目录

[1. Tracking::Track() 1](#_Toc2919)

[1.1 StereoInitialization() 1](#_Toc10529)

[1.2 MonocularInitialization() 1](#_Toc29217)

[1.3 TrackReferenceKeyFrame（） 1](#_Toc22730)

[1.4 TrackWithMotionModel（） 2](#_Toc10047)

[1.4.1 SearchByProjection（） 2](#_Toc7748)

[1.5 Relocalization（） 3](#_Toc5369)

[1.5.1 DetectRelocalizationCandidates（mCurrentFrame） 3](#_Toc21695)

[1.5.2 SearchByBoW(pKF , mCurrentFrame , vvpMapPointMatches[i]) 4](#_Toc17457)

[1.6 TrackLocalMap() 4](#_Toc23102)

[2. LocalMapping::Run() 5](#_Toc22750)

[2.1 ProcessNewKeyFrame（） 5](#_Toc4309)

[2.2 SearchInNeighbors（） 6](#_Toc10966)

[2.2.1 ComputeDistinctiveDescriptors（） 6](#_Toc16081)

[3. LoopClosing::Run() 7](#_Toc9689)

[3.1 DetectLoop() 7](#_Toc27913)

[3.1.1 DetectLoopCandidates (mpCurrentKF，minScore) 7](#_Toc13069)

[3.1.2 7](#_Toc7414)

[3.2 ComputeSim3() 8](#_Toc16205)

[3.2.1 SearchBySim3(mpCurrentKF，pkF，vpMapPointMatches，s，R，t，7.5) 8](#_Toc15423)

[3.2.2 OptimizeSim3(mpCurrentKF,pKF,vpMapPointMatches,gScm,10,mbFixScale) 8](#_Toc19704)

[3.2.3 SearchByProjection （mpCurrentKF,mScw,mvpLoopMapPoints,mvpCurrentMatchedPoints,10） 9](#_Toc10706)

[3.3 CorrectLoop() 9](#_Toc13600)

[3.3.1 SearchAndFuse（CorrectedSim3） 10](#_Toc213)

[3.3.2 OptimizeEssentialGraph( ) 10](#_Toc6984)

[3.3.3 RunGlobalBundledAdjustment（nloopKF） 11](#_Toc16358)

ORB-SLAM2自带摄像头：https://blog.csdn.net/zhangqian\_shai/article/details/88406981

G2O1: <https://mp.weixin.qq.com/s/j9h9lT14jCu-VvEPHQhtBw>

G2O2（顶点）: <https://mp.weixin.qq.com/s/12V8iloLwVRPahE36OIPcw>

G2O3（边）: <https://mp.weixin.qq.com/s/etFYWaZ6y4XPiXrfqCm53Q>

G2O : <https://zhuanlan.zhihu.com/p/36889150>

# Tracking::Track()

1. 地图初始化

## StereoInitialization()

1. 特征点数大于500，设置初始位姿为单位矩阵，将当前帧（第一帧）构建为初始关键帧，并将其添加进局部地图中。
2. 将特征点构造地图点，通过反投影变换获取该特征点在世界坐标系下的3D坐标。

## MonocularInitialization()

1. 创建单目初始器，特征点数目大于100,。

用当前帧（mCurrentFrame）更新上一帧（mLastFrame），将上一帧（mLastFrame） 的特征点保存在mvbPrevMatched中。

1. 进行ORB特征点匹配(SearchForInitialization)，匹配关系保存在mvIniMatches中，返回匹配点数nmatches（要求要大于100）。
2. 通过H或F模型进行初始化。得到两帧之间的相对运动（Rcw，tcw），匹配关系存储在mvIniMatches中，三角化后的空间点集合保存在mvIniP3D中。
3. 将初始化第一帧作为世界坐标系（单位矩阵），求出当前帧（mCurrentFrame）的变换矩阵Tcw。
4. 将三角化后的3D点包装成地图点存入KeyFrame和Map中。
5. 进入SLAM模式
6. 检查并更新上一帧被替换的地图点。
7. 如果运动模型（mVelocity）为空或当前帧紧靠上一帧，则用参考关键帧跟踪

