两张图片拼接

在实现真实环境的 3 维重建时，我们所需要的往往不是单幅图片的点云生成，而是使用多张图片并综合 RGB 信息、深度信息，进行 3D 拼接生成环境的点云。用于测试的图片如下所示：

图 3-9 RGB 图 1

图 3-10 RGB 图 2

图 3-11 深度图 1

图 3-12 深度图 2

细细对比可以发现，用于拼接的两幅图片尽管非常相似，拍摄的角度却有细微的差别，而下面的深度图则与 RGB 图一一对应。要实现这两幅图片的拼接，我们首先要做的，是对图片本身进行特征提取。在 OpenCV 库中，提供了多种的图片

特征检测函数，编程实现所采用的函数位于 OpenCV 库的 feature2d 模块中。在计算“关键点”之后，我们就可以用“描述子”来计算关键点周围的像素。编写代码时，只需要用 cv::FeatureDetector 和 cv::DescriptorExtractor 来计算。提取到的关键点效果如下：

图 3-13 特诊检测

在提取到特征点之后，我们就需要进行两幅图像的特征匹配。OpenCV 提供了丰富的特征匹配的算法，这里选用近似最近邻的匹配算法。通过引入 cv::FlannBasedMatcher 和向量容器 vector< cv::DMatch >，调用 matches.match()函数就可以返回匹配的结果，经过优化和阈值滤波的特征匹配效果如下：

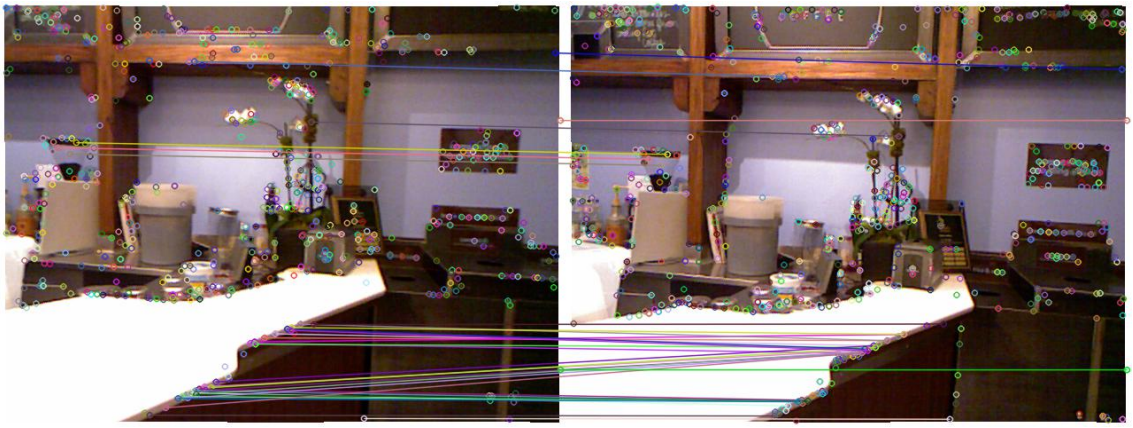


图 3-14 两帧图片特征匹配

经过特征匹配的两幅图片，可以把图片中相同的部分通过算法对应起来，这些部分在图像 3 维拼接时，在三维网格坐标系中是重合的，那么借助于这样一些特征点的位置还原，我们就可以借助数学模型求解出第二帧图相对于第一帧图的旋转与位移，这一点也是 SLAM 研究中最基础的部分。场地建图的核心，就在于多帧图片的准确匹配和拼接，如果求解得到的旋转和位移矩阵有误差，那么随着图片的增多，就会有误差的不断累积，最终拼接得到的点云就与实际环境相差太远。而如果求解旋转与位移矩阵的算法太过复杂，运算量大，当图片量大或者图片帧率极快时，在处理平台计算能力有限、没有进行并行计算时，就很难实现实时的点云生成。项目测试中，使用的是 PnP 求解，在程序中用 rvec 和 tvec 分别存储位移和旋转的信息。计算出的图像之间的运动用图像表示如下：

图 3-15 帧间运动计算

最终由两帧图片拼接得到的效果截图如下，

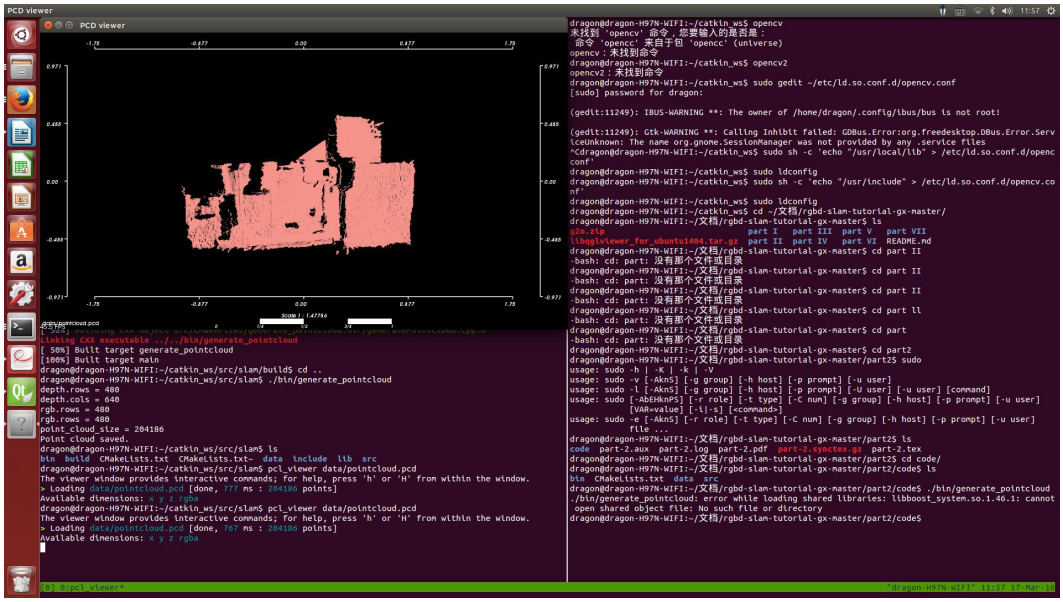


图 3-16 两帧图片拼接

3.3.4 基于图片集的拼接

在真正的 SLAM 地图构建中，需要对庞大的图像数据进行处理，当计算机本身没有开启并行计算或者应用英伟达显卡的 cuda 加速时，就比较考验 CPU 的性能。多帧图片的拼接难度除了较大的运算量之外，还必须考虑到误差的积累，所以拼

接过程中除了使用上节提到的特征匹配、运动计算，还需要进行滤波，一般常用的有网格滤波、粒子滤波或者扩展卡尔曼滤波。项目测试时使用为 PCL 点云库提供的网格滤波函数，实测效果满足计算需要。

用于图片拼接的数据集采用 tum 的数据集，在 intel i5 4570CPU 上运行的效率为 4 帧/s 左右。基于 39 张图片拼接的效果如下：

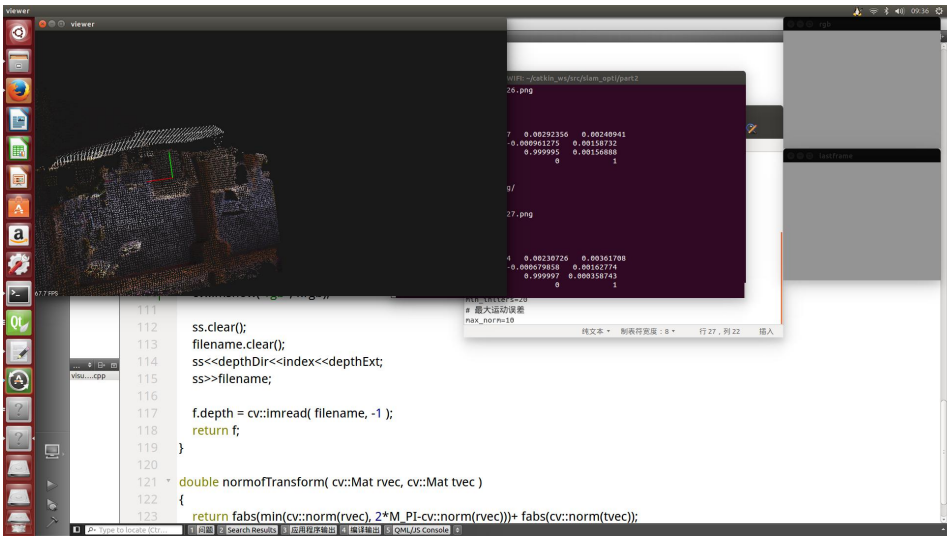


图 3-17 多帧连续图片拼接

整个公开数据集共有 800 多帧图片，但是最终当程序运行到第 39 帧时，因为特征匹配的误差过大，以至于可用的良好匹配特征点低于设定的阈值 10 个，接下来的点云指针指向的内容为空，程序报错中断。由于阈值设定了匹配的最少特征点，可行的方式不是调低阈值，而是采用鲁棒性、识别率更高的匹配算法。由于项目需求在于搭建基于飞行器的 SLAM 系统，不再做更多深究。至此，地图重建的基本原理测试完成。

