ORBSLAM3

bitterengsci

Outline

- History
- Architecture
- System Overview
- Tracker
- LocalMapper
- LoopCloser
- Initialization
- Relocalization

History

- 2011 ORB feature "ORB: An efficient alternative to SIFT or SURF"
- 2012 "Bags of Binary Words for Fast Place Recognition in Image Sequences"
- 2015 ORB-SLAM (Monocular) "ORB-SLAM: A Versatile and Accurate Monocular SLAM System"
- 2017 ORB-SLAM2 (Monocular + Stereo + RGBD) "an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras"
- 2017 ORB-SLAM-VI (Mono + Inertial) "Visual-inertial monocular SLAM with map reuse"
- 2019 ORB-SLAM-Atlas (submap & map merge)"a robust and accurate multi-map system"
- 2020 IMU-Initialization "Inertial-Only Optimization for Visual-Inertial Initialization"
- Now ORB-SLAM3 ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM

Trend: 各种途径利用更多信息(multi-sensor for estimation/initialization, map reuse)

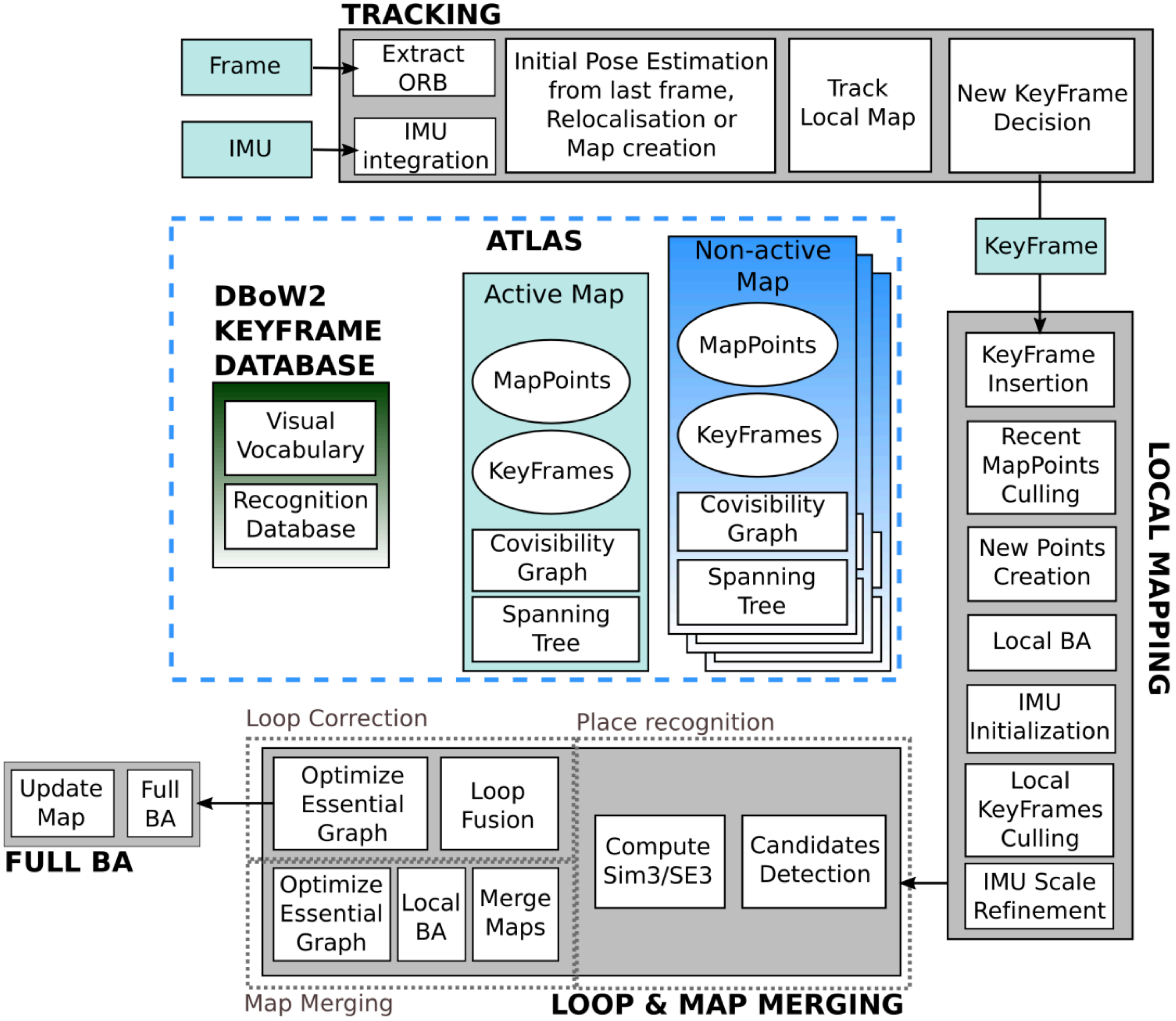
→ more and easier to be generalized for different configurations,