## 1.3 TrackReferenceKeyFrame（）

1. 计算当前帧的词袋模型
2. 通过词袋加速当前帧（mCurrentFrame）与参考帧（mpReferenceKF）之间的特征点匹配，将匹配关系存储在vpMapPointMatches中。返回匹配点数目（nmatches）。nmatches小于15,认为跟踪失败。
3. 用上一帧的位姿（mLastFrame）作为当前帧（mCurrentFrame）的位姿初始值。
4. 对当前帧（mCurrentFrame）的位姿进行BA优化（PoseOptimization（））。
5. 对优化后的匹配点剔除外点（3D），统计匹配的内点（3D）数目（nmatchesMap++），nmatchesMap大于10，则认为跟踪成功。
6. 如果不满足B中的条件，则进行恒速模型跟踪

## 1.4 TrackWithMotionModel（）

1. 更新上一帧的位姿（UpdateLastFrame（））

* 获取上一帧（mLastFrame）的参考关键帧（pRef），计算出参考关键帧到上一帧的位姿变换（Tlr）,计算上一帧在世界坐标系下的位姿。
* 对于双目或者RGB-D。获取上一帧（mLastFrame）有效的特征点（深度大于0），将他的深度值和ID存入vDepthIdx中。并从小到大排序。
* 遍历vDepthIdx，判断其每个ID所对相应的特征点是否在上一帧中的地图点中，或者并没有被观测到，便新增地图点（bCreateNew = true）
* 新增地图点（pNewMP），将其添加进上一帧（mLastFrame）的地图点（mvpMapPoints）中。

1. 根据恒速模型（mVelocity）得到当前帧（mCurrentFrame）的位姿
2. 进行投影匹配

### 1.4.1 SearchByProjection（）

SearchByProjection（mCurrentFrame , mLastFrame , th , mSensor== System::MONOCULAR）

* 建立旋转直方图

源代码：const float factor = 1.0f / HISTO\_LENGTH 改为：

const float factor = HISTO\_LENGTH / 360.0f

* 获取当前帧相对于上一帧的平移向量tlc，判断当前帧是前进还是后退（bForward , bBackward）。
* 将上一帧的地图点投影到当前帧，获得其像素坐标

1. 获取上一帧（mLastFrame）的地图点（pMP）在世界坐标系下的坐标（x3Dw），然后再经过当前帧的旋转和平移（Rcw , tcw）计算出其在当前帧相机坐标系下的坐标（x3Dc）。再将其投影，获得其像素坐标u , v。
2. 计算上一帧（mLastFrame）地图点对应的二维特征点在金字塔的层级（nLastOctave），设置搜索半径radius。

* 获取当前帧搜索范围内的所有特征点（2D）ID（vIndices2）

GetFeaturesInArea（u，v，radius，nLastOctave）

获取上一帧（mLastFrame）的地图点（pMP）的描述子dMP

1. 遍历vIndices2中的特征点所对应的地图点，如果存在，则退出循环。
2. 对于双目或RGB-D相机，计算出当前特征点在右图中的像素坐标（ur），计算误差er，误差大于radius，退出。
3. 计算当前帧和上一帧之间的描述子（d , dMP）距离，获得最小描述子距离（bestDist）及其对应的ID（bestIdx2）。
4. 当bestDist小于设定的阈值（TH\_HIGH），将上一帧中最佳ID（bestIdx2）对应的地图点（pMP）保存进当前帧的地图点中，匹配数量（nmatches）加1。

