## 2.3 项目设计阶段

根据需求设计，经过项目成员讨论，设计了项目静态原型和产品架构。

经过大家的讨论，决定将项目拆分为采集端和云平台两个子系统。采集端负责使用传感器采集实际物体数据，并转换为3D点云，交付给云平台系统处理。云平台对采集端上传的数据进行分析存储和处理，并通过前端与运维人员和客户进行交互。

采集端子系统的设计交给小枘，小芮和阿真负责，云平台交给老王负责。大家开始紧锣密鼓设计子系统的架构。

对于采集端子系统，项目开始决定使用扫描机器人来进行数据采集。机器人采用ROS系统作为底层，基于ROS基础上加载传感器和算法进行数据采集和处理。需要解决的问题有点云的数据大小量级、点云数据处理的方案、机器人路径规划的方案、是否采用图像的三维重建来实现。一个房间采集后的点云数据规模在10GB-100GB量级，使用机器人进行处理需要很高的计算和存储能力。点云数据将上传至性能较为强大的服务器进行处理。机器人路径规划使用SLAM算法解决。ROS中常用SLAM算法有，HectorSLAM、Gmapping、LogoSLAM、KartoSLAM，其比较分析如下：

1) HectorSLAM是一种结合了鲁棒性较好的扫描匹方法2D SLAM方法和使用惯性传感系统的导航技术。传感器的要求：高更新频率小测量噪声的激光扫描仪.不需要里程计,使空中无人机与地面小车在不平坦区域运行存在运用的可能性.作者利用现代激光雷达的高更新率和低距离测量噪声，通过扫描匹配实时地对机器人运动进行估计。所以当只有低更新率的激光传感器时，即便测距估计很精确，对该系统都会出现一定的问题。

2) Gmapping是一种基于激光的SLAM算法，它已经集成在ROS中，是移动机器人中使用最多的SLAM算法。这个算法已经由Grisetti等人提出是一种基于 Rao-Blackwellized的粒子滤波的 SLAM方法。基于粒子滤波的算法用许多加权粒子表示路径的后验概率，每个粒子都给出一个重要性因子。但是，它们通常需要大量的粒子才能获得比较好的的结果，从而增加该算法的的计算复杂性。此外，与PF重采样过程相关的粒子退化耗尽问题也降低了算法的准确性。粒子退化问题包括在重采样阶段从样本集粒子中消除大量的粒子。发生这种情况是因为它们的重要性权重可能变得微不足道。因此，这意味着有一定的小概率时间会消除正确的假设的粒子。为了避免粒子的退化问题，已经开发了自适应重采样技术。

3)LagoSLAM 是线性近似图优化,不需要初始假设。基本的图优化slam的方法就是利用最小化非线性非凸代价函数.每次迭代, 解决局部凸近似的初始问题来更新图配置,过程迭代一定次数直到局部最小代价函数达到.。

4)KartoSLAM是基于图优化的方法，用高度优化和非迭代 cholesky矩阵进行稀疏系统解耦作为解．图优化方法利用图的均值表示地图，每个节点表示机器人轨迹的一个位置点和传感器测量数据集，箭头的指向的连接表示连续机器人位置点的运动，每个新节点加入，地图就会依据空间中的节点箭头的约束进行计算更新。

对于传感器的选择，

1)使用激光雷达

优势：a) 扫描精准b) 可以直接生成点云

劣势：a) 价格过高：低端激光雷达价格在2000左右，只支持，工作较慢，只支持2D，有效距离较短；高端雷达价格在3W以上，精度高，距离长。

2) 使用光学摄像头

优势：a) 价格便宜b) 有图像还原为模型的开源库

劣势：a) 不利于自规划路径的实现

经过经济型，可用性的分析，决定采用光学摄像头作为传感器，并依次为基础设计算法，将光学图像还原为3D点云。

对于三维重建技术，比较流行的是单、双目的重建。根据重建的稀疏程度不同，可以分为以下以下几类：

稀疏重建：通常是重建一些图像特征点的深度，这个在基于特征的视觉SLAM里比较常见，得到的特征点的深度可以用来计算相机位姿。稀疏重建在实际应用，比如检测，避障，不能满足需求。

