ORB SLAM

数据结构

跟踪

局部地图

闭环检测

数据结构

- Frame
  - 计算出的姿态
  - 图像金字塔
  - ORB 特征点
    - 描述子
    - 与地图中 3d 点的对应关系（跟踪获得）
- KeyFrame
  - Frame 中的所有信息

数据结构

- MapPoint
  - 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]
    - 根据引用关键帧金字塔的尺度算出
    - Dmin = d / (1.2^(level +1)*1.5)
    - Dmax = d * (1.2^(maxlevel – level +1)*1.5)

数据结构

- cv::KeyPoint
  - 图像坐标
  - 尺度
  - 方向
  - 描述子
  - ...

数据结构

- Convisibility Graph
  - 节点
    - 代表关键帧
  - 边
    - 两个关键帧找到足够多的相同的 3d 点
    - 边权为 3d 点的个数
- Essential Graph
  - Covisibility Graph 的最大生成树

数据结构

- 视觉词袋
  - 词汇树
    - 叶子节点，单词， ORB 特征点
    - 父节点，由子节点的聚类获得
    - 需要预先计算得到
  - DBoW2 数据库
    - 查询地图中和当前关键帧相似的所有帧

Tracking

- V-A: 对于当前的图像，将其封装为 Frame（金字塔和 ORB 特征点）
- V-B: 初始估计
  - 利用速度模型和上一帧的姿态估计当前帧的姿态
  - 将上一帧图像的所有 3d 点利用估计的姿态投影在当前帧上，在投影点周围利用描述子寻找当前帧匹配的特征点
  - 利用 bundle adjustment 优化求解当前帧姿态 (g2o)，利用优化结果筛选当前帧特征点与上一帧 3d 点的匹配

Tracking

- V-D:Track Local Map
  - V-B 的步骤能计算出当前帧看到的一些 3d 点 (特征点与上一帧的匹配关系)
  - 在地图中找到看到这些 3d 点的关键帧集合 K1(其中看到最多点的关键帧记录为 Kref)
  - 在 Covisibility Graph 中找到和 K1 相邻的关键帧集合 K2
  - 利用 K1 和 K2 中所有的点和当前帧的特征点进行匹配，利用匹配的结果再次优化当前帧的姿态

•V-D:Track Local Map

- 匹配方法如下
- 对于 K1 和 K2 中的每个 3d 点 (V-B 步骤中已经确定与当前帧匹配的点)，投影到当前帧上，然后先进行如下检查
  - 检查投影坐标是否合法，计算 3d 点与当前帧的距离 d 和方向向量 v
  - 检查 v 与 3d 点平均观测方向的夹角，检查 d 是否在 3d 点观测距离范围内

•V-D:Track Local Map

- 检查合格后，进行匹配
  - 根据 d/dmin 预测该 3d 点在当前帧中的金字塔层数
  - 在投影点附近找到层数相近的特征点集合
  - 利用描述子距离在特征点集合中找到与该 3d 点的匹配 (找到的特征点必须是还没有进行 3d 点匹配)
  - 匹配结束后，利用 BA 对当前姿态进行优化，并根据优化结果对当前帧与 3d 点的匹配进行筛选

Tracking

- V-E: 检测新的关键帧
  - 距离上次 relocalization 已经至少 20 帧
  - 局部地图线程当前空闲，或者距离上次关键帧的插入已经有至少 20 帧
  - 当前的帧至少匹配到了 50 个 3d 点
  - 当前帧与 Kref(V-D 最开始的步骤中产生) 之间匹配的 3d 点不超过 90%

Local Mapping

- VI-A: 插入关键帧 K
  - 更新 Covisibility 图和 Enssential 图
  - 更新视觉词汇数据库 DBoW2
- VI-C(1): 产生新的 3d 点
  - 在 Covisibility 图中找出所有与 K 相邻的关键帧 {Kc}
  - 对 {Kc} 中的每一个关键帧 Kci
  - 利用视觉词汇树、特征点描述子和极线搜索找到 K 和 Kci 中未匹配特征点的匹配对 (搜索匹配方法见后文)
  - 检测匹配对产生的 3d 点是否合法 (检测方法见后文)，合法的 3d 点被封装为 MapPoint 插入到地图中

Local Mapping

- VI-C(2): 合并刚刚产生的 3d 点
  - 产生的 3d 点可能被 {Kc} 中的相邻关键帧看到，或者 K 与 Kci 和 K 与 kcj 之间产生的 3d 点有交集，同样的 3d 点只需要保留一个
- VI-C(3): 更新 Covisiblity 图、 Enssential 图和 MapPoint 类

Local Mapping

- VI-C(1) 搜索匹配方法
  - 找出 K 中所有未匹配的特征点 Sk
  - 找出 Kci 中所有为匹配的特征点 Skci
  - 在 Sk 与 Skci 中按照词汇树 w 级节点划分特征点，通过搜索在两个集合中相同 w 级节点对 (减少搜索时间)

Local Mapping

- VI-C(1) 搜索匹配方法
  - 每一对 w 级节点，对应 Sk 和 Skci 的子集
  - 在子集中搜索满足极线约束和描述子距离较小的特征点对，将这样的特征点对加入匹配结果中

Local Mapping

- VI-C(1): 检测方法
  - 对 K 和 Kci 每一个特征点对，利用两个关键帧的姿态和特征点的像素坐标计算 3d 坐标
  - 检查 3d 点是否都在两帧相机的前方 (z>0)
  - 检查 3d 坐标在两帧上的重投影误差
  - 检查特征点相对与两帧相机的距离与金字塔层数是否大致对应
  - ……

Loop Closing

- VII-A: 闭环检测
  - 对于当前的关键帧 K ，计算他与 {Kc} 中每一个关键帧相似度最小的值 Smin
  - 在 DBoW2 数据库中找出所有与 K 相似度不小与 Smin 的关键帧 {Ks} ， {Ks} 不能包含 {Kc}
  - 在 {Ks} 中找到合适的关键帧作为闭环检测的候选帧 (寻找方法见后文)

Loop Closing

- VII-A: 闭环检测——寻找候选帧
  - {Ks} 为通过数据库查询出来的相似帧集合
  - 上一次闭环检测的集合为 {Ks'} ，上上次的集合为 {Ks''}
  - 如果 {Ks} 中存在一个关键帧 Kp ， {Ks'} 中存在一个关键帧 Kp',{Ks''} 中存在一个关键帧 Kp'', 使得这三个关键帧在 Covisibility 图中非常相近（或者重合），那么就认为 Kp 是候选帧

Loop Closing

- VII-B 计算候选帧和当前关键帧的匹配
  - 计算候选帧和当前关键帧的 3d 点匹配用于产生闭环约束关系，与 VI-C 在局部地图中产生新 3d 点时采用的搜索方法类似，但是这里是进行 3d 与 3d 点的匹配，并且不考虑极线约束
  - 由于没有考虑极线约束，所以需要利用 ransac 方法来对匹配进行去噪
  - 如果候选帧和当前帧匹配的 3d 点较多，那么就认为候选帧真正通过了闭环的检测

Loop Closing

- VII-C 闭环融合
  - 候选帧通过闭环融合后，需要合并匹配的 3d 点，还要更新 covisibility 图和 enssential 图
  - 当前帧 K 的姿态和与 K 相邻关键帧的姿态需要利用产生的笔闭环进行纠正，相应的 3d 点的坐标也需要进行纠正
- VII-D Enssential 图姿态优化
  - 在 Enssential 图上进行姿态图优化