* 计算匹配点旋转角度差坐在的直方图
* 进行旋转一致性检查，删除不一致的匹配点，最终返回nmatches。

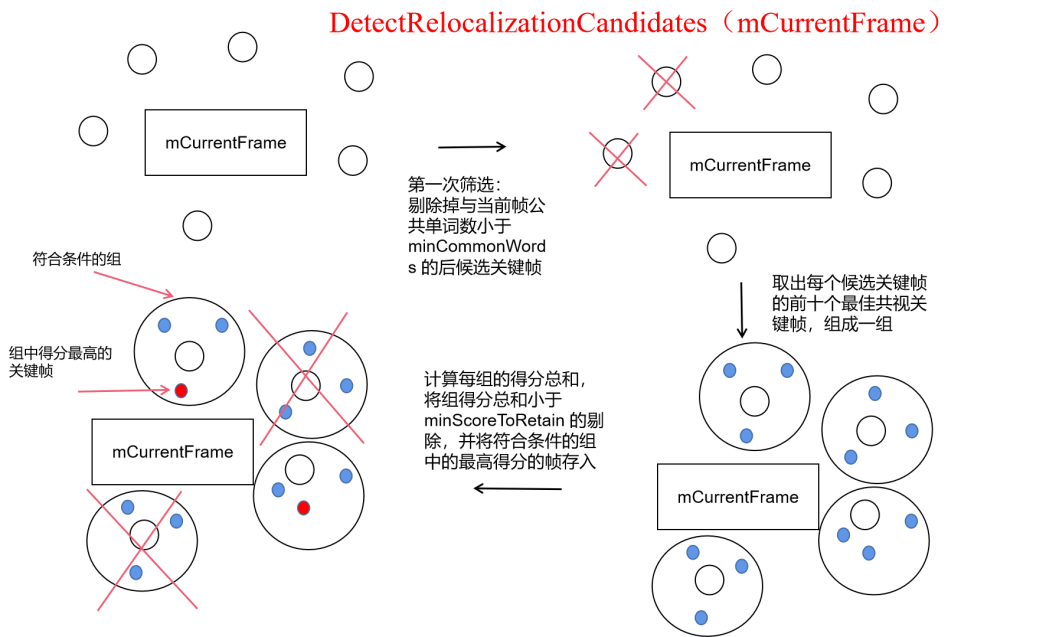
1. 对当前帧进行BA优化PoseOptimization()。
2. 对优化后的匹配点剔除外点（3D），统计匹配的内点（3D）数目（nmatchesMap++），nmatchesMap大于10，则认为跟踪成功。（d，e步和TrackReferenceKeyFrame（）中的一样）。
3. 如果上述跟踪不成功，进行重定位跟踪

## 1.5 Relocalization（）

1. 计算当前帧的词袋
2. 用词袋找到与当前帧相似的候选关键帧，返回vpCandidateKFs

### 1.5.1 DetectRelocalizationCandidates（mCurrentFrame）

* 找出与当前帧具有共同wordId的所有关键帧（lkFs），遍历lkFs，对没有标记的关键帧（pKFi）进行重定位候选帧标记（mnRelocQuery）,将其（pKFi）存入lkFsSharingWords中，重定位单词数（mnRelocWords）加1。
* 遍历lkFsSharingWords，统计关键帧组（lkFs）中与当前帧（mCurrentFrame）具有的最多公共单词的数量（maxCommonWords），设定最小单词数minCommonWords = maxCommonWords \* 0.8 。
* 再次遍历lkFsSharingWords，取出所有候选关键帧（pKFi），对满足公共单词数满足minCommonWords 要求的候选关键帧，利用词袋模型计算与当前帧（mCurrentFrame）的得分si，并将得分和符合条件的候选关键帧（si，pKFi）共同存入lScoreA
* ndMatch中。
* 遍历lScoreAndMatch，取出其中的关键帧pKFi，获取它pKFi的前十个最佳共视关键帧（vpNeighs），遍历vpNeighs，将符合条件的关键帧得分全部总和存入accScore中，统计得到该组中得分最高的关键帧存入pBestKF，将最高分数存入bestScore中。最后将总分和最佳关键帧（accScore，pBestKF）存入lAccScoreAndMatch中。统计各组中总分最高的组得分（bestAccScore）。
* 设置阈值minScoreToRetain = 0.75 \* bestAccScore，遍历lAccScoreAndMatch，取出总得分si和总得分大于阈值minScoreToRetain的组中最佳关键帧pKFi，将pKFi存入vpRelocCandidates中。