→ more accurate & robust, longer period survival

Novelty

- Mono/Stereo (-inertial) SLAM fully relying on Maximum-a-Posteriori estimation
- High-recall place recognition
  - geometry consistency check first, then local consistency with three covisible keyframes
  - in contrast to DBoW2 bag of words lib, temporal consistency(matching three consecutive keyframes to the same area), then geometric consistency
  - recall++ (more data association & map accuracy), computation cost++ 😢
- Multi-map SLAM
  - a set of disconnected maps
  - map operations: place recognition, camera relocalization, loop closure, map merge
- Abstract camera model
  - new model can be applied to this system provided its projection, unprojection and Jacobian
  - implemented pinhole, fisheye

System Architecture



System Overview

./Examples/ (Monocular/Monocular-Inertial/Stereo/Stereo-Inerial/RGB-D/ROS) / (EuRoC/TUM/KITTI)

- It can deal with multiple sequences (System::ChangeDataset) in one run
- LoadImages && LoadIMU
- Initialize a **System** object
- Run in image frame temporally for(int ni=0; ni<nImages[seq]; ni++, proccIm++)
    - (image preprocessing: undistort EuRoC, CLAHE TUM)
    - IMU: stack temporally corresponding measurements into a vector
    - System::Track📷
    - Wait function
- Change dataset…
- after all done, stop all threads with System::Shutdown,
- Output trajectories System::SaveTrajectory🤖 & System::SaveKeyFrameTrajectory🤖

📷 camera type, i.e. Monocular, Stereo, RBGD 🤖 dataset/format name

System Overview

- **System.cc** main thread
  - Construction:
    - Load ORB Vocabulary, Create KeyFrame Database, Create the Atlas(map manager)
    - Tracker
    - thread LocalMapper
    - thread LoopCloser
    - thread Viewer (including FrameDrawer, MapDrawer)
  - System::Track📷
    - Check Localization Mode
      - True → LocalMapper stopped, Tracker onlyTracking
    - Check reset (Reset, ResetActiveMap)
    - (Tracking::GrabImuData)
    - Tracking::GrabImage📷
      - Convert images to grayscale,
      - Create Frame (CurrentFrame)　在此处extract ORB, put keypoints into grid
      - Tracking::Track()　main function for tracking
    - Update Tracker state, Mappoints, Keypoints

Class Frame

- Construction:
  - Set frame id & scale level info
  - Extract ORB features
  - 若能提取到KeyPoints
    - UndistortKeyPoints
    - 处理第一个Frame的时候，需要计算一下Image的Bound
    - AssignFeaturesToGrid
      - 先根据FRAME_GRID_COLS和FRAME_GRID_ROWS来创建Grid (和GridRight)
      - 在grid中对应位置存入KeyPoint的序号
    - 若前一个Frame的信息被传入此constructor，则拷贝前一帧的Velocity到此帧；否则 Velocity set 0
- Methods:
  - ComputeBoW(), Set/GetPose(), SetVelocity(), etc.

Tracking State

```
enum eTrackingState{      //枚举类型

    SYSTEM_NOT_READY = -1,      // 未尝一见 🈚

    NO_IMAGES_YET = 0,          // initialized/reset to be 👶
    NOT_INITIALIZED = 1,

    OK = 2,                     // 皆大欢喜 ☀️

    RECENTLY_LOST = 3,          // ORB-SLAM3 新增, 短时间内丢失; 期待通过IMU & 重定位可以recover

    LOST = 4,                   // 潸然泪下 🔥

    OK_KLT = 5                  // 未尝一见 🈚

    };
```

