东北大学 Northeastern University

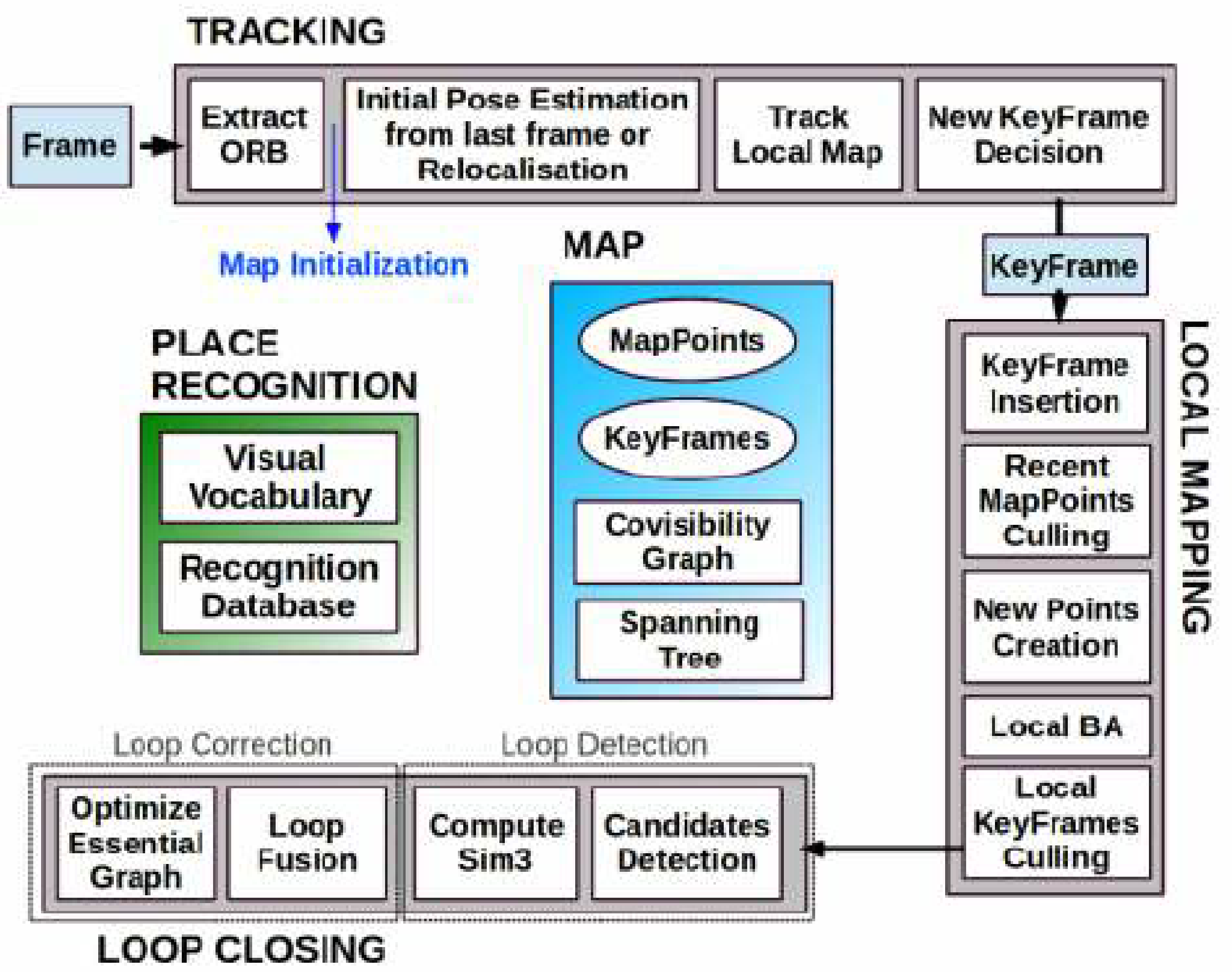
信息科学与工程学院
人工智能与机器人研究所

# ORB-SLAM代码详细解读

吴博 @泡泡机器人
657390323@qq.com
2016.8.29

# 代码主要结构






Tracking.cpp

LocalMapping.cpp

LoopClosing.cpp

Viewer.cpp

变量命名规则：

"p" 表示指针数据类型　　"n"表示int类型

"b" 表示bool类型　　"s"表示set类型

"v" 表示vector数据类型　　'l'表示list数据类型

"m" 表示类成员变量

# System入口：

**输入图像**
- GrabImageStero(imRectLeft, imRectRight)
- GrabImageRGBD(imRectLeft, imRectRight)
- GrabImageMonocular(im)

↓

**转为灰度图**
- Stero: mImGray, imGrayRight
- RGBD: mImGray, imDepth
- Mono: mImGray

↓

**构造Frame**
- Stero: Frame(mImGray, imGrayRight, mpORBextractorLeft, mpORBextractorRight)
- RGBD: Frame(mImGray, imDepth, mpORBextractorLeft)
- Mono(未初始化): Frame(mImGray, mpIniORBextractor)
- Mono(已初始化): Frame(mImGray, mpORBextractorLeft)