1. 遍历vpCandidateKFs，将候选关键帧（pKF）和当前帧进行词袋匹配

### 1.5.2 SearchByBoW(pKF , mCurrentFrame , vvpMapPointMatches[i])

* 获取关键帧（pKF）的地图点vpMapPointsKF，取出该关键帧的词袋特征向量vFeatVecKF，构建旋转直方图。
* 取出词袋中该节点下的特征点索引，vIndicesKF（pKF的）和vIndicesF（mCurrentFrame的）。
* 遍历vIndicesKF，取出索引所对应的特征点的地图点（pMP），计算该特征点的描述子（dKF）。
* 遍历vIndicesF，计算相应的特征点的描述子（dF），计算dKF和dF的距离dist。记录最佳距离和次佳距离bestDist1,bestDist2。
* 进行匹配地图点点筛选，将合格的存入vpMapPointMatches。
* 删除误匹配的点，返回匹配成功的点数（nmatches）。
* 进行EPnP计算位姿，返回Tcw
* Iterate（5，bNoMore , vbInliers , nInliners）
* 对位姿Tcw进行BA优化，返回内点个数nGood
* 如果内点个数小于50，进行投影匹配（SearchByProjection见1.4.1）生成新的匹配
* 。。。。。。。。。。。。。。。。。。。。。

1. 仅定位模式。。。。。。。。。。。。。。。。。。。。
2. 局部地图跟踪，更新mState

## 1.6 TrackLocalMap()

1. 更新局部地图（更新局部关键帧和局部地图点）
2. SearchLocalPoints（）

* 遍历当前关键帧（mCurrentFrame ）的地图点pMP，更新观测，进行标记，在后面的搜索匹配中不投影
* 遍历局部地图点，判断该地图点是否在当前帧的观测范围内，nToMatch加1。
* nToMatch大于0，进行投影匹配SearchByProjection()（见1.4.1），获得更多的匹配点。

1. BA优化位姿
2. 更新当前还帧的地图点被观测程度，并统计跟踪到的地图点数目（mnMatchesInliers）。
3. 根据跟踪匹配数和重定位情况判断是否跟踪成功
4. 更新显示线程中的图像、特征点、地图点等信息。
5. 如果跟踪成功，更新很俗运动模型mVelocity。
6. 清空观测不到的点
7. 删除临时地图点mlpTemporalPoints
8. 插入关键帧CreateNewKeyFrame（）
9. 删除外点
10. 如果跟踪失败，重置系统
11. 记录位姿Tcr（由参考关键帧到当前帧的位姿变换），将Tcr存入 mlRelativeFramePoses中。
12. Tracking线程结束

# 2. LocalMapping::Run()

1. 不接受关键帧
2. 处理列表中的关键帧

## ProcessNewKeyFrame（）

1. 取出关键帧队列（mlNewKeyFrame，追踪线程向局部地图线程插入的关键帧队列）中第一帧作为当前关键帧（mpCurrentKeyFrame），然后将取出的关键帧从队列删除。
2. 计算当前关键帧（mpCurrentKeyFrame）的词袋向量。
3. 取出当前关键帧（mpCurrentKeyFrame）的所有地图点存入vpMapPointMatches中。遍历vpMapPointMatches，取出地图点pMP，如果地图点不在当前帧的观测中，则为添加观测，更新平均观测方向和观测距离。否则，将pMP存入mlpRecentAddedMapPoints中。
4. 更新当前关键帧（mpCurrentKeyFrame）的连接关系
5. 将该关键帧插入地图中
6. 删除质量不好的地图点MapPointCulling（）
7. 当前关键帧和相邻关键帧三角化产生新的地图点

CreateNewMapPoints（）

1. 检查并融合当前关键帧和相邻关键帧中重复的地图点

## SearchInNeighbors（）

1. 获取与当前关键帧共识程度最高的前nn（单目20，其他10）帧，存入vpNeighKFs中。

* 遍历vpNeighKFs，取出其中的关键帧pKFi（一级相连关键帧）存入vpTargetKFs 中。
* 获取与一级相连关键帧pKFi相连的共识程度最高的前五帧二级相连关键帧存入vpSecondNeighKFs中。
* 遍历vpSecondNeighKFs，取出其中的关键帧pKFi2，将其也存入vpTargetKFs中。
* 获取当前关键帧的地图点存入vpMapPointMatches中，
* 遍历vpTargetKFs，取出其中的关键帧pKFi。将vpMapPointMatches匹配到关键帧中进行融合。Fuse（pKFi，vpMapPointMatches），大致步骤可参考1.4.1SearchByProjection（）。

