`MapPointCulling`函数

理解几个关键属性：

`Visible`属性

该属性表示，能观测到该`MapPoint`的图像帧数目计数器.

在`tracking`线程跟踪局部地图时，使用 `SearchLocalPoints()` 向当前帧增加新的`MapPoints`时，

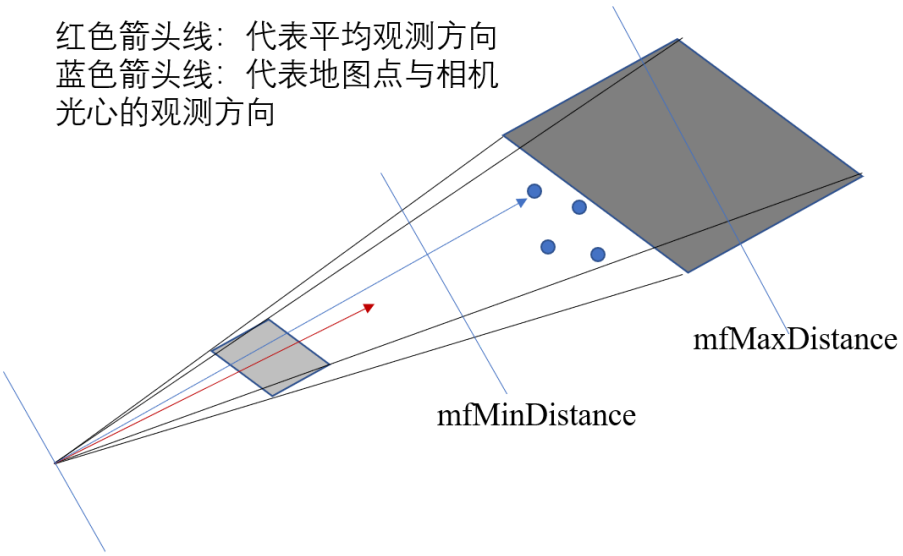
- 首先对当前帧已经匹配上MapPoints各自的`Visible`属性增加计数
- 然后搜索局部地图，只要该MapPoint满足到当前帧的投影条件，就增加一次计数，如下：

```
if(mCurrentFrame.isInFrustum(pMP,0.5)){
    pMP->IncreaseVisible();
    nToMatch++;
}
```

满足上面的投影条件就增加计数，但这些`MapPoint`并不一定能和当前帧的特征点匹配上，这就是 `SearchByProjection()`函数要做的事情. 例如：有一个`MapPoint`(记为M)，在某一帧F的视野范围内，但并不表明该点M可以和F这一帧的某个特征点能匹配上。
注意，这里并没有对这些匹配上的`MapPoints`与当前帧进行关联，仅仅是把匹配上的`MapPoints`添加到当前帧`mvpMapPoints`属性中（因为不是关键帧，所以只做单向关联）。

在`local mapping`中`fuse`当前关键帧新生成的`MapPoints`时、以及形成闭环时，会调用`Replace()`函数进行替换，会对观测更多的`MapPoint`增加计数，增加的数目为原来那个`MapPoint`的计数值。

`isInFrustum`判断地图点是否在视野内



1. 先将地图点反投影到像素坐标系，判断像素坐标是否在图像范围内；
2. 计算地图点到相机光心的距离，判断是否满足图像金字塔的尺度不变特性；
3. 计算地图点到相机光心的观测方向与平均视角向量之间构成的夹角是否满足阈值；
4. 进行地图点的投影；

`mnFound`属性

相比`Visible`属性，`mnFound`的要求就要严的多.

`TrackLocalMap()`函数中，在`PoseOptimization()`优化位姿之后，对非`mvbOutlier`的点执行`IncreaseFound()`，如下：

```
if(!mCurrentFrame.mvbOutlier[i]){
    mCurrentFrame.mvpMapPoints[i]->IncreaseFound();
    ......
}
```

在`local mapping`中`fuse`当前关键帧新生成的`MapPoints`时、以及形成闭环时，会调用`Replace()`函数进行替换，会对观测更多的`MapPoint`增加计数，增加的数目为原来那个`MapPoint`的计数值.