System::ResetMap V.S. System::ResetActiveMap

- 每次System::Track📷 的时候会检查是否需要Reset
- ResetMap
    - 结果: Tracking::ResetMap
        - Viewer stopped, LocalMapper::Reset, LoopCloser::Reset, clear BoW database
        - Atlas clear map (mappoints & keyframes) & create new map
        - Frame & KeyFrame Id set to 0, State to 0 (NO_IMAGE_YET), Initializer to NULL, 所有tracking存储的estimated keyframes poses/referenced frame poses/times信息都清除
        - Tracking::ResetActiveMap
- ResetActiveMap
    - 结果: Tracking::ResetActiveMap
        - Viewer stopped, LocalMapper ResetActiveMap, LoopCloser ResetActiveMap, clear BoW database
        - Atlas clear map (mappoints & keyframes)
        - LastInitFrameId & LastRelocFrameId set to Frame::NextId, State to 0 (NO_IMAGE_YET), Initializer to NULL, 检查哪些frames lost了
    - 什么时候会ResetActiveMap

LocalMapper::Reset/ResetActiveMap, LoopCloser::Reset/ResetActiveMap

- LocalMapper::Reset/ResetActiveMap
    - 结果:
        - clear NewKeyFrames,
        - clear RecentAddedMapPoints
        - reset inertial parameters, (Reset only) Index Init to 0
- LoopCloser::Reset
    - 结果:
        - clear LoopKeyFrameQueue
        - LastLoopKFid to 0
- LoopCloser::ResetActiveMap
    - 结果:
        - loop through LoopKeyFrameQueue, 若KeyFrame对应的Map是需要reset的这个map, 则从 LoopKeyFrameQueue 擦除这个KeyFrame

Tracking::Track() 1

- StepByStep (GUI clickbox控制, 一图像帧一图像帧地处理dataset)
- 若LocalMapper::BadIMU, System::ResetActiveMap
- State != NO_IMAGE_YET, 检查时间戳是否合理
    - 若 LastFrame.Time > CurrentFrame, 清空 IMU data, 在atlas中建立map　RETURN
    - 若 LastFrame.Time > CurrentFrame + 1 （检测到时间跳跃）
        - 若系统中使用IMU且IMU初始化成功, 在atlas中建立map; 否则, System::ResetActiveMap
        - RETURN
- 若系统中使用IMU, 给CurrentFrame赋予上一个KeyFrame的IMU Bias
- 若State 为 NO_IMAGE_YET, 切换为 NOT_INITIALIZED
- 若使用IMU, 预积分
- 关于Map Index的一些操作
- 若State 为 NOT_INITIALIZED, 进行MonocularInitialization() 或者 StereoInitialization(); 若Initialize成功, State 变为 OK　☞LocalMapper::InsertKeyFrame
    - 若初始化不成功 (State不为OK), 则将CurrentFrame拷贝至LastFrame　RETURN
    - 若现在Atlas中只有一个地图, 则FirstFrameId 设置成当前Frame Id
- 若State 为 NOT_INITIALIZED 以后的状态, 说明已经初始化完成了 (TBD)

Tracking::Track() 2 CONTINUED

- 若State 为 NOT_INITIALIZED 以后的状态, 说明已经初始化完成了 (TBD)

  相机位姿变化的初步估计, 开始跟踪 (!OnlyTracking mode)

  - State 为OK
    - 检查一下MapPoints, 因为有一些MapPoint在LocalMapping中会被改变
    - 若速度为空, IMU没有初始化, 当前帧距离上次初始化的帧小于2, 则TrackReferenceKeyFrame(); 否则, TrackWithMotionModel(). 另外, 如果TWMM不成功, 则再尝试TRKF
    - 若TRFK和TWMM都没有成功
      - 若使用IMU且当前帧距离上次初始化的帧小于一个可以重置IMU的阈值的时候, State设置为LOST
      - 若当前地图的KeyFrame大于10, 设置为Recently_Lost
      - 否则, 设置为LOST
  - State 为Recently Lost
    - 若系统有IMU且当前地图IMU初始化了, 则利用IMU来预测State
      - 若丢失的时间超过阈值 time_recently_lost, 则State变为LOST
    - 若系统没有IMU, 重定位; 重定位不成功的话, 则State变为LOST (未修改完成的代码)
  - State 为Lost
    - 若当前的地图中Keyframe数量小于10, 则 System::ResetActiveMap (认为当前地图信息很少, 不值得留下); 否则建立一个新的地图 RETURN