4 系统硬件组成

4.1 系统概述与框图

经过不断的调整，最终所得到的整套系统由 Pixhawk 飞控系统，四旋翼机架与无刷电机，飞行器挂载的 miniPC，Kinect for Windows 2.0，以及地面的 intel i5 主机作为图像处理工作站，电源系统组成。原理框图如下：

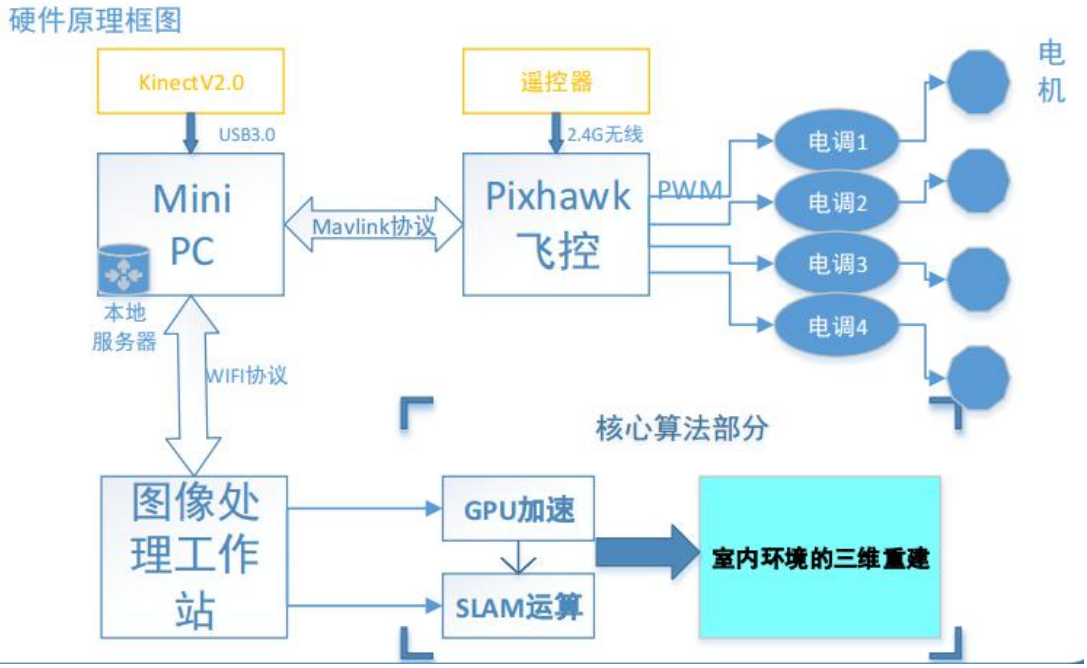
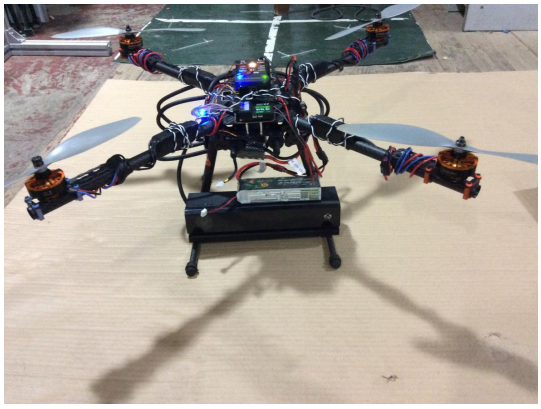


图 4-1 系统框图

最终的 SLAM 算法是在图像处理工作站上完成的。系统实物图如下：



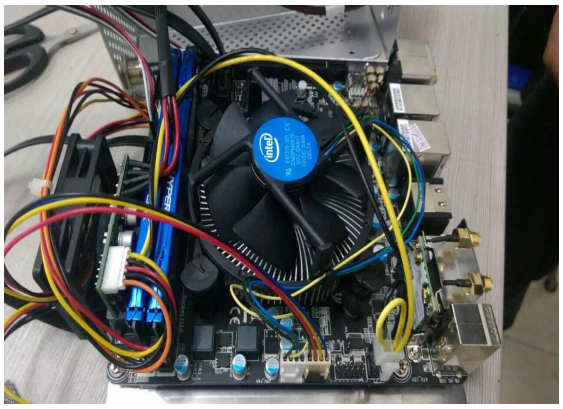


图 4-2 SLAM 系统实物图

4.2 Pixhawk 飞控组成

Pixhawk 系列开源飞控，由著名的 3DR 公司从 APM 系列开源飞控系统全面升级而得到，主控核心 MCU 为意法半导体公司生产的 STM32F427，具有如下特性：

(1) 168MHz/252 MIPS Cortex-M4；

(2) 可运行 uttx TTOS 实时操作系统；

(3) 同时有多达 14 路 PWM/舵机输出（其中 8 个带有失效保护功能，可人工设定。6 个可用于输入，全部支持高压舵机）；

(4) 有大量的外设接口（UART，I2C，SPI，CAN）

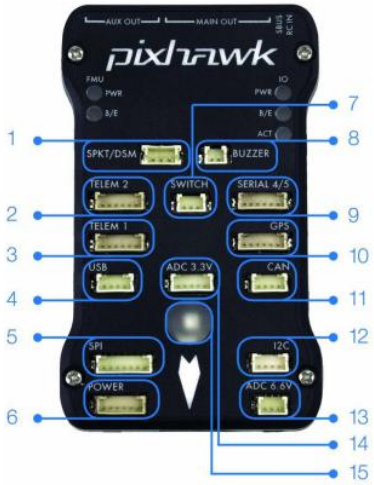


图 4-3 pixhawk 接口图

其中，各个接口的定义如下：

(1) Spektrum DSM 接收机

(2) 遥测（电台遥测）

(3) 遥测（屏幕上显示）

(4) USB

(5) SPI （串行外设接口）总线

(6) 电源模块

(7) 安全开关按钮

(8) 蜂鸣器

(9) 串行

(10) GPS 模块

(11) CAN （controller area network）总线

(12) I2C 分流器或罗盘模块

(13) 模拟至数字转换器 6.6 V

(14) 模拟至数字转换器 3.3 V

(15) LED 指示器

4.3 Kinect2.0 介绍与测试

图 4-4 Kinect 实物效果图

Kinect2.0 在 Kinect1.0 的基础上，配备的红外深度传感相机，还有颜色相机，性能均得到提升，实现了硬件与算法全面的升级。第一则是深度传感：可以呈现出空前惊艳的 3D 视觉效果，因为有了更高的深度保真，和大幅改进的噪声基底。能够看到清晰更小的物体，并提高骨骼追踪的稳定性。 第二体现为 1080p 高清视频：可以呈现高清晰画质，以及流畅的场景效果，产生强大的视觉冲击和震撼效果。Kinect2.0 极大地增强并改进视频通信、视频分析的应用程序，同时提供捕捉静态图像和打造高质量、增强现实场景的功能。 除了以上所说的，Kinect2.0 具备更宽广的视角，视野更广：相机利用宽广的视角，能够拍摄到更大范围的图像。

论文前面提到 Kinect 提供了一种比较廉价的方式来获取到带有深度的图像，在对 Kinect 的性能进行测试时，我们集中测试了如下的性能，证明 Kinect 的确是非常好用的图像输入传感器：