↓

**Track**

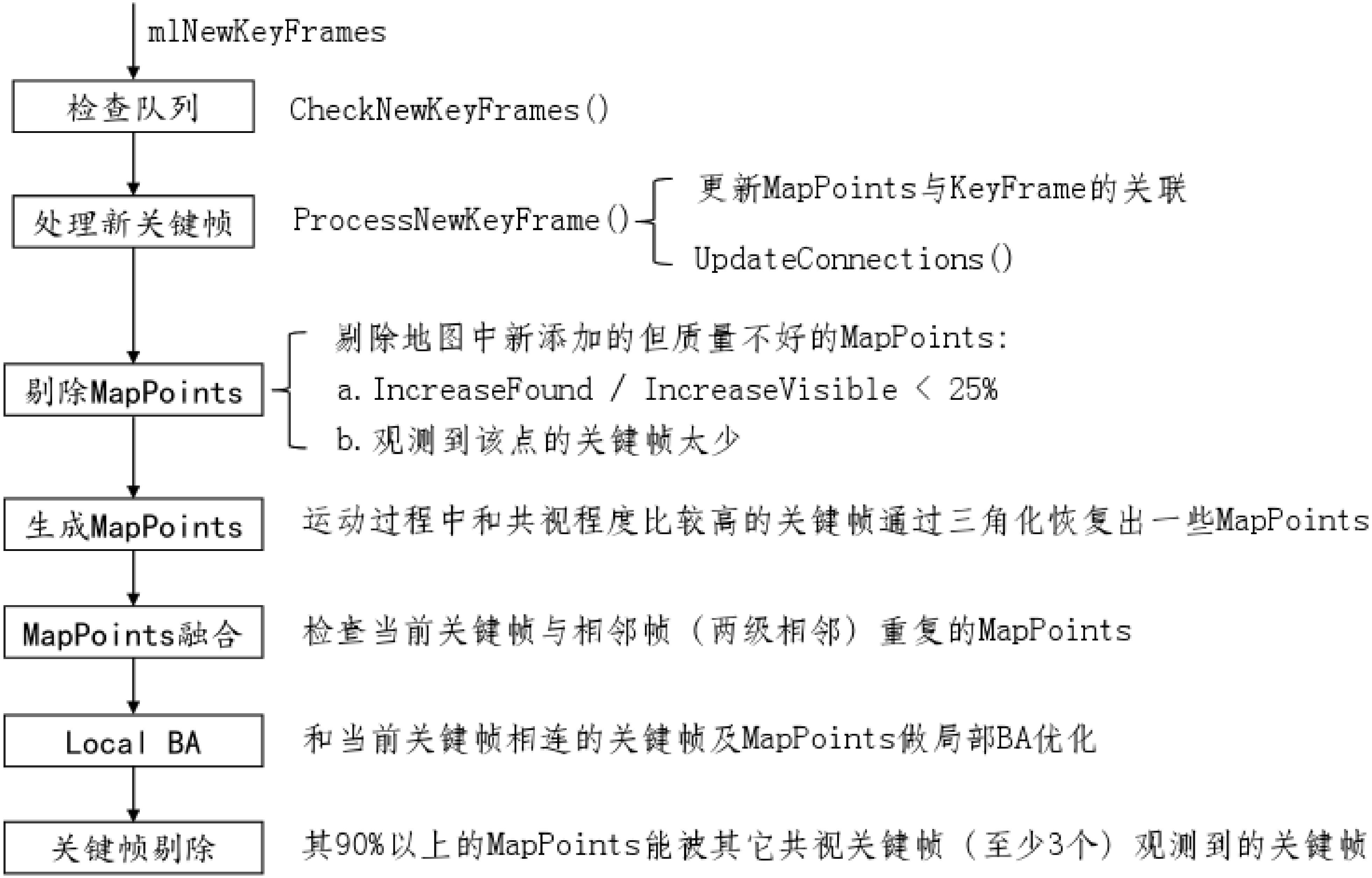
数据流进入Tracking线程

注：mpIniORBextractor相比mpORBextractorLeft提取的特征点多一倍

# Tracking线程：


注： mbOnlyTracking默认为false，用户可通过运行界面选择仅跟踪定位模式

# LocalMapping线程：



# LocalClosing线程(闭环检测):

mlploopKeyFrameQueue

队列中取一帧

mpcurrentKF

判断距离上一次闭环检测是否超过10帧

计算当前帧与相连关键帧的Bow最低得分

mpcurrentKF
minscore

检测得到闭环候选帧

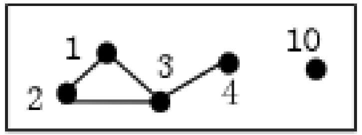
vpLoopCandidates

检测候选帧连续性

1、三个阈值都是计算获得，鲁邦性好
minscore mincommons minscoreToRetain

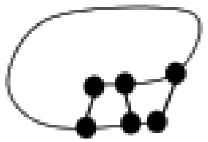