Tracking::Track() 3 CONTINUED

- 跟踪局部地图 (!OnlyTracking mode)
    - 若局部变量bOK, TrackLocalMap()
- 若bOK, 皆大欢喜, 一切顺利! State 😊 OK
- 若!bOK
    - 当前State OK,
        - 若使用IMU且IMU初始化完成, BA成功, 则System::ResetActiveMap, State 设置为 Recently_Lost; 否则State变为Lost
        - 若当前帧距离上次重定位的帧超过一定阈值, 则将TimeStampLost这个变量置为当前时间戳
- bOK 或者 当前State Recently_Lost
    - 若当前帧和上一帧都有位姿估计, 则更新motion model (velocity), 否则velocity置为空
    - 清空Visual-Odometry的匹配; 删除temporal的Mappoints
    - 检查是否需要新的关键帧 NeedNewKeyFrame(), 若需要则增加 ☞LocalMapper::InsertKeyFrame
    - 移除那些当前帧Mappoints的outlier

Tracking::Track() 4 CONTINUED

- 若State为LOST
    - 当前地图的关键帧小于5，System::ResetActiveMap RETURN
    - 若系统使用IMU且IMU未初始化，System::ResetActiveMap RETURN
    - 上面两个情况均不成立时，在atlas中建立新地图
- 位姿记录
    - 若State为OK或者Recently Lost，则将当前帧的位姿存入System对应的vectors里
        - 这些信息会用在所有跟踪建图回环优化结束后，输出相机完整的轨迹
        - 包括：当前Frame相对他的keyframe的位姿，指向它所参照的KeyFrame的指针，Frame的时间，以及State是否为LOST
        - 若当前Frame的位姿为空，那么当前Frame相对他的keyframe的位姿存当前帧的前一帧的相对位姿

SideNote: TrackWithMotionModel V.S. TrackReferenceKeyFrame

- Tracking::TrackWithMotionModel
  - 更新前一帧的pose
  - 若IMU已经初始化完成，但当前帧距离上次重定位帧超出一定阈值，则重新使用IMU预测，RETURN TRUE
  - 使用速度和前一帧，计算当前帧的位姿
  - 当前帧和前一帧匹配 ((ORBMatch::SearchByProjection), 匹配数量少于20时，提高阈值 (x2) 再进行一次匹配 (wider search)
  - Wider search还是少于20的话，如果系统使用IMU, RETURN TRUE; 否则 FALSE
  - 利用所有匹配优化当前帧位姿
  - 去外点；去外点后，若系统使用IMU, RETURN TRUE; 否则 RETURN 匹配数量 >= 10
- Tracking::TrackReferenceKeyFrame
  - 计算当前帧的Bag of Words
  - 匹配当前帧和ReferenceKeyFrame (ORBMatch::SearchByBoW), 匹配数量少于15, RETURN FALSE
  - 优化当前位姿
  - 去外点；去外点后，若系统使用IMU, RETURN TRUE; 否则 RETURN 匹配数量 >= 10

SideNote: TrackLocalMap

- 在对相机位姿有一个初步估计之后，在localmap中寻找匹配
- UpdateLocalMap() = UpdateLocalKeyFrames() + UpdateLocalPoints()
- SearchLocalPoints()
  - 检查现在CurrentFrame中的mappoints，擦除bad的；其他mappoints的visibile加1
  - 将localmap中的mappoints映射到当前frame，并检查它们的visibility
- 如果IMU没有初始化，优化当前帧位姿
- 如果IMU初始化完成，但当前帧距离上一次重定位帧的距离小于一个阈值，优化当前帧位姿；否则，优化当前帧和前一帧，或者当前帧和前一关键帧
- 更新当前帧的mappoints的信息
  - 若此mappoints不是外点，他的found变量加1；否则，在擦除此mappoint
- 若当前帧距离上次重定位帧距离小于一定阈值 但匹配数量小于50，RETURN FALSE
- 若匹配数量大于10，State为Recently_Lost，RETURN TRUE
- 若系统为Mono_i，匹配小于15 RETURN FALSE；否则 TRUE
- 若系统为Stereo_i，匹配小于15 RETURN FALSE；否则 TRUE
- 其他情况匹配小于30 RETURN FALSE；否则 TRUE