4.3.1 RGB 相机拍照测试

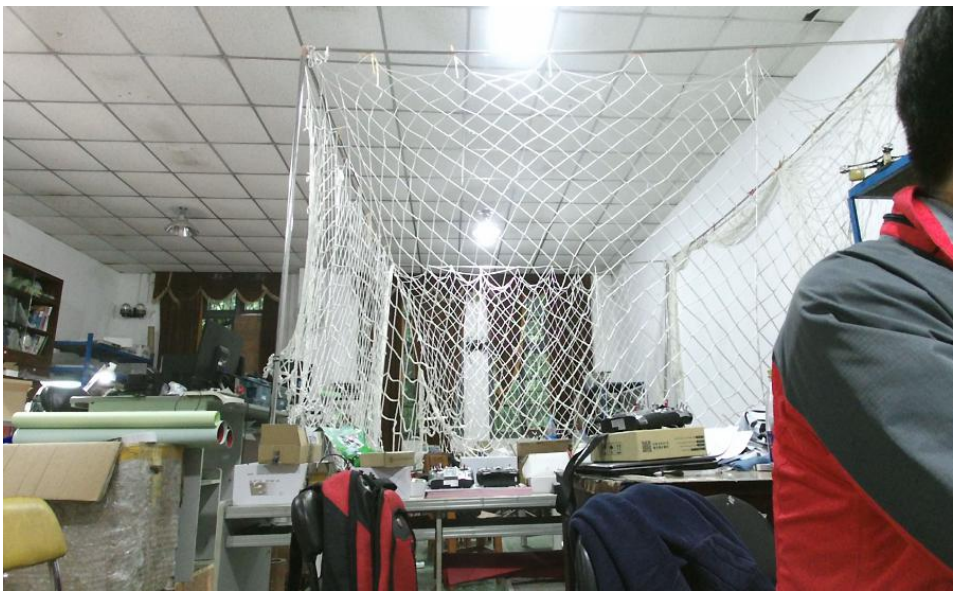


图 4-5 RGB 实际拍摄图

4.3.2 测试点云的生成

图 4-6 Kinect 点云生成

4.3.3 红外摄像头测试深度信息

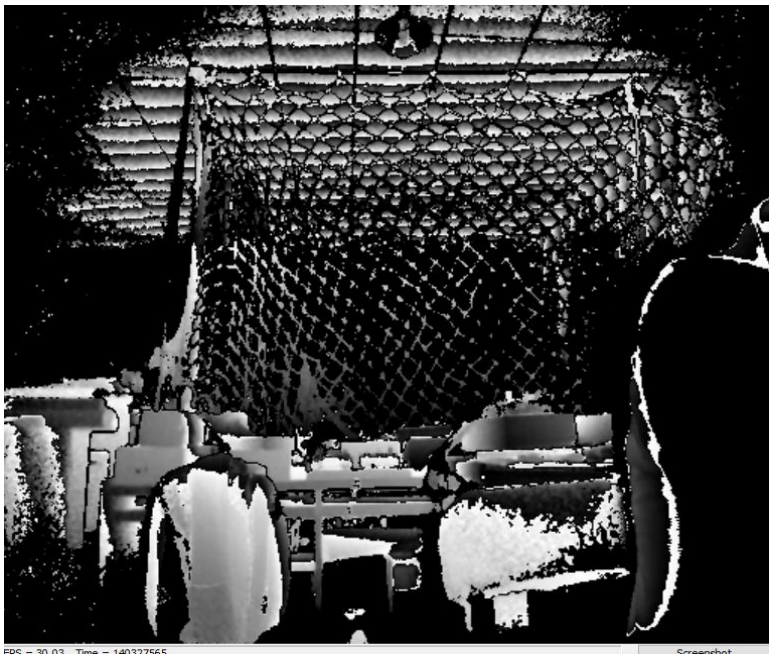


图 4-7 深度图

4.3.4 图像融合的测试

图 4-8 Kinect 效果融合图

4.3.5 深度摄像头获取到的图像

图 4-9 IR 摄像头图像

4.3.6 姿势识别

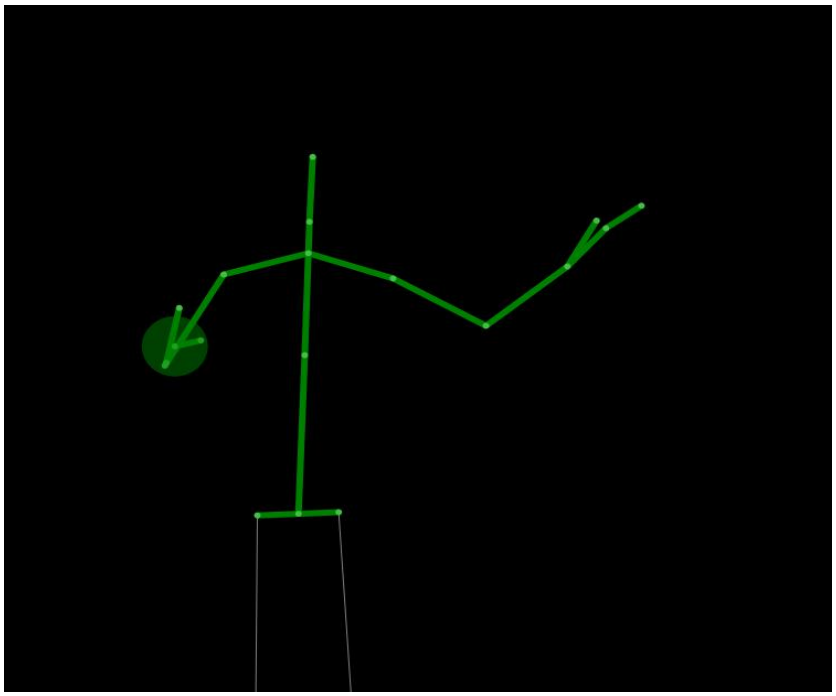


图 4-10 姿态图

测试结论：Kinect 通过安装驱动，可以正常地工作并传回图像，之后的算法测试也正常。原始的图片拍摄效果以及处理得到的图像数据，都相当清晰可靠，而在之后进行校准后，点云的效果会更好。

4.4 图像处理工作站

作为整个系统最关键的部分，图像处理工作站承担了处理视觉传感器数据、运行 SLAM 算法并进行环境 3 维重建的功能，因此在系统配置上必须要合理考量，

最终确定下来的配置参数为：intel i5 4570 CPU，主频 3Ghz，睿频可以达到 3.5Ghz；双 8G 采用三星存储颗粒的内存条；256G 固态硬盘，搭配 500G 机械硬盘；mini 主板。系统运行 win10 双系统和 ubuntu14.04，同时在 ubuntu 上配置好 ROS（机器人操作系统）indigo 版本。

5 系统软件的实现

5.1 飞行器飞控系统的架构

Pixhawk 飞控，内部运行的固件版本为开源的 ardupilot，代码在 github 上可以下载到。系统的软件架构图如下：

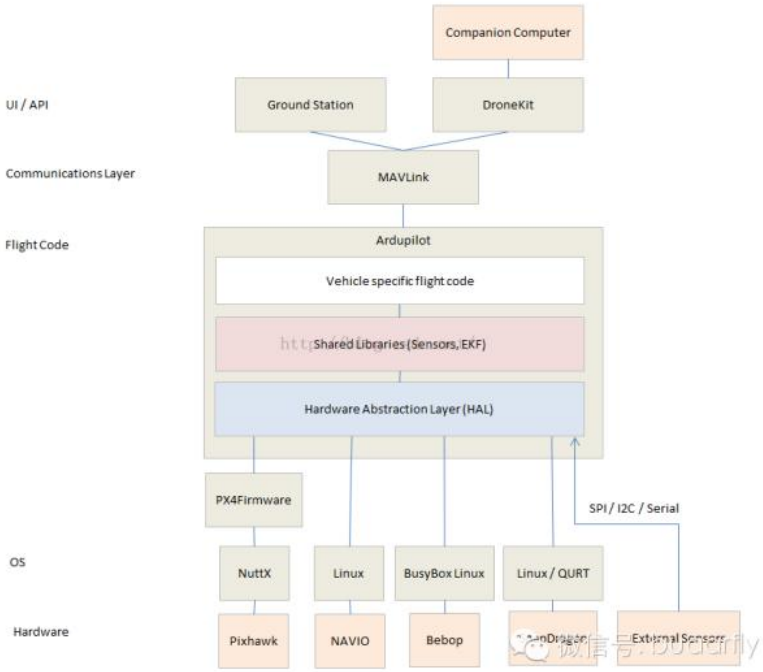


图 5-1 软件架构框图

5.2 图像处理站开发环境

由于这样一个项目需要的软件支持库相当庞大，加上 linux 下环境配置具有一定的难度，因此项目初期在进行库安装上走了很多弯路，可行的软件方案为：

(1) Ubuntu 14.04 LTS 64 位；

(2) ROS indigo 版本；

(3) OpenCV 2.4.12；

(4) PCL 点云库 1.7.1；

(5) Kinect 驱动 iai-kinect2；

(6) Qt Creator 5.0

实际测试时，发现 OpenCV 的 nonfree 模块还需要从独立源单独安装。

5.3 关于 ROS

5.3.1 ROS 综述

ROS（机器人操作系统，RobotOperatingSystem），是专为机器人软件开发所设计的，可以完全模拟电脑的操作系统架构。同时，它也是一个开源的元级操作系统（后操作系统），能够提供类似于操作系统的服务，这就包括硬件抽象描述、、共用功能执行、程序包管理，底层驱动程序管理、程序间消息传递。它也提供一些工具库，用于完成获取、建立、编写和执行多机融合程序的全套任务。

ROS 的运行架构，深入探究，其本质是一种模块间进行 P2P 的松耦合的网络连接的处理架构，它使用 ROS 通信模块，实现包括基于服务的同步 RPC（远程过程调用）通讯、还有参数服务器上的数据存储、基于 Topic 的异步数据流通讯。ROS 的首要设计目标，则是在机器人研发领域最大程度提高代码复用。

5.3.2 设计目标

图 5-2 ROS 应用

ROS 的首要设计目标，是在机器人研发领域提高代码复用率。ROS 是一种分布式处理框架（又名 Nodes）。这样的话，就使可执行文件能被单独设计，并且在运行时松散耦合。