1. 遍历vpTargetKFs，取出其中的关键帧pKFi，获取pKFi的地图点存入vpMapPointsKFi中，遍历其中（vpMapPointsKFi）的地图点pMP，将其存入存储融合点的向量中（vpFuseCandidates）。然后再次进行融合。
2. 重新获取当前关键帧的地图点存入vpMapPointMatches中，遍历vpMapPointMatches中的地图点pMP，找出其中（pMP）的最佳描述子，更新pingju8n观测方向和观测距离。

### 2.2.1 ComputeDistinctiveDescriptors（）

1. 获取该地图点所有有效的观测关键帧信息，存入observations中
2. 遍历observations，取出其对应的关键帧pKF，同时将其对应的描述子向量存入vDescriptors中
3. 计算两两描述子之间的距离，存入Distances中。
4. 求第i个描述子与其他所有描述子之间的距离，并从小到大进行排序，取出其中的中值median。通过循环求出所有描述子之间的最小中值BestMedian和其对应的索引BestIdx。将最小中值描述子对应的索引存入mDescriptor中
5. 更新当前帧和其他帧的连接关系。
6. 局部BA，LocalBundleAdjustment（mpCurrentKeyFrame，mbAbortBA，mpMap）， 优化了当前关键帧的位姿和LocalKeyFrames能观测到的所有地图点的位置
7. 删除冗余关键帧，90%的地图点能被其他关键帧观察到，则判定为冗余关键帧 KeyFrameCulling（）。
8. 将当前关键帧插入闭环检测队里（mpLoopCloser）
9. 结束线程

# LoopClosing::Run()

## 3.1 DetectLoop()

1. 提取关键帧队列中第一帧，赋值为当前帧（mpCurrentKF），然后从队列删除
2. 检测距离上次闭环时间
3. 找出当前帧的连接关键帧，并计算出于当前关键帧相似度得分最低的连接关键帧作 为最低分阈值（minScore）
4. 在所有的关键帧中找出闭环候选帧组vpCandidateKFs（闭环候选帧不与当前帧连接）

### 3.1.1 DetectLoopCandidates (mpCurrentKF，minScore)

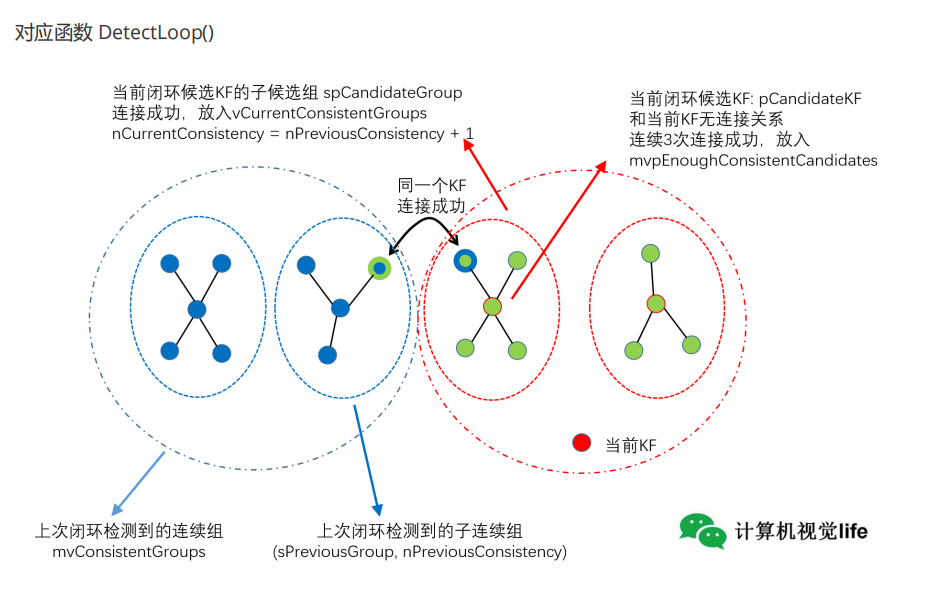
取出与当前关键帧相连的所有关键帧，存储到spConnectedKeyFrames中

