ORBSLAM3 系列代码讲解

关键帧 专题一

主 讲 人：魏宏宇

公 众 号：3D视觉工坊

主要内容

1. 关键帧的选取准则
2. 关键帧的创建
3. 关键帧的属性
4. 关键帧数据库类
5. 讨论与交流
   （SLAM相关书目推荐与建议）

1 关键帧的选取准则

关键帧的定义和作用

- 定义：图像流的频率基本在30HZ或20HZ， 即每一秒记录30或20个图像帧，这些图像帧具有较高的相似性，可用这些图像帧中具有较高代表性的个别图像帧来表征周围图像帧，这些具有代表性的图像帧被称为关键帧，可被长期保留存储在轨迹图像库中。
- 作用：
  - 节约内存，不需要存储所有的图像帧，只需要存储关键帧
  - 利用关键帧之间的几何关系，构建更多的地图点
  - 优化地图点和关键帧的位姿，从而约束全局轨迹位姿

关键帧的选取准则

- **什么样的关键帧是好的关键帧?**

  特征分布均匀，图像清晰，与其他关键帧的共视程度适当，既具有辨识度又具有关联性

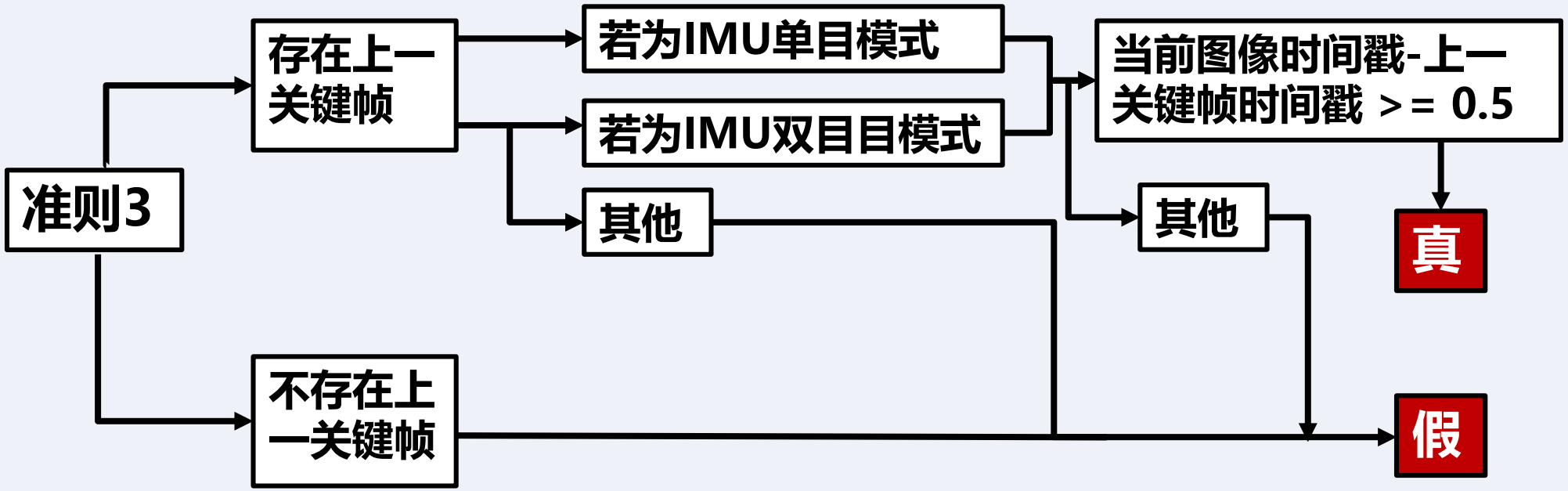
- **关键帧普遍意义上的选取指标：**
  - 与上一关键帧之间间隔的图像**帧数**足够多
  - 与距离最近的关键帧之间的**相机运动**足够大
  - **跟踪质量和共视特征点数量**

关键帧的选取准则 `Tracking::NeedNewKeyFrame()`

- 准则1：控制两个关键帧之间的时间和空间重合度
  - 1.1： 当前图像帧与上一个关键帧之间 需要至少相隔 mMaxFrames个图像帧
  - 1.2：当前图像帧与上一个关键帧之间 至少相隔mMinFrames个图像帧
    且 局部建图线程属于可接受KF状态
  - 1.3：非单目、单目IMU、双目IMU模式
    且 当前图像帧中特征匹配成功数目< 1/4*参考关键帧中跟踪成功的数目 或 需要插入最近地图点
- 准则2： （跟踪和匹配情况）当前图像帧匹配成功数目< thRefRatio*参考KF中跟踪成功的数目
  且 当前图像帧中匹配成功的数目大于15