Visible与mnFound作用：
`local mapping`线程中的`MapPointCulling()`函数，会根据VI-B条件1，能找到该点的帧不应该少于理论上观测到该点的帧的`1/4`，如果低于阈值，调用`SetBadFlag()`函数，擦除该`MapPoint`。

`nObs`属性

记录哪些`KeyFrame`的那个特征点能观测到该`MapPoint`，单目`+1`，双目或者grbd`+2`. 注意，该属性只对关键帧有效，如下：

```
void MapPoint::AddObservation(KeyFrame* pKF, size_t idx) {
    unique_lock<mutex> lock(mMutexFeatures);
    if(mObservations.count(pKF))
        return;
......
}
```

这个函数是建立关键帧共视关系的**核心函数**，能共同观测到某些`MapPoints`的关键帧是共视关键帧。

`AddObservation`增加计数

1）在`tracking`线程，`CreateNewKeyFrame()`函数中对双目和`RGBD`相机，需要生成新的`MapPoints`，这一步跟`updateLastFrame()`函数内容相似，但这里生成的`MapPoints`不再是临时的`MapPoints`（`mlpTemporalPoints`），而是添加到关键帧里面，同时`MapPoint`也会添加对该关键帧的观测、计算该`MapPoint`的平均观测方向、观测距离范围、最佳描述子，并加入到`mpMap`中（因为这是新生成的`MapPoint`）。

除了上面这种情况和初始化外，`tracking`线程跟踪过程中，都只与已存在地图中的`MapPoints`进行匹配，并不进行关联（因为不是关键帧，所以只做单向关联），只有在该普通帧确定为关键帧时，才进行关联，关联这步是发生在`local mapping`线程中`ProcessNewKeyFrame()`函数。

2）`local mapping`线程，`ProcessNewKeyFrame()`函数，由于`mpCurrentKeyFrame`中一些`MapPoints`在`TrackLocalMap()`函数中的`MapPoints`与当前关键帧进行了匹配，但没有对这些匹配上的`MapPoints`与当前帧进行关联，所以在这里添加其对`mpCurrentKeyFrame`的观测。

```
if(!pMP->IsInKeyFrame(mpCurrentKeyFrame)){
    // 添加观测
    pMP->AddObservation(mpCurrentKeyFrame, i);
    // 获取该点的平均观测方向和观测距离范围
    pMP->UpdateNormalAndDepth();
    // 加入关键帧后，更新3D点的最佳描述子
    pMP->ComputeDistinctiveDescriptors();
}
else /** @todo this can only happen for new stereo points
inserted by*/
{

    // 将双目或RGBD跟踪过程中新插入的MapPoints放入
mlpRecentAddedMapPoints，等待检查
    // CreateNewMapPoints函数中通过三角化也会生成MapPoints

    // 这些MapPoints都会经过MapPointCulling函数的检验

    mlpRecentAddedMapPoints.push_back(pMP);
```

```
}
```

3）`local mapping`线程，`CreateNewMapPoints()`函数中，通过三角化生成新的3D点（注意，这里还不能叫`MapPoint`），这些3D点需要通过平行、重投影误差、尺度一致性等检查后，才建立一个对应3D点的`MapPoint`对象，然后添加对该关键帧的观测、计算该`MapPoint`的平均观测方向、观测距离范围、最佳描述子，最后加入到`mlpRecentAddedMapPoints`列表中（还要继续检查）。
4）在`local mapping`中`fuse`当前关键帧新生成的`MapPoints`时、以及形成闭环时，会调用`Replace()`函数进行替换，会对观测更多的`MapPoint`，让该`MapPoint`替换掉原来`MapPoint`对应的`KeyFrame`，让原来`MapPoint`对应的`KeyFrame`用`pMP`替换掉原来的`MapPoint`，详细解释见下面`mpReplaced`属性。

`EraseObservation`减少计数

整个擦除过程分三步进行：

- 减少`nObs`属性，与增加相反，单目-1，双目或者grbd -2，并在`mObservations`属性擦出对该关键帧的观测
- 如果要擦出的关键帧是这个`MapPoint`的参考关键帧(`mpRefKF`），则需要重新设置参考关键帧
- 如果少于2个关键帧观测到该`MapPoint`，则删除该`MapPoint`，即通过`MapPoint::SetBadFlag()`实现

```
mObservations.erase(pKF);
// 如果该keyFrame是参考帧，该Frame被删除后重新指定RefFrame
if(mpRefKF==pKF)
    mpRefKF=mObservations.begin()->first; //重设参考关键帧

// 如果少于2个关键帧观测到该MapPoint，则删除该MapPoint*/
if(nObs<=2)
    bBad=true;
```

1）`LocalBundleAdjustment()`函数会对误差比较大的边，在关键帧中剔除对该`MapPoint`的观测`KeyFrame::EraseMapPointMatch()`，同时在`MapPoint`中剔除对该关键帧的观测`MapPoint::EraseObservation()`实现，如下

```
if(!vToErase.empty())
{
    for(size_t i=0;i<vToErase.size();i++)
    {
        KeyFrame* pKFi = vToErase[i].first;
        MapPoint* pMPi = vToErase[i].second;
        pKFi->EraseMapPointMatch(pMPi);
        pMPi->EraseObservation(pKFi);
    }
}
```

2）在擦除关键帧的时候，记得要解除关键帧和`MapPoints`的观测关系，即`KeyFrame::SetBadFlag()`函数要做的事之一。