1. 找出和当前关键帧具有公共单词的所有关键帧（存储到lkFsSharingWords），不包括与当前帧相连的关键帧,并统计当前帧和闭环候选帧具有相同的words个数
2. 统计闭环候选关键帧中与当前帧具有公共单词最多的单词数（maxCommonWoeds），让最小单词数（minCommonWoeds）等于0.8倍最大单词数
3. 遍历上述的候选关键帧，挑选出共有单词数大于minCommonWoeds，且匹配得分大于minScore的候选关键帧，并将它存入lScoreAndMatch（符合条件的候选关键帧）
4. 取出与lScoreAndMatch相连的10个关键帧，存入vpNeights中

计算最佳得分（bestScore）和累计得分（accScore）并将得分最高的帧赋给 pBestKF，再将accScore和pBestKF存入lAccScoreAndMatch,将所有租的最佳 累计得分保存在bestAccScore，minScoreToRetain = 0.75 \* bestAccScore

1. 将候选帧组总得分大于minScoreToRetain的组中的最高得分的帧放入vpLoopCandidates中

### 3.1.2



1. 遍历3.1.1中得到的vpCandidateKFs,取出与其相连的关键帧构成一个子候选组（spCandidateGroup(包括自己)）。
2. 获取上一次回环检测的子连续组（sPreviousGroup）,检测sPreviousGroup，spCandidateGroup是否有相同的关键帧。
3. 如果有相同的关键帧，则表示两个自候选组相连，记录连续次数的nCurrentConsistency + 1，如果nCurrentConsistency >= 3，达到闭环条件，就将此候选关键帧放入mvpEnoughConsistentCandidates中，同时将此组的相应信息保存进上次闭环检测到的连续组（mvConsistentGroups）中

## 3.2 ComputeSim3()

1. 遍历候选关键帧集mvpEnoughConsistentCandidates，初步筛选出与当前关键帧匹配 点数大于20的候选关键帧，并 为每一个候选帧构建一个Sim3Solver，并将他们存 储在vpSim3Solver向量组中，记录 保留下来的关键帧数量（nCandidates）
2. 对每个候选帧使用Sim3Solver迭代匹配，如果计算出Sim3变换，进行优化匹配

取出Sim3Solver匹配后的内点集合，保存在vpMapPointMatches中

求解出候选帧到当前帧的R、t、s

### 3.2.1 SearchBySim3(mpCurrentKF，pkF，vpMapPointMatches，s，R，t，7.5)

1. 计算内参，sim3的逆，以及关键帧的地图点（vpMapPoints1,vpMapPoints2）
2. 记录已经匹配的地图点（向量vbAlreadyMatched1,vbAlreadyMatched2）
3. 将pkF1的地图点投影到pkF2中的相机坐标系下坐标p3Dc2，然后再转换为像素坐标
4. 预测投影点在金字塔的层数，并计算搜索半径
5. 获取该区域内的所有候选匹配特征点（vIndices），计算描述子距离，找出最佳匹配点。保存在vnMatch1
6. 后续同上，寻找pkF2中的特征点在pkF1中的最佳匹配vnMatch2
7. 一致性检查，只有当lpkF1和pkF2中的两次投影的点是同一个点时才算匹配成功，将最后匹配成功的点数保存在nFound中，并返回

### 3.2.2 OptimizeSim3(mpCurrentKF,pKF,vpMapPointMatches,gScm,10,mbFixScale)

1. g2o配置操作
2. 设置顶点，配置相机内参，添加顶点
3. 添加pkF1和pkF2的匹配点作为顶点（但是不优化（->setFixed(true)））
4. 添加边（正向投影，从比黄候选关键帧到当前关键帧的投影）