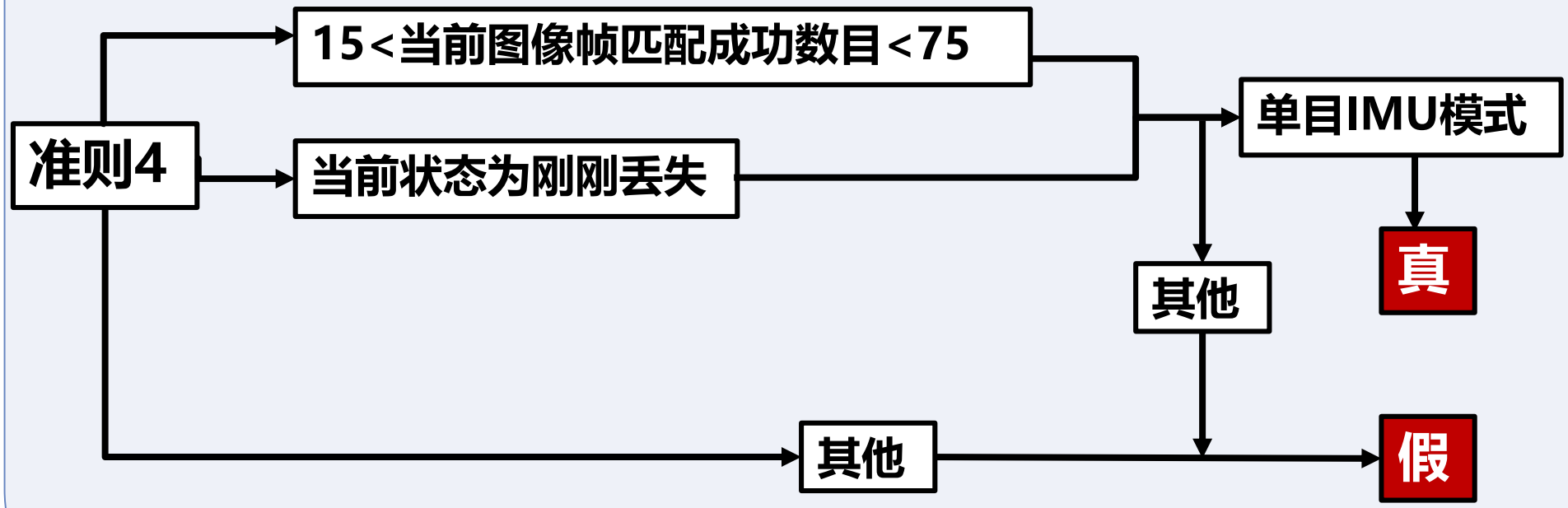
关键帧的选取准则

- 准则3：控制IMU模式下的时间间隔



关键帧的选取准则

- 准则4：



2 关键帧的创建

关键帧的创建

整个系统中有两个部分需要进行关键帧的创建：
1、初始化阶段，构建初始关键帧和当前关键帧
2、跟踪阶段，判断需要插入关键帧，将当前图像帧转为关键帧

```
//用当前图像帧构造关键帧
KeyFrame* pKF = new KeyFrame(mCurrentFrame,mpAtlas->GetCurrentMap(),mpKeyFrameDB);
```

关键帧构造函数：
输入：1、作为关键帧的图像
　　2、当前地图
　　3、当前的关键帧数据库

关键帧的创建函数中主要包括以下步骤：

1、用关键帧的构造函数构造关键帧pKF

2、设置当前图像帧的IMU偏置为新创建的关键帧的IMU偏置

3、设置新关键帧为跟踪线程中的参考关键帧

4、链接新关键帧和上一关键帧之间的关系

5、若是IMU模式，计算IMU预积分

6、将新生成的关键帧插入局部建图线程中

3 关键帧的属性

- 词袋模型 ComputeBoW()
- **关键帧共视图**
- **生成树**
- 地图点观测等操作
- **中值场景深度**

中值场景深度

场景深度即我们平时说的景深，由于单目情况下对场景中地图点的距离值估计不准确，因此需要估计场景深度以对地图点位置进行约束。一般在单目时，采用平均场景深度的概念，即地图点的深度中值，用地图点的深度中值对所有地图点进行统一，保证地图点的场景深度一致。
（KeyFrame::ComputeSceneMedianDepth）

其实现过程是对关键帧中的所有地图点的深度值进行从小到大排序，返回1/2位置的深度值，作为当前场景的平均深度

中值场景深度主要在系统中有两个作用：
1、初始化构建关键帧和地图点时，对新构建的初始地图点进行深度图统一/归一
2、为每个新关键帧构造新的地图点时，利用场景深度和光心之间的基线之比来判断是否两个关键帧距离太近