KeyFrame Strategy

需要新关键帧的情况:

1. 系统使用IMU, 且IMU未初始化, 且当前帧到上一个关键帧的时间大于等于0.25

    几个条件

    C1a: 当前帧距离上一关键帧的距离, 大于一定阈值

    C1b: 当前帧距离上一关键帧的距离, 大于一定阈值 (和C1a不一样的值) 且 LocalMapper接受新的关键帧

    C1c: 系统不为Mono, MonoI, StereoI时, 匹配内点少于一定阈值 且 NeedToInsetClose = Track 的效果不好

        NeedToInsetClose = (TrackedClose < 100) && (NonTrackedClose > 70)

    C2: 跟踪点和参照的关键帧的地图点重合少, 匹配内点多来源于VO而不是Map matches

    C3: 针对inertial的一个temporal constraint (当LastKeyFrame存在时)

    - 当前帧距离LastKeyFrame时间大于0.5秒

    C4: 当匹配的内点数量大于15但小于75 或者 State为Recently_LOST, 且系统为Mono_I 时

2. ((C1a || C1b || C1c) && C2) || C3 || C4, 且 LocalMapper接受新的关键帧

3. ((C1a || C1b || C1c) && C2) || C3 || C4, 且 LocalMapper不接受新的关键帧, LocalMapper中的关键帧小于3时

Pipeline: Tracker ➡️ LocalMapper ➡️ LoopCloser

- LocalMapper::InsertKeyFrame
  - Tracking::StereoInitialization()

    In Tracking::Track(), when State == NOT_INITIALIZED
  - Tracking::CreateInitialMapMonocular()

    In Tracking::MonocularInitialization() → Tracking::Track() when State == NOT_INITIALIZED)
  - Tracking::CreateNewKeyFrame()

    In Tracking::Track(), when Tracking::NeedNewKeyFrame() returns True

- LoopClosing::InsertKeyFrame
  - LocalMapping::Run()

LocalMapping::Run()

- ProcessNewKeyFrame （BoW conversion and insertion in Ma)
  - 计算此KF的BoW, 关联mappoints并更新normal和descriptor, 更新covisibility graph, 把当前KF加入atlas地图
- MapPointCulling
  - 检查所有最近加入的mappoints, 在以下条件下擦除该map point
  - ①bad ②found ratio < 0.25 ③ 当前KF的id和该point的第一个kf的id距离超过2 且 观察数少于2 (3 for stereo) ④当前KF的id和该point的第一个kf的id距离超过3
- CreateNewMapPoints
  - Create new Map Points by triangularization
- 若没有新的keyframes in queue需要被处理, SearchInNeighbors
  - Find more matches in neighbor keyframes and fuse point duplications

LocalMapping::Run() CONTINUED

- 若没有新的keyframes in queue需要被处理 且 LocalMapper没有被请求停止
  - 地图中关键帧数量大于2
    - 系统使用IMU且初始化完毕，计算①当前帧到当前关键帧的相机中心距离 与 前一帧到前前帧的相机中心距离之和 小于 0.02 且 ②当前关键帧对应的地图中的InertialBA2变量为false 且 ③当前关键帧到前一关键帧的时间小于10秒，认为相机的动作太小，无法用于完成初始化，ResetActiveMap 且 设置BadIMU为true
    - 否则，直接进行localbundleadjustment
    - 系统使用IMU但没有进行初始化，IMU初始化
    - KeyFrameCulling()
    - Continued..

LocalMapping::Run() CONTINUED

- 若没有新的keyframes in queue需要被处理 且 LocalMapper没有被请求停止
    - 地图中关键帧数量大于2
        - continued...
        - 若当前关键帧距离上一次IMU初始化时的时间戳小于100
            - 当前IMU初始化完成且State为OK
                - Visual-Inertial BA1 (当前关键帧距离上一次IMU初始化时的时间戳大于5秒时)
                    - Set InertialBA1 为True
                    - 若系统为Mono, 重新初始化IMU
                - Visual-Inertial BA2 (当前关键帧距离上一次IMU初始化时的时间戳大于15秒时)
                    - Set InertialBA2 为True
                    - 若系统为Mono, 重新初始化IMU
                - Scale Refinement (Mono only)
        - LoopCloser::InsertKeyFrame()

Next Time

- Initialization
- Relocalization
- Local Map
- Loop Closing