同理，添加边（反向投影）

1. 迭代优化
2. 用卡方检验删除误差大的边
3. 将优化后的结果赋值给g2oS12（其实就是gScm），返回匹配的内点个数（nInliers）。

如果内点数（nInliers）大于等于20，将闭环候选关键帧存入mpMathedKF中， 将vpMapPointMatches存入mvpCurrentMatchedPoints中

1. 取出与候选关键帧相连的共视关键帧，将他们一起存入vpLoopConnectedKFs中，

遍历vpLoopConnectedKFs中的所有帧的地图点，将有效的地图点存入 mvpLoopMapPoints中，将当前帧（mpCurrentKF）的Id赋值给mnLoopPointForKF

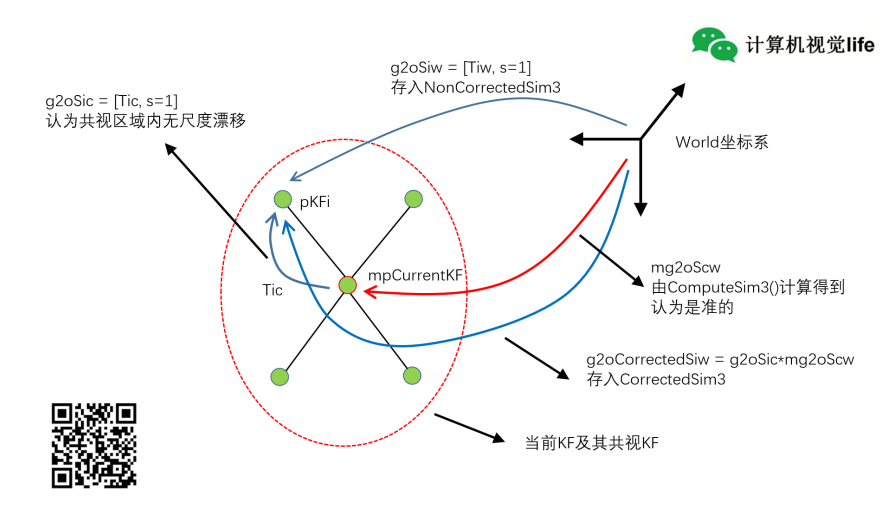
1. 将闭环候选关键帧及其连接关键帧的所有地图点投影到当前关键帧中，进行投影匹 配。

### 3.2.3 SearchByProjection （mpCurrentKF,mScw,mvpLoopMapPoints,mvpCurrentMatchedPoints,10）

投影匹配，参考3.2.1SearchBySim3，返回新的匹配成功的点的个数。并将新的匹 配成功的添加进mvpCurrentMatchedPoints中。

1. 计算匹配成功的地图点个数（nTotalMatches）,如果nTotalMatches大于等于40，保 留当前闭环候选帧，删掉其他的闭环候选关键帧，返回true。否则，闭环不可靠， 删除所有闭环候选帧，返回false。

## 3.3 CorrectLoop()



1. 更新当前帧的连接关系
2. 选出当前关键帧的连接关键帧，并将他们一起存入mvpCurrentConnectedKFs中

遍历mvpCurrentConnectedKFs，获取当前帧（mpCurrentKF）到连接帧（pKFi） 的位姿变换（Tic，Ric，tic）。

获取当前帧到其共识关键帧的Sim3变换，g2oCorrectedSiw（认为是准确的）， 并将g2oCorrectedSiw存入CorrectedSim3中。g2oSiw（认为是不准确的）， 并将g2oSiw存入NonCorrectedSim3中。

1. 遍历CorrectedSim3，取出当前关键帧的连接关键帧（pKFi）以及其Sim3变换（g2oCurrectedSiw）。同时取出pKFi对应NonCorrectedSim3的Sim3变换（g2oSiw）。

获取pKFi的对应地图点（vpMPsi）,遍历vpMPsi，获取其世界坐标（P3Dw）、 并将其转换为Eigen形式（eigP3Dw）.