5.3.3 主要特点

ROS 的主要特点可以归纳为以下几条：

（1）点对点设计

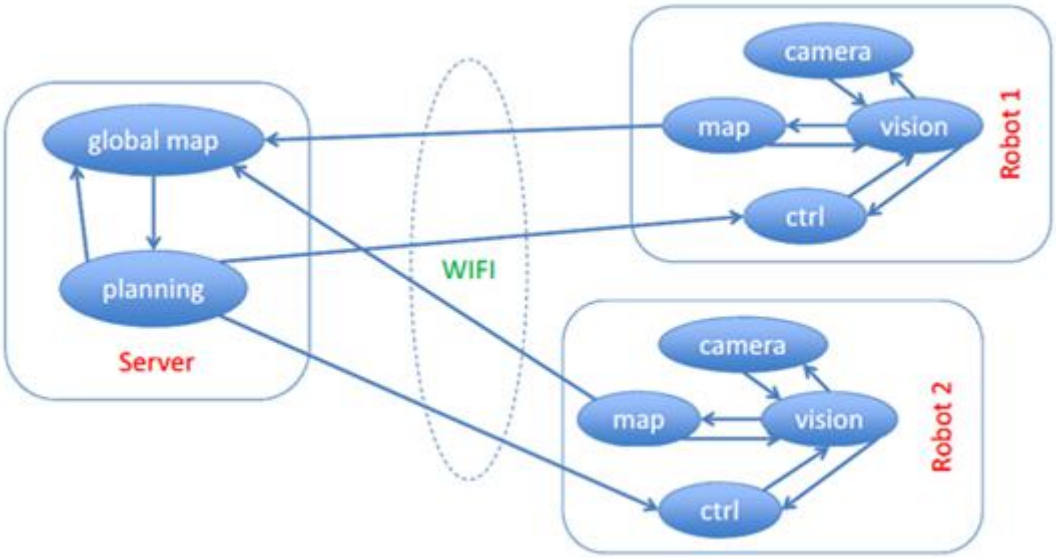


图 5-3 点对点设计

一个使用 ROS 的系统，一定会包括一系列进程，这些进程可以存在于多个不同的主机，并且通过端对端的拓扑结构，进行运行过程中的联系。ROS 的点对点设计，以及服务和节点管理器等机制，可以分散实时计算压力，特别是由计算机视觉和语音识别等功能带来的，因此能够适应多机器人、智能体通信与决策遇到的挑战。

（2）多语言支持

当我们在写代码的时候，会比较偏向某一些编程语言。那么这些偏好，都一般是编程人员在每种语言的编程时间、调试效果，以及语法、执行效率以及各种技术甚至是文化的原因导致的。为了解决这些问题，ROS 设计者经过考虑之后，将它设计成了语言中立性的框架结构。ROS 现在支持许多种不同的语言，例如 C++、Python、Octave 和 LISP，也包含其他语言的多种接口实现。

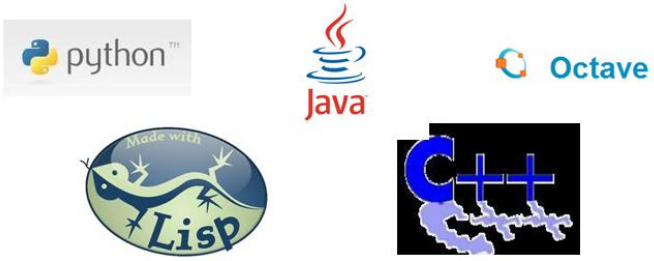


图 5-4 所支持的语言

其结果就是一种语言无关的消息处理，能够让多种语言自由的混合和匹配使用，而不用担心编译器不能支持或者通讯出错。

（3）系统极大精简

ROS 系统，具有一个很重要的特点，那就是模块化。各模块中的代码可以单独编译，而且编译使用的 CMake 工具本身，就很容易实现精简的理念。ROS 总是试图把复杂的代码，全部封装在库里，然后创建了一些小的应用程序为 ROS 显示库的功能，这样就允许了重新使用，以及对简单的代码超越代码原型进行移植。还有另外一点，单元的测试在代码于库中分散后，也变得非常的容易，一个单独的测试程序，就可以完成库测试。

（4）工具包丰富

为了管理复杂的 ROS 软件框架，系统的设计者利用了大量的小工具去编译，并运行丰富多样的 ROS 组建，从而设计成了内核，而不是去选择构建一个庞大的开发与运行环境。

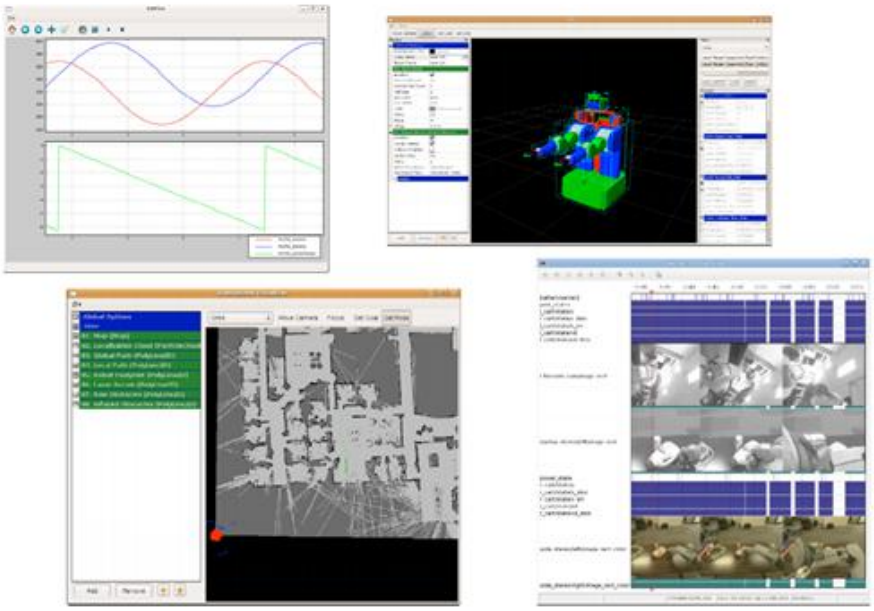


图 5-5 ROS 中的工具软件

这些工具担任了多种任务，例如，组织源代码的结构，获取和设置配置参数，还有形象化端对端的拓扑连接，频带使用宽度的测量，以及生动的信息数据的描绘，甚至包括自动生成文档等等。系统的设计者希望能把所有的代码模块化。因为效率上的损失远远是稳定性和管理的复杂性上无法弥补的。

5.4 Kinect2.0 校准与标定

Kinect2.0 校准的过程要比一般的 RGB 摄像头要复杂，计算量也会大很多。这里我们采用 university of Bremen 的 Thiemo Wiedemeyer 教授开发的 kinect 校准方法。这里我们使用 5x7x0.03 的棋盘格，标定的时候，我们要注意用于标定的棋盘格必须平铺在木板或者其它平面上，同时最好用三角架同时固定 Kinect 和棋盘格，在校准的过程中，启动校准软件后，需要将棋盘格进行左右的摆放角度变化，按下空格键进行图片的采集，每个参数文件需要采集的图片最好多一些，在 100 张左右，这样计算得到的 Kinect 参数会更加精确，之后用于三维重建的效果会更好。实际测试图如下：

1、在 linux 下的终端中输入命令用于启动 kinect 的驱动：rosrun kinect2_bridge kinect2_bridge _fps_limit:=2；

2、在主文件夹中创建数据存放目录：mkdir ~/kinect_cal_data; cd ~/kinect_cal_data；

3、记录颜色相机的数据，按 space 键进行记录: rosrun kinect2_calibration kinect2_calibration chess5x7x0.03 record color；

4、运行命令计算并校正相机内部参数: rosrun kinect2_calibration kinect2_calibration chess5x7x0.03 calibrate color；

5、记录红外相机拍摄的画面: rosrun kinect2_calibration kinect2_calibration chess5x7x0.03 record ir；

6、计算并校正红外相机的内部参数: rosrun kinect2_calibration kinect2_calibration chess5x7x0.03 calibrate ir；

7、同时记录两个相机拍摄到的画面: rosrun kinect2_calibration kinect2_calibration chess5x7x0.03 record sync；

8、计算内部参数: rosrun kinect2_calibration kinect2_calibration chess5x7x0.03 calibrate sync；

9、计算深度相机的参数: rosrun kinect2_calibration kinect2_calibration chess5x7x0.03 calibrate depth；

10、在 kinect2_bridge 的输出参数第一行中得到 kinect 的序列号；

11、在 kinect2_bridge/data/$serial 创建文件目录：roscd kinect2_bridge/data; mkdir 012526541941；

12、把你之前得到的参数文件拷贝到目标文件夹：calib_color.yaml calib_depth.yaml calib_ir.yaml calib_pose.yaml；

13、重启 kinect2_bridge，查看校正后的效果

图 5-6 Kinect 深度相机校正

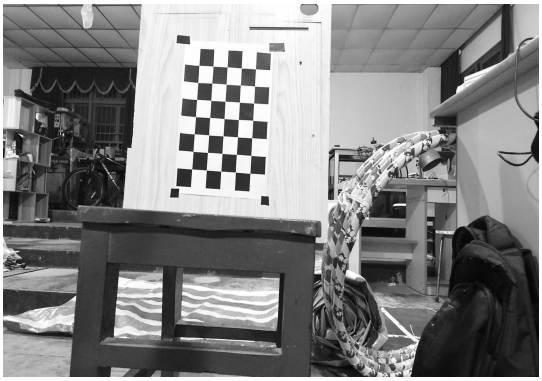


图 5-7 Kinect RGB 相机校正

5.4 3D 视频信息的网络传输

当在飞行器上要使用 Kinect 作为图像传感器的时候，除了要考虑如何把 Kinect 正确安装到飞机上之外，还需要编写 Kinect 数据无线传输的代码。这里的传输方式采用 socket 大容量的短距离无线通信，以客户端的形式访问服务器端的 Kinect 数据，客户端的软件开发环境为 ROS 次级系统，同时传输 RGB 图像数据、深度数据、还有声音等。程序编写时参考了卡内基梅隆大学的开源代码，并结合项目需要对网络通信的参数进行了重新的设置。服务器端软件运行在预装有 windows 的迷你 PC 上运行，采用 Csharp 进行编写，并可以作为系统进程在 windows 里不间断运行，通过安装 Kinect 官方的 SDK 软件，服务器软件可以把 Kinect 的数据正确地读出来，并通过网络通信发出去。而运行在 Linux 上的客户端则可以通过访问充当服务器的 mini 电脑的 IP 地址，来实时读取 kinect 的数据。