半稠密重建：通常是重建图像纹理或梯度比较明显的区域，这些区域特征比较鲜明。半稠密重建在直接法视觉SLAM里比较常见。重建的三维点云相对稠密，可以满足部分应用需求。

稠密重建：稠密重建是对整个图像或者图像中的绝大部分像素进行重建。与稀疏、半稠密相比，稠密重建对场景的三维信息理解更全面，更能符合应用需求。但是，由于要重建的点云数量太多，相对耗时。

由于项目目标为得到室内环境的轮廓，因此决定采用半稠密重建来建立3D点云。

采集端的设计架构如图 1所示。



图 1

老王同学有丰富的大数据和前端开发经验，针对本项目，设计了云平台的结构，前端架构vue2+react，后台用Hadoop＋strom/spark＋分布式数据库的架构，架构设计如图 2所示。

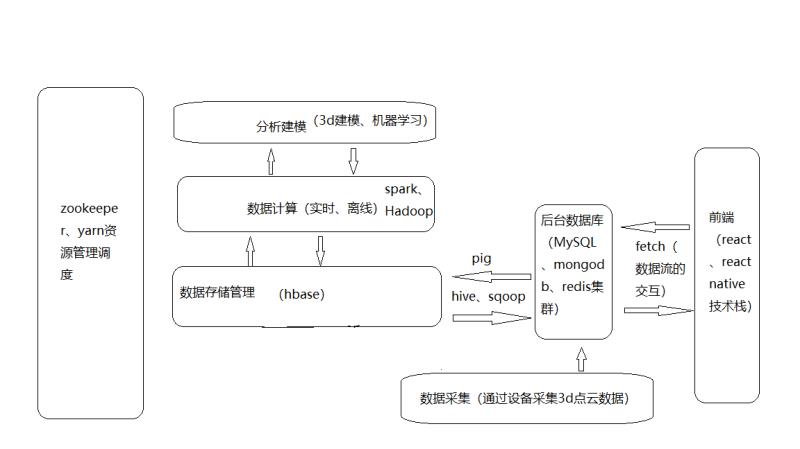


图 2

对于前端架构：vue2来开发组件，vuex负责多个组件间的数据通信和状态管理, vue-router为各组件的路由管理，parcel前端打包，fetch用来跟服务器进行异步通信（ajax）,用sass管理样式，布局采用flex来实现响应式布局。

消息中间件Kafka，在分布式环境下Kafka的吞吐性能非常优秀，并且其持久化和订阅/发布的功能与物联网的场景非常匹配。

混合数据库（HBase、RDBMS Mysql、Redis）分为结构化和非结构化数据。

HBase适合于存储半结构化的数据，可以很好的支持海量物联网终端的历史数据的查询。

RDBMS适合于存储结构化的数据，通常根据具体的数据库采用不同的高可用部署方案，在系统中主要用来存储终端基础数据、字典数据和数据分析的结果等。

Redis是基于内存的KV数据库，在系统中通常用来缓存需要频繁更新和访问的数据。

数据处理：spark即可批处理也可流处理 （实时计算spark streaming/storm 离线计算MapReduce和Hive）DataxDataX 是一个异构数据源离线同步工具，致力于实现包括关系型数据库(MySQL、Oracle等)、HDFS、Hive、MaxCompute(原ODPS)、HBase、FTP等各种异构数据源之间稳定高效的数据同步功能。

设计的美观大方的产品原型如图 3所示。



图 3

设计文档在完成后，在项目组内进行评审，根据评审意见进行修改。

最终统一提交到文档版本库中。