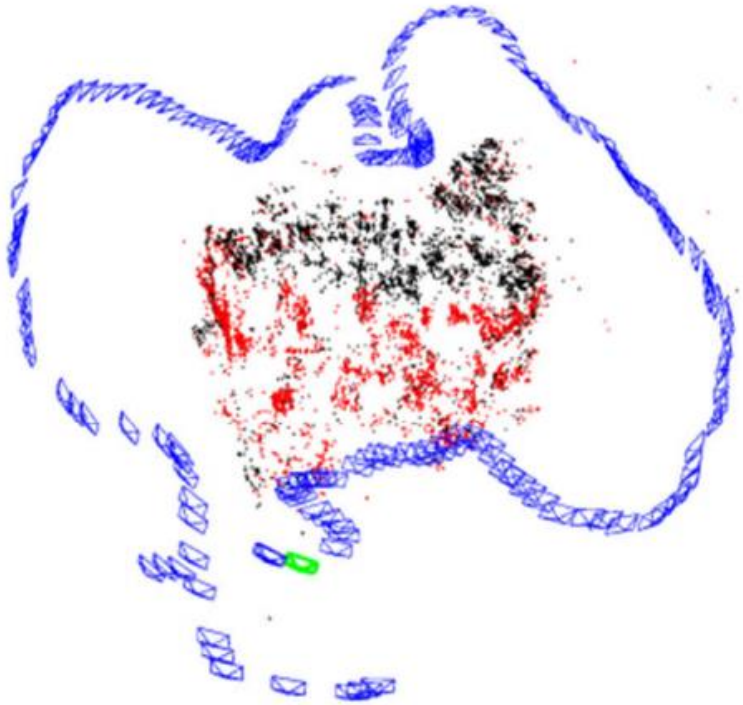
1、初始化构建关键帧和地图点时，对新构建的初始地图点进行深度图统一/归一

```
// 由于只需要计算PC的Z坐标，只取了R的最后一行和t的第三行
cv::Mat Rcw2 = Tcw_.row(2).colRange(0,3);
Rcw2 = Rcw2.t();
float zcw = Tcw_.at<float>(2,3);
for(int i=0; i<N; i++)
{
    if(mvpMapPoints[i])
    {
        MapPoint* pMP = mvpMapPoints[i];
        cv::Mat x3Dw = pMP->GetWorldPos();
        float z = Rcw2.dot(x3Dw)+zcw;
        vDepths.push_back(z);
    }
}

sort(vDepths.begin(),vDepths.end());

return vDepths[(vDepths.size()-1)/q];
```

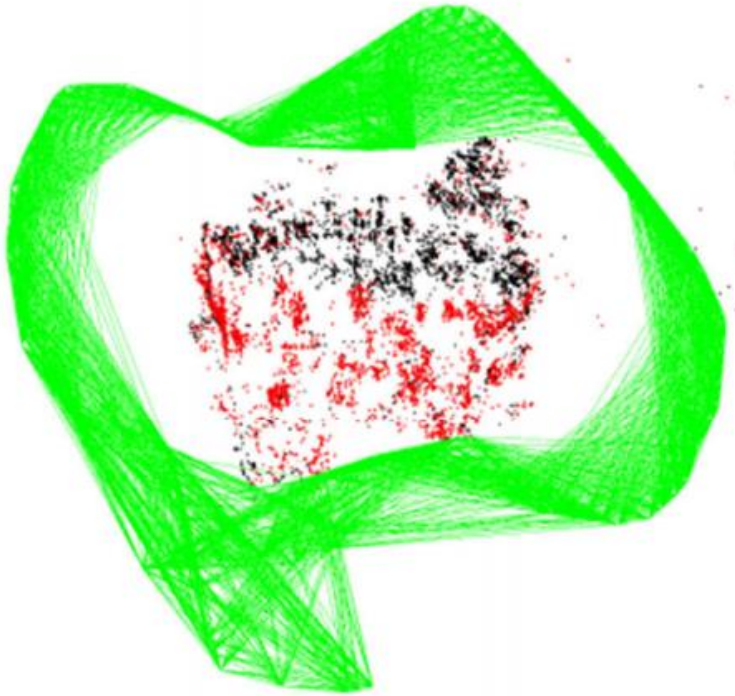
2、为每个新关键帧构造新的地图点时，利用场景深度和光心之间的基线之比来判断是否两个关键帧距离太近

```
    //如果是单目情况
    else
    {
        //计算周围关键帧的场景深度
        const float medianDepthKF2 = pKF2->ComputeSceneMedianDepth(2);
        //基线长度/景深
        const float ratioBaselineDepth = baseline/medianDepthKF2;

        //若比例过小，说明两个关键帧距离太近，恢复的3D点不准，跳过此关键帧
        if(ratioBaselineDepth<0.01)
            continue;
    }
```