5.5 G2O 原理与算法实践

G2O 是一个图形图形的方式来描述一种解决非线性最小方差问题的通用框架，非线性的最小二乘法问题，通常是通过在当前的状态下构建一个线性的系统来解决这个典型的问题。一个比较可行的用于解决这样的线性系统的方法是预处理共轭梯度法（ PCG），Konolige 、Montemerlo 和 Thrun 使用 PCG 作为一种高效的用于解决有较大稀疏的约束系统。由于它针对特殊的问题会有较高的效率，G2O 包含了应用在一个雅可比块的预先调整器中实现稀疏 PCG 的解决方法。近期， Dellaert 和他的同事提出了一种叫做 的系统，他们通过使用稀疏矩阵求解器来执行这个系统。 Kaess 等人，在 Dellaert 的基础上进行了转化，形成了一种叫做 iSAM 系统，这个系统可以更新与非线性最小二乘问题相关的线性矩阵。Konolige 等人，通过利用典型的线性系统的稀疏结构来展示如何高效的构造线性矩阵。然而，后一种方法受到 2D 位姿图的约束。在 G2O 中我们会用到和这些系统相似的方法。我们的系统可以应用到 SLAM 和 BA 优化问题以及他们的所有转换当中，例如：有地标的 2DSLAM、单目相机的 BA 问题、立体视觉的 BA 问题等。我们只要把观测和运动信息丢到求解器里就行。这个优化器会为我们求出机器人 的轨迹和路标位置。如下图，红点是路标，蓝色箭头是机器人的位置和转角（2D SLAM）。细心观察，会发现它往右偏转了一些。：

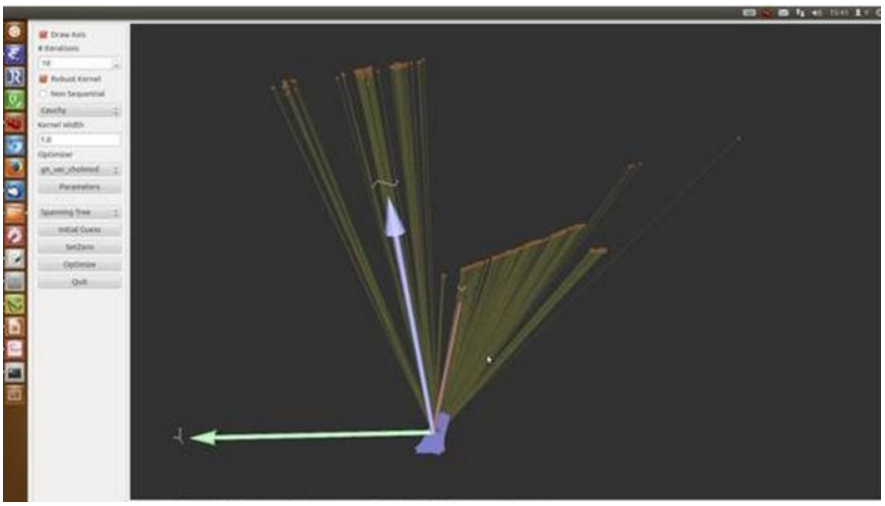


图 5-8 g2o 软件包

5.6 基于 ORB-SLAM 算法内核的实践与优化

5.6.1 ORB-SLAM 原理

ORB-SLAM 是 15 年提出的一个单目 SLAM 算法，地图还是单目常见的稀疏特征点图。所有优化问题全是基于 g2o 来进行计算，特征匹配直接用 ORB 的描述符，回环检测用 BOW。随后，作者推出了针对 RGB-D 相机和双目相机的版本，简单实用，效果好，代码结构清晰，命名规范。从工程的角度看，非常适合移植到实际项目。

ORB-SLAM 对所有的任务采用相同的特征，追踪、地图构建、重定位和闭环控制。从而使得系统更有效率、简单可靠。

ORB 特征，可以应用于实时图像系统中，具有很好的旋转不变特性。同时也可应用于实时室内和户外环境操作。另外，它可以独立工作，因为其视图内容关联的特性，追踪和地图构建可在局部视图关联中处理。在 SLAM 研究领域，基于位置优化的实时闭环控制，称作必须路径。原理上是通过生成树构建，而生成树由系统、闭环控制链接，还有视图内容关联强边缘进行维持。实时相机重定位，具有明显的旋转不变特性，这样的话，就使得跟踪丢失可以一遍遍地重做，同时获取到的地图也可以重复使用。选择不同的模型，可以创建不同的平面或者非平面的初始化地图，另外自动的、具有良好鲁棒性的初始化过程也是基于模型而进行选择。

另外一方面，有一个问题是，大量地图云点和关键帧，需要经过严格的挑选，这里就必须找到一个最合适的办法。好的挑选方法可以增强追踪的鲁棒性，同时能够辅助去除冗余的关键帧 以增强程序的可操作性。项目中，我在公共数据集上，主要对程序在室内的环境进行了评估：(1) 相机的位置比现在最新的方法更精确，它通过像素扩展集进行优化，并非特征的重映射；(2) 提高基于特征的方法的准确性；(3) 单目 SLAM 可以通过滤波方案初始化，每一帧估计地图特征都用到滤波器。位置和相机位姿，一起关联，但是处理连续的图像帧需要进行大量运算，线性误差会累积，然而现在的方法中，由于地图构建并不依赖于帧率，基于关键帧的方法，用筛选的关键帧估计地图，采用精确的 BA 优化。

ORB-SLAM 系统，一共整合了三个并行的线程，其中包括追踪、局部地图构

建，还有闭环回路控制。追踪视通过每帧图像定位相机位置，决定时机插入一个新的关键帧。我们可以先通过前一帧图像帧初始化特征匹配，采用运动 BA 优化位姿。如果追踪过程中目标丢失，位置识别模块执行全局重定位。一旦获得最初的相机位姿估计和特征匹配，通过内容相似视图的关键帧，可以提取一个局部可视化地图。

然后全面进行匹配，比如映射搜索局部地图云点，并根据匹配优化相机位姿。最后，追踪到图片像素点，局部地图构建处理新的关键帧，在相机位姿的环境中，可以执行局部 BA 优化重构。根据交互视图中的标志进行判断，如已经连接的关键帧，搜索新关键帧中未匹配的 ORB 特征的对应关系，来三角化新的云点。有时尽管已经创建了新的云点，在基于追踪线程过程新收集到信息，但是为了获得高质量的云点，根据云点筛选策略可能会临时删除一些点。局部地图构建删除冗余关键帧。对每个新的关键帧都要进行闭环搜索，以确认是否形成闭环。如果闭环被侦测到，我们就可以计算相似变换来查看闭环的累积误差。这样闭环的两端就可以对齐，重复的云点被融合。

5.6.2 编程要点

编程中使用的结构由下列组成：

(1)、Frame，包括计算出的姿态，图像金字塔，ORB 特征点；

(2)、关键 KeyFrame，包含 Frame 中的所有信息；

(3)、地图上点 MapPoint，包括坐标、平均观测方向、引用关键帧（该 3d 点产生时的关键帧）、观测到该点的所有关键帧和相应的特征点 (cv::KeyPoint)、最佳描述子（与其他关键帧对该点描述符距离较小的）、有效观测距离范围 [dmin,dmax]

(4)、 cv::KeyPoint，包括图像坐标、尺度、方向、描述子；

(5)、Convisibility Graph，由代表关键帧的节点，边组成；

(6) Essential Graph：Covisibility Graph 的最大生成树

(7) 视觉词带：词汇树，DBoW2 数据库

整个过程用语言可以描述为：封装当前图像为 Frame，进行初始估计，匹配关键帧优化姿态；检测新的关键帧，从而产生新的 3d 点，合并 3d 点并更新 Covisiblity

图、Enssential 图和 MapPoint 类，最后是运行搜索算法，进行闭环检测、融合，以及 Enssential 图姿态优化。

5.7 粒子滤波

5.7.1 滤波原理

由于 SLAM 算法中，匹配图像时存在大量的误差点，如果不进行处理，误差会累积，同时导致系统的运算量大大增大。采用滤波算法，进行似然估计，预测下一步的图像运动与变换，才可以在基于重要性采样的蒙特卡洛模拟方法中，估计后验滤波概率。估计时，需要利用所有的观测数据。同时，每次新的观测数据来到，都需要用算法重新计算整个状态序列的重要性权值。序贯重要性采样既然作为粒子滤波的基础，是在将统计学中的序贯分析方法，应用到蒙特卡洛方法，从而实现后验滤波概率密度的递推估计。假设重要性概率密度函数 $q(x_{0:k}\mid y_{1:k})$ 可以分解为

$$q(x_{0:k}\mid y_{1:k})=q(x_{0:k-1}\mid y_{1:k-1})q(x_k\mid x_{0:k-1},y_{1:k}) \tag{5-1}$$