对于eigP3Dw，首先通过g2oSiw获取其在pKFi下的坐标。然后再通过 g2oCorrectedSiw再次获取其在世界坐标系下的坐标（eigCorrectedP3Dw）。

记录矫正该地图点的关键帧（mpCurrentKF）Id，以及当前地图点所在的连接 关键帧（pKFi）的Id。更新平均观测。

1. 获取g2oCorrectedSiw的R，t，求出变换矩阵correctedTiw，更新pKFi矫正后的位姿，更新连接关系
2. 获取mvpCurrentMatchedPoints（投影匹配成功）中的地图点pLoopMP，以及在同 样的索引下当前关键帧mpCurrentKF对应的地图点pCurMP。如果pCurMP存在， 就用pLoopMP将其替换，不存在的话，就将pLoopMP添加进去

### 3.3.1 SearchAndFuse（CorrectedSim3）

1. 遍历CorrectedPosesMap（即CorrectedSim3），取出关键帧pKF（矫正后的与当前关键帧相连的关键帧），sim3变换g2oScw、cvScw。

#### Fuse（pKF，cvScw，mvpLoopMapPoints,4，vpReplacePoints）

1. 取出Scw(即cvScw)的旋转矩阵，尺度，计算得到Rcw，tcw，Ow。

取出pKF的地图点，存入spAlreadyFound中，获取已经闭环匹配上的地 图点（mvpLoopMapPoints）数量nPoints，

1. 遍历所有地图点，获取地图点在相机坐标系下的坐标p3Dc。
2. 获取地图点（闭环候选关键帧的地图点）投影在pKF（当前关键帧的相连关键帧）上的像素坐标
3. 投影匹配操作同3.2.1 SearchBySim3
4. 替换或新增地图点，记录需要替换的地图点存入vpReplacePoints，返回替换或新增的地图点数目nFused。
5. 替换需要替换的地图点。（替换是用mvpLoopMapPoints中的地图点替换pKF中的地图点）
6. 遍历当前关键帧相连的关键帧组（mvpCurrentConnectedKFs）,得到与当前关键帧相连关键帧的相连关键帧（当前关键帧的二级相连关键帧），存入vpPreviousNeighbors。
7. 更新一级相连关键帧（pKFi）的连接关系，将连接关系存入LoopConnections中。
8. 首先，将LoopConnections中的二级连接关系vpPreviousNeighbors删除。然后，再将LoopConnections中的原来的当前关键帧连接关系（mvpCurrentConnectedKFs）删除掉。

⑥

### 3.3.2 OptimizeEssentialGraph( )

OptimizeEssentialGraph( mpMap ,mpMatchedKF, mpCurrentKF , NonCorrectedSim3, CorrectedSim3 , LoopConnections，mbFixScale )

1. 构造优化器
2. 添加顶点，优先使用调整后的Sim3，否则使用追踪时的位姿。对闭环候选关键帧不进行优化
3. 添加第一种边，闭环时因为地图点调整而出现的关键帧间的新连接关系
4. 添加第二种边，跟踪时形成的边，即生成树的边（有父关键帧）

添加第三种边，闭环匹配形成的边，当前帧和闭环匹配帧之间的连接关系

添加第四种边，共视程度超过100的关键帧也作为边进行优化

1. 开始g2o优化
2. 将优化后的位姿更新进关键帧中
3. E，F得到优化后的位姿后，地图点根据参考帧优化前后的相对关系调整自己的位置。

⑦ 添加当前帧和闭环匹配帧之间的边

⑧ 进行全局BA

### 3.3.3 RunGlobalBundledAdjustment（nloopKF）

A.

GlobalBundledAdjustment（mpMap , 10 , mbStopGBA , nloopKF , false）

#### BundledAdjustment（vpKFs , vpMP , nIterations , pbStopFlag , nLoopKF , bRobust）

1. 初始化g2o优化器
2. 向优化器中添加顶点

添加关键帧位姿作为优化器顶点

添加地图点作为顶点（当顶点为三维点时，需要进行边缘化，即： setMarginalized（true））

1. 向优化器中添加投影边。。。。。
2. 获取优化后的结果
3. 遍历并更新全局地图中的所有spanning tree中的关键帧
4. 遍历每一个地图点，并用更新的关键帧位姿来更新地图点位置