蓝色表示地图中的关键帧，绿色表示当前帧，红色表示共视图关键帧的共同观测，黑色表示当前帧观测不到的地图点



(a) KeyFrames (blue), Current Camera (green), MapPoints (black, red), Current Local MapPoints (red)



(b) Covisibility Graph

共视图的含义即是与当前关键帧有共同观测的关键帧所组成的图

共视图：由当前关键帧和当前关键帧的共视关键帧链接构建的网状图
结点: 关键帧们
边：链接关系
权重：共视程度（共视关键帧与当前关键帧共同看到的地图点的数目）

从代码角度出发理解更加容易些：
1、遍历当前关键帧KF_C的所有地图点，获得每个地图点的观测，遍历观测（观测到此地图点的KF和对应的特征点），添加观测情况到向量KFcounter
：《<关键帧1， 与KF_C共有N个共视地图点> <关键帧2， 与KF_C共有M个共视地图点>...》（此处的共视地图点的数目就是边的权重）
2、筛选关键帧，设置阈值为15，选取共视程度超过阈值的关键帧KFs，为选中的Kfs添加链接
3、按照共视程度对选中的关键帧们进行排序，依次作为当前关键帧KF_C的共视图
4、若此共视图是第一次生成，则初始化当前关键帧的父节点为共视程度最大的关键帧，并为父节点添加链接

主要的作用：扩大搜索范围

- **Tracking::UpdateLocalKeyFrames()** ：构造当前图像帧的局部关键帧，利用关键帧的共视关键帧扩大局部关键帧的搜索范围，从而形成更多局部地图点，提高当前图像帧和局部地图点2D-3D匹配数量，优化当前图像帧的位姿
- **LocalMapping::CreateNewMapPoints()**：为局部建图线程中的关键帧构造更多的地图点，需要获得当前关键帧的共视关键帧，以及共视关键帧的共视关键帧以寻找更多匹配，构造更多的可以长期存储的地图点
- **LocalMapping::SearchInNeighbors()**：为防止地图点冗余，融合当前关键帧和周围共视关键帧中相同的地图点，也需要寻找一级和二级共视关键帧
- **LoopClosing::DetectCommonRegionsFromBoW()**：利用选取的候选关键帧判断是否发生回环或者融合，获取更多的共视关键帧也可以提高判断的准确性

生成树：简易版的共视图，共视图保存了当前关键帧和周围共视关键帧的所有共视关系，但生成树只保留了和当前关键帧共视程度最大的共视关键帧，形成父子节点关系

```
// Spanning tree functions
// 生成树
void AddChild(KeyFrame* pKF);
void EraseChild(KeyFrame* pKF);
void ChangeParent(KeyFrame* pKF);
std::set<KeyFrame*> GetChilds();
KeyFrame* GetParent();
bool hasChild(KeyFrame* pKF);
void SetFirstConnection(bool bFirst);
```

- **AddChild:** 添加孩子结点：
  - 1、更新共视图链接时，若当前的关键帧是第一次被连接，则将此关键帧的父节点设置为与此关键帧共视程度最大的关键帧，并为此父节点添加当前关键帧为孩子节点
  - 2、修改父节点时，对孩子节点进行修改

- **EraseChild:** 删除孩子节点：
  - 若当前关键帧被设置为坏关键帧，则删除此关键帧的父节点的孩子节点（当前关键帧）

- **ChangeParent:** 更新父节点
  - 若当前关键帧被设置为坏关键帧，需要对此关键帧的孩子节点的父节点进行修改调整

- **作用：**
  - 1、作为一级关键帧（当前关键帧的共视关键帧）的二级关键帧（父节点，子节点），扩大搜索范围
  - 2、若当前关键帧是坏的关键帧时，可以暂时用父关键帧代替当前关键帧
  - 3、回环线程中，构造本质图和进行地图融合过程

本质图 Essential Graph

本质图的可以说是介于共视图和生成树之间的一种图，主要用于回环检测过程中，利用相似变换来校正尺度漂移，从而实现全局轨迹的优化

4 关键帧数据库类

5 讨论与交流
（SLAM相关书目推荐与建议）

欢迎关注3D视觉工坊

公众号

交流群请添加客服

我们这里有3D视觉算法、SLAM、点云处理、三维重建、计算机视觉、深度学习、自动驾驶、图像处理、技术干货以及前沿paper分享！

如果你也想成为主讲人，欢迎加入我们。

- 报名方式：请发送邮件至vision3d@yeah.net

客服微信，咨询课程

3D视觉工坊知识星球

- 课程PPT和注释代码
- 补充知识点　PDF版和视频版
- 答疑

感谢聆听

Thanks for Listening