设系统状态是一个马尔可夫过程，且给定系统状态下各次观测独立，则有

$$p(x_{0:k})=p(x_0)\prod_{i=1}^{k}p(x_i\mid x_{i-1}) \tag{5-2}$$

$$p(y_{1:k}\mid x_{1:k})=\prod_{i=1}^{k}p(y_i\mid x_i) \tag{5-3}$$

后验概率密度函数的递归形式可以表示为

$$\begin{aligned}p(x_{0:k}\mid Y_k)&=\frac{p(y_k\mid x_{0:k},Y_{k-1})p(x_{0:k}\mid Y_{k-1})}{p(y_k\mid Y_{k-1})}\\&=\frac{p(y_k\mid x_{0:k},Y_{k-1})p(x_k\mid x_{0:k-1},Y_{k-1})p(x_{0:k-1}\mid Y_{k-1})}{p(y_k\mid Y_{k-1})}\\&=\frac{p(y_k\mid x_k)p(x_k\mid x_{k-1})p(x_{0:k-1}\mid Y_{k-1})}{p(y_k\mid Y_{k-1})}\end{aligned} \tag{5-4}$$

粒子权值 $w_k^{(i)}$ 的递归形式可以表示为

$$w_k^{(i)} \propto \frac{p(x_{0:k}^{(i)} \mid Y_k)}{q(x_{0:k}^{(i)} \mid Y_k)}$$

$$= \frac{p(y_k \mid x_k^{(i)}) p(x_k^{(i)} \mid x_{k-1}^{(i)}) p(x_{0:k-1}^{(i)} \mid Y_{k-1})}{q(x_k^{(i)} \mid x_{0:k-1}^{(i)}, Y_k) q(x_{0:k-1}^{(i)} \mid Y_{k-1})}$$

$$= w_{k-1}^{(i)} \frac{p(y_k \mid x_k^{(i)}) p(x_k^{(i)} \mid x_{k-1}^{(i)})}{q(x_k^{(i)} \mid x_{0:k-1}^{(i)}, Y_k)} \tag{5-5}$$

通常，需要对粒子权值进行归一化处理，即

$$\tilde{w}_k^{(i)} = \frac{w_k^{(i)}}{\sum_{i=1}^{N} w_k^{(i)}} \tag{5-6}$$

序贯重要性采样算法，可以从重要性概率密度函数中，逐步生成采样粒子，并随着测量值的依次到来递推求得相应的权值，最终以粒子加权和的形式，来对后验滤波概率密度进行描述，进而得到状态估计。序贯重要性采样算法的流程，也可以用如下伪代码描述：

$$[\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^{N}] = SIS(\{x_{k-1}^{(i)}, w_{k-1}^{(i)}\}_{i=1}^{N}, Y_k) \tag{5-7}$$

For i=1:N

(1)时间更新，根据重要性参考函数 $q(x_k^{(i)} \mid x_{0:k-1}^{(i)}, Y_k)$ 生成采样粒子 $x_k^{(i)}$；

(2)量测更新，根据最新观测值计算粒子权值 $w_k^{(i)}$；

粒子权值归一化，并计算目标状态。

为了得到正确的状态估计，通常希望粒子权值的方差尽可能趋近于零。然而，序贯蒙特卡洛模拟方法一般都存在权值退化问题。在实际计算中，经过数次迭代，只有少数粒子的权值较大，其余粒子的权值可忽略不计。粒子权值的方差随着时间增大，状态空间中的有效粒子数较少。随着无效采样粒子数目的增加，使得大量的计算浪费在对估计后验滤波概率分布几乎不起作用的粒子更新上，使得估计性能下降。通常采用有效粒子数 N_{eff} 来衡量粒子权值的退化程度，即

$$N_{eff} = N / (1 + \mathrm{var}(w_k^{*(i)})) \tag{5-8}$$

$$w_k^{*(i)} = \frac{p(x_k^{(i)} \mid y_{1:k})}{q(x_k^{(i)} \mid x_{k-1}^{(i)}, y_{1:k})} \tag{5-9}$$

有效粒子数越小，表明权值退化越严重。在实际计算中，有效粒子数 N_{eff} 可以近似为

$$\hat{N}_{eff} \approx \frac{1}{\sum_{i=1}^{N}(w_k^{(i)})^2} \tag{5-10}$$

另外，在进行序贯重要性采样时，若 $\hat{N}_{eff}$ 小于事先设定的阈值，则应当采取一些措施，对结果加以控制。最直接的方法来克服序贯重要性采样算法权值退化是增加粒子数，而这会造成计算量的相应增加，影响计算的实时性。因此，一般采用以下两种途径：(1)选择合适的重要性概率密度函数；(2)在序贯重要性采样之后，采用重采样方法。

5.7.2 重要密度函数的选择

重要性概率密度函数的选择对粒子滤波的性能有很大影响，在设计与实现粒子滤波器的过程中十分重要。在工程应用中，通常选取状态变量的转移概率密度函数 $p(x_k \mid x_{k-1})$ 作为重要性概率密度函数。此时，粒子的权值为

$$w_k^{(i)} = w_{k-1}^{(i)} p(y_k \mid x_k^{(i)}) \tag{5-11}$$

转移概率的形式简单且易于实现，在观测精度不高的场合，将其作为重要性概率密度函数可以取得较好的滤波效果。然而，采用转移概率密度函数作为重要性概率密度函数没有考虑最新观测数据所提供的信息，而且会比较繁琐，从中抽取的样本与真实后验分布产生的样本存在一定的偏差，特别是当观测模型具有较高的精度或预测先验与似然函数之间重叠部分较少时，这种偏差尤为明显，当误差积累大一定误差之后，需要进行改进。

选择重要性概率密度函数的一个标准是使得粒子权值 $\{w_k^{(i)}\}_{i=1}^{N}$ 的方差最小。Doucet 等给出的最优重要性概率密度函数为

$$\begin{aligned} q(x_k^{(i)} \mid x_{k-1}^{(i)}, y_k) &= p(x_k^{(i)} \mid x_{k-1}^{(i)}, y_k) \\ &= \frac{p(y_k \mid x_k^{(i)}, x_{k-1}^{(i)}) p(x_k^{(i)} \mid x_{k-1}^{(i)})}{p(y_k \mid x_{k-1}^{(i)})} \\ &= \frac{p(y_k \mid x_k^{(i)}) p(x_k^{(i)} \mid x_{k-1}^{(i)})}{p(y_k \mid x_{k-1}^{(i)})} \end{aligned} \tag{5-12}$$

此时，粒子的权值为

$$w_k^{(i)} = w_{k-1}^{(i)} p(y_k \mid x_{k-1}^{(i)}) \tag{5-13}$$

以 $p(x_k^{(i)} \mid x_{k-1}^{(i)}, y_k)$ 作为重要性概率密度函数需要对其直接采样。此外，只有在 x_k 为有限离散状态或 $p(x_k^{(i)} \mid x_{k-1}^{(i)}, y_k)$ 为高斯函数时，$p(y_k \mid x_{k-1}^{(i)})$ 才存在解析解。在实际情况中，构造最优重要性概率密度函数，与直接从后验概率分布中抽取样本，两者的困难程度等同。另外，从最优重要性概率密度函数的表达形式来看，就会发现难度在于产生下一个预测粒子依赖于已有的粒子和最新的观测数据，这对于设计重要性概率密度函数具有重要的指导作用，即应当更加有效利用最新的观测信息，在易于采样实现的基础上，将更多的粒子移动到似然函数值较高的区域，如图 5-9 所示。

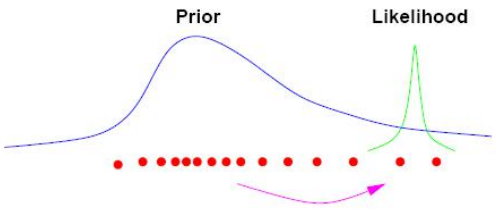


图 5-9 移动粒子至高似然区域

辅助粒子滤波算法，能够利用 k 时刻的信息，将 $k-1$ 时刻最有前途(预测似然度大)的粒子扩展到 k 时刻，从而生成采样粒子。这种机制与 SIR 滤波器相比，当粒子的似然函数，出现位于先验分布的尾部或者是似然函数形状比较狭窄的曲线特征时，辅助粒子滤波能够得到更精确的估计结果。辅助粒子滤波，引入辅助变量 m 来表示 $k-1$ 时刻的粒子列表，应用贝叶斯定理，联合概率密度函数 $p(x_k, m \mid y_{1:k})$ 可以描述为

$$\begin{aligned} p(x_k, m \mid y_{1:k}) &\propto p(y_k \mid x_k) p(x_k, m \mid y_{1:k-1}) \\ &= p(y_k \mid x_k) p(x_t \mid m, y_{1:k-1}) p(m \mid y_{1:k-1}) \\ &= p(y_k \mid x_k^m) p(x_k \mid x_{k-1}^m) w_{k-1}^m \end{aligned} \tag{5-14}$$

生成 $\{x_k^{(i)}, m^{(i)}\}_{i=1}^N$ 的重要性概率密度函数 $q(x_k, m \mid x_{0:k-1}, y_{1:k})$ 为

$$q(x_k, m \mid x_{0:k-1}, y_{1:k}) \propto p(y_k \mid \mu_k^m) p(x_k \mid x_{k-1}^m) w_{k-1}^m \tag{5-15}$$

其中 μ_k^m 为由 $\{x_{k-1}^{(i)}\}_{i=1}^N$ 预测出的与 x_k 相关的特征，可以是采样值 $\mu_k^m \sim p(x_k \mid x_{k-1}^m)$ 或预测均值 $\mu_k^m = E\{x_k \mid x_{k-1}^m\}$ 。

定义 $q(x_k \mid m, y_{1:k}) = p(x_k \mid x_{k-1}^m)$ ，由于

$$q(x_k, m \mid y_{1:k}) = q(x_k \mid m, y_{1:k}) q(m \mid y_{1:k}) \tag{5-16}$$

则有

$$q(m \mid y_{1:k}) = p(y_k \mid \mu_k^m) w_{k-1}^m \tag{5-17}$$

此时，粒子权值 $w_k^{(i)}$ 为

$$w_k^{(i)} \propto w_k^{m^{(i)}} \frac{p(y_k \mid x_k^{(i)}) p(x_k^i \mid x_k^{m^{(i)}})}{q(x_k, m \mid x_{0:k-1}^m, y_k)} = \frac{p(y_k \mid x_k^{(i)})}{p(y_k \mid \mu_k^{m^{(i)}})} \tag{5-18}$$

最后需要说明，在本项目中，参考开源项目采用了局部线性化的方法来逼近 $p(x_k \mid x_{k-1}, y_k)$ ，这也是另一种有效地提高粒子采样效率的方法。扩展 Kalman 粒子滤波，以及 Uncented 粒子滤波算法，会在滤波的每一步迭代过程中，最先引入最新观测值，然后采用 UKF 或者 EKF 对各个粒子的状态进行更新，得到随机变量经非线性变换后的均值和方差，并将它作为重要性概率密度函数[138]。另外，利用似然函数的梯度信息，采用牛顿迭代[139]或均值漂移[140]等方法移动粒子至高似然区域，也是一种可行的方案，如图 2.5 所示。以上这些方法的共同特点，就是引导粒子到高似然区域，将最新的观测数据融入到系统状态的转移过程中，由此产生的预测粒子可较好地服从状态的后验概率分布，类似于热力学统计物理中的分子等粒子的运动，从而有效地减少描述后验概率密度函数所需的粒子数。

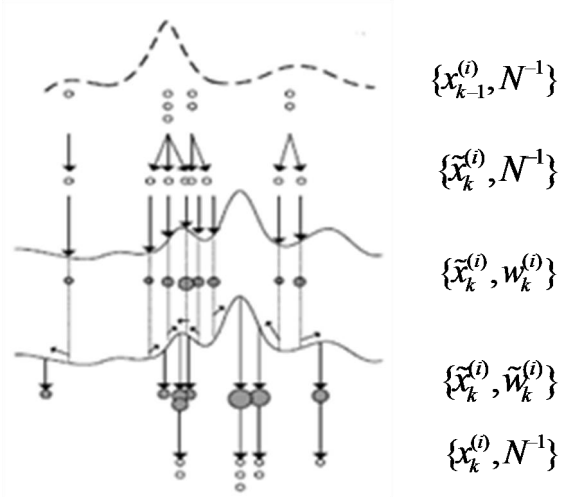


图 5-10 结合均值漂移的粒子滤波算法

6 SLAM 算法运行的实际效果

6.1 ORB-SLAM 算法测试

采用单目 SLAM 得到的路径追踪效果为：

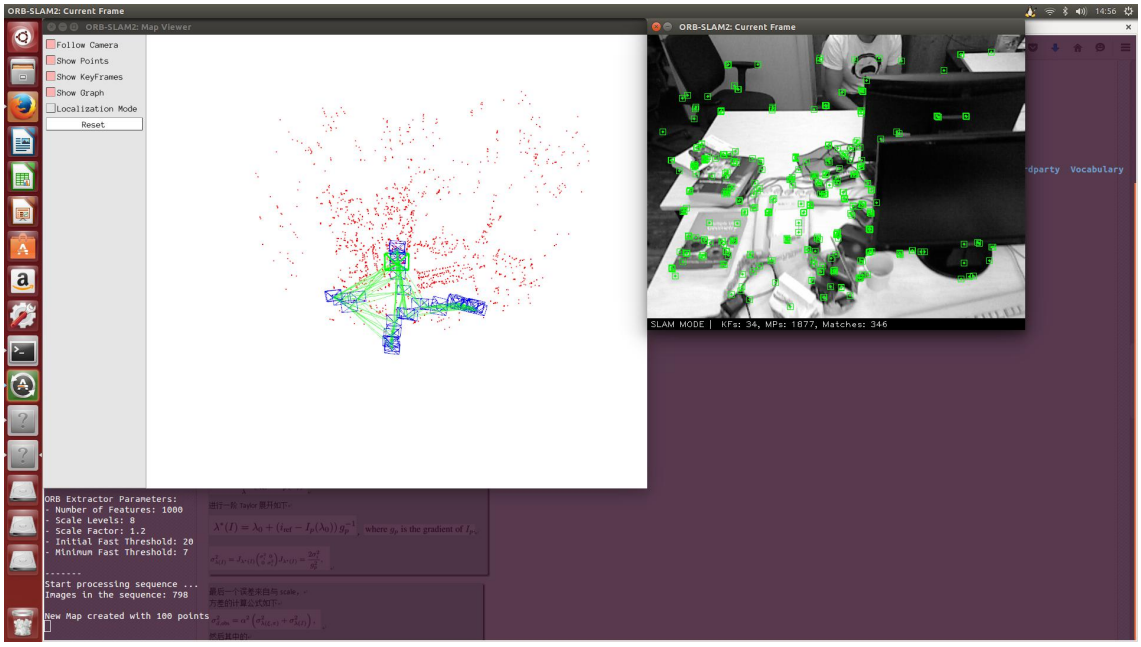


图 6-1 ORB-SLAM 测试

6.2 RGB-D SLAM 算法测试

考虑到飞行器可承载的重量，最终项目采用六旋翼（升力更大）和 kinect1.0（质量较轻，方便挂载）完成，利用 gazabo 仿真软件可以查看实时生成的房间的点云模型。

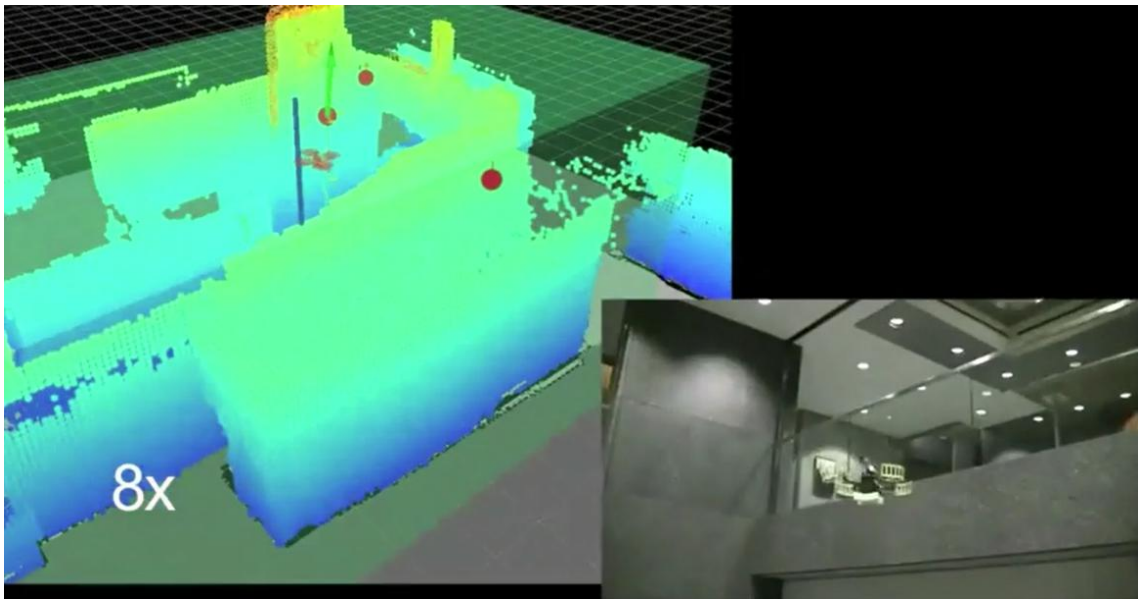


图 6-2 RGBD-SLAM 算法测试

参考文献

[1] G. Blais and M.D. Levine,” Registering Multiview Range Data toCreate 3D Computer Objects,” TR-CIM-93-16, Center for IntelligentMachines, McGill Univ., 1993

[2] P. Boulanger, G. Godin, and M. Rioux, “Applications of 3-D Active Vision to Rapid Prototyping and Reverse Engineering,” Proc. Third Int’l Conf. Rapid Prototyping, pp. 213-223, Dayton, Ohio, June 1992.

[3] Y. Chen and G. Medioni, “Object Modeling by Registration of Multiple Range Views,” Proc. IEEE Int’l Conf. Robotics and Automation, pp. 2,724-2,729, Sacramento, Calif., 1991.

[4] F. P. Ferrie, M. D. Levine, “Integrating Information from Multiple Views”, Proc. IEEE CS Workshop Computer Vision, pp. 117-122, Miami, Fla, 1987.

[5] H. Gagnon, M. Soucy, R. Bergevin, and D. Laurendeau, ”Registration of Multiple Range Views for Automatic 3-D Model Building”, Puoc. IEEE Conf. Computer Vision and Pattern Recognition, Seattle, Wash., June 1994

[6] R. I. Hartley, A. Zisserman. Multiple View Geometry. Cambridge University Press, 2000.

[7] http://grail.cs.washington.edu/projects/cpc/.

[8] M. Habbecke and L. Kobbelt. A Surface-growing Approach to Multi-view Stereo Reconstruction. In Proc. CVPR, 2007.

[9] Y. Furukawa and J. Ponce. Accurate, Dense, and Robustmultiview Stereopsis. In Proc. CVPR, 2007.

[10] Michael Goesele. Multi-view Stereo for Community Photo Collections, In Proc. ICCV 2007.

[11] Derek Bradley, Tamy Boubekeur, Wolfgang Heidrich. Accurate Multi-View Reconstruction Using Robust Binocular Stereo and Surface Meshing, In Proc. CVPR, 2008.

[12] Min H. Kim, Todd Alan Harvey, etc. 3D Imaging Spectroscopy for Measuring

Hyperspectral Patterns on Solid Objects. ACM Transactions on Graphics (TOG), Volume 31 Issue 4, July 2012, Article No. 38.

[13] 王新宇，基于计算机立体视觉的三维重建[D]，中南大学，2004，5；

[14] 林育斌，基于计算机视觉的三维重建技术研究[D]，天津大学，2010，6；

[15] 方磊，基于特征的图像序列三维场景重建技术研究[D]，华中科技大学，2006，6；

[16] 田文，多视图图像的快速三维场景重建[D]，华中科技大学，2010，6；

[17] 张勇，金学波，用单数码相机实现物体表面的三维重建[J]，计算机工程与设计，2008，6，29 卷 11 期；

[18] H. P. moravec. Towards Automatic Visual Obstacle Avoidance [C]. Proc. 5th International Joint Conference on Artificial Intelligence, 1977.

[19] C. Harris, M. Stephens. A Combined Corner and Edge Detector [C]. Proceedings of the 4th Alvey Vision Conference, 1988: 147-151.

[20] K. Mikolajcyk, C. Schmid, An Affine Invariant Interest Point Detector [C]. In Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada, 2002.

[21] D. G. Lowe. Object Recognition from Local Scale-invariant Features[C]. In Proceedings of the IEEE intl. Conference on Computer Vision (ICCV'1999): 1150–1157.

[22] D. G. Lowe. Distinctive Image Features From Scale-invariant Keypoints [J]. International Journal of Computer Vision, 60(2), 2004, 91-110.

[23] Y. Ke and R. Sukthankar, PCA-SIFT: "A More Distinctive Representation for Local Image Descriptors" [J], Proc. Conf. Computer Vision and Pattern Recognition, 2004: 511-517.

[24] H. Bay, T. Tuytelaars. SURF: Speeded Up Robust Features [C], 9th European Conference on Computer Vision (ECCV), 2006.

[25] E. Rublee, V. Rabaud, K. Konolige and G. Bradski. "ORB: An Effcient Alternative to SIFT or SURF" [C], Intenational Conference on Computer Vision, 2011.

[26] E. Rosten and T. Drummond. Machine Learning for High Speed Corner Detection [C]. In European Conference on Computer Vision, volume 1, 2006.

[27] M. Calonder, V. Lepetit, C. Strecha, P. Fua. BRIEF: Binary Robust Independent Elementary Features [C]. ECCV, 2010.

[28] O. Faugeras and Q. T. Luong. "The Fundamental Matrix: Theory Algorithms, and stability Analysis"[J], Int' 1J. Computer Vision, 17, PP. 43-45, 1996.

[29] H. C. Longuet-Higgins. "A Computer Algorithm for Reconstruction A Scene from Two Projections"[J], Nature, 293: 133-135.

[30] R. I. Hartley. In Defense of the Eight-Point Algorithm [J]. IEEE Transaction on Pattern Recognition and Machine Intelligence(S0162-8828), 1997, 19(6): 580-593.

[31] D. Nister. An Efficient Solution to The Five-point Relative Pose[J]. IEEE PAMI, 26(6):756–770, 2004.

[32] 吴福朝, 胡占义, 基本矩阵的 5 点和 4 点算法[J]. 自动化学报, 2002, 39(12): 175－180.

[33] H. Li and R. Hartley. Five-point Motion Estimation Made Easy [C]. ICPR, 2006.

[34] Z. Kukelova, M. Bujnak, T. Pajdla. Polynomial Eigenvalue Solutions to The 5-pt and 6-pt Relative Pose Problems[C], BMVC, 2008.

[35] 张红斌. 遥感图像拼接算法研究[D]. 西安电子科技大学，2006.

[36] 宁书年，吕松棠等，遥感图像处理及应用[M]，北京：地震出版社，1995: 105-110.

[37] Matthew Brown, D. G. Lowe. Automatic Panoramic Image Stitching using Invariant Features[J], International Journal of Computer Vision, v.74 n.1, p.59-73, August 2007

[38] P. Burt and E. Adelson. A Multiresolution Spline with Application to Image Mosaics[J].ACM Transactions on Graphics, 2(4):217–236,1983.

[39] Wei Wang, K. Ng Michael. A Variational Method for Multiple-Image Blending[J]. IEEE Transactions on Image Processing 21(4): 1809-1822 (2012).

[40] 李海生. Delaunay 三角剖分理论及可视化应用研究[M]. 哈尔滨：哈尔滨工业大学出版社,2010.

[41] M I. Shamos and D. Hoey. Closest-point Problems[J]. Proceedings of the 16th Annual Symposium on the Foundations of Computer Science, 1975.151-162.

[42] B A. Lewis and J S. Robinson. Triangulation of Planar Regions with Applications[J], The Computer Journal, 1978,21(4)：324-332.

[43] D T. Lee and B J. Schachter. Two Algorithms for Constructing a Delaunay Triangulation[J], Int. J. of Computer and Information Sciences, 1980,9(3).

[44] C L. Lawson. Software for C' Surface Interpolation[M]. Mathematical Software III. J. Rice, Ed. New York： Academic Press, 1977.

[45] A. Bowyer. Computing Dirichlet Tesselations[J], Computer Journal, 1981(24)：162-166.

[46] Watson D F. Computing the n-dimension Delaunay Tesselation with Application to Voronoi Polytopes. Computer Journal, 1981(24):167-172.

[47] S W.Sloan. A fast Algorithm for Constructing Delaunay Triangulations in the Plane[J]. Andvanced Engineering Software, 1987(9):34-55.

[48] G. Macedonia and M T. Pareschi. An Algorithm for the Triangulation of Arbitrarily Distributed Points: Applications to Volume Estimate and Terrain Fitting[J], Computers & Geosciences, 1991(17):859-874.

[49] De Floriani L and Puppo E. An on-line Algorithm for Constrained Delaunay Triangulation[J], CVGIP: Graphical Models and Image Processing, 1992,54(3):290-300.

[50] V J D. Tsai. Delaunay Triangulations in TIN Creation: an Overview and a Linear-time Algorithm[J]. Int. J. of GIS, 1993,7(6):501-524.

[51] P J. Green and R. Sibson. Computing Dirichlet Tesselations in the Plane[J]. The Computer Journal, 1978,21(2):168-173.

[52] K E. Brassel and D. Reif. Procedure to Generate Thiessen Polygons[J]. Geophysical Analysis, 1979(11):289-303

致　谢

此次毕业设计的工作的完成得到了广大师生同学的大力支持，特别是思飞工作室的队友与教授，不仅仅在理论知识的指导上，还是研究方法、思维方法乃至一般的处理事务的方法上都给予了我很大的帮助和深刻的启发。回顾刚开始接触 S 三维重建这一论题时，自己对这个论题充满着好奇和和兴趣，但特别担忧的又是自己在多视三维重建研究上的理论不足，甚至是有些这方面的东西从来没有接触过，心里产生了很大的压力、信心全无，开始着手做的时候真是举步维艰。但是在和机械创新基地的队员一起，讨论项目的方案，不断地改进方案，调试代码。华科图像所的杜敏学长在 Bug 调试上非常耐心，帮助我度过了难关。在此次此次毕业设计中，我的指导老师雷鑑铭老师在毕业设计的指导上给予的很大的帮助，特别是对学生毕业设计工作的关心和负责使我由衷敬佩。雷老师给了我足够的空间去自由探索，不为眼前的困难所吓倒。

感谢 ROS 机器人群、SLAM 研究者群的伙伴们，你们是当代中国研究机器人自动导航领域的尖兵，感谢清华大学的高翔博士，你的入门博客给予我很大的引导和启发，如果没有你生动有趣同时满满干货的文字，我估计还停留在无奈地看代码的地步，谢谢你为我开了一扇科研的窗，领略到做研究应有的大胆与谨严风格。同时，我要真诚感谢舍友、同学给予我的关心和帮助，并预祝他们在此次毕业设计之中取得优异成绩。

我还要感谢我的女朋友，是她的鼓励和引导使我走出了写论文的困境，能够冷静下来理清思路，全身心投入到论文的写作中。

感谢印刷与包装系的全体领导和老师，在学业上给予我的支持和帮助。

由衷感谢我的父母家人朋友在学习和生活上的无私关怀和帮助。

最后，向所有参加论文评审和答辩的老师表示衷心的感谢和崇高的敬意。

李小龙

2016 年 6 月于武汉