2、通过分组可以将单独得分很高的无匹配关键帧剔除

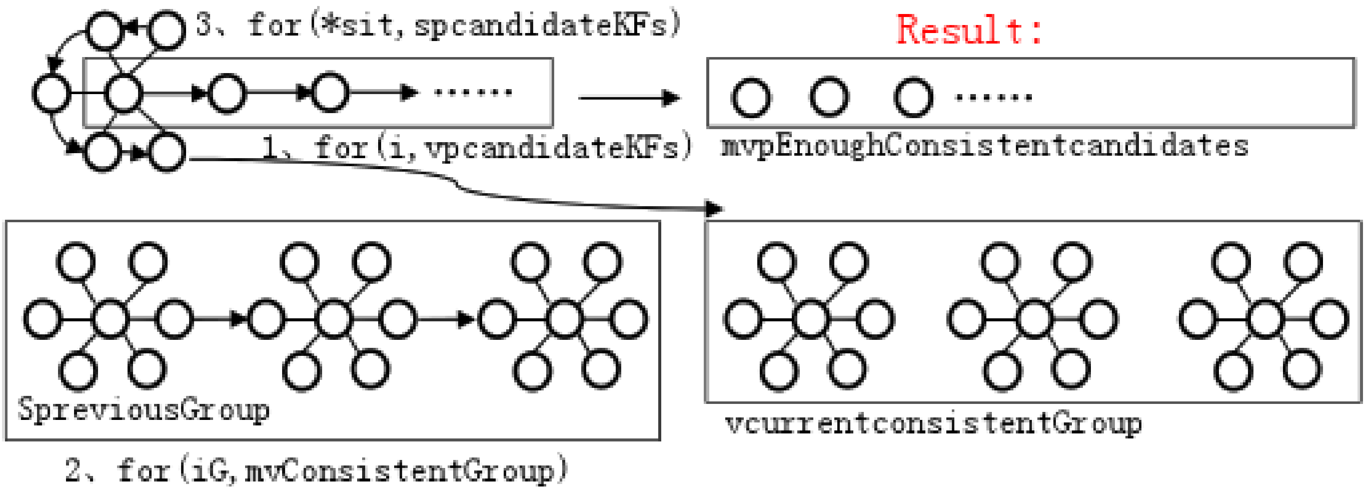
分组示意图:




如图：1、2、3、4、10都是闭环候选帧。
节点1：与2、3相连，1与2、3分为一组
节点2：与1、3相连，2与1、3分为一组
节点3：与1、2、4相连，3与1、2、4分为一组
节点4：与3相连，4与3分为一组
节点10：10自己单独一组

连续性检测示意图:






(pKF, minscore)

找出与当前帧有公共单词的关键帧，但不包括与当前帧相连的关键帧

lKFsharingwords

统计候选帧中与pKF具有共同单词最多的单词数

maxcommonwords

得到阈值
mincommons=0.8*maxcommonwords

maxcommonwords
mincommons
minscore

筛选共有单词大于mincommons且Bow得分大于minscore的关键帧

lscoreAndMatch

将存在相连的分为一组，计算组最高得分bestAccScore，同时得到每组中得分最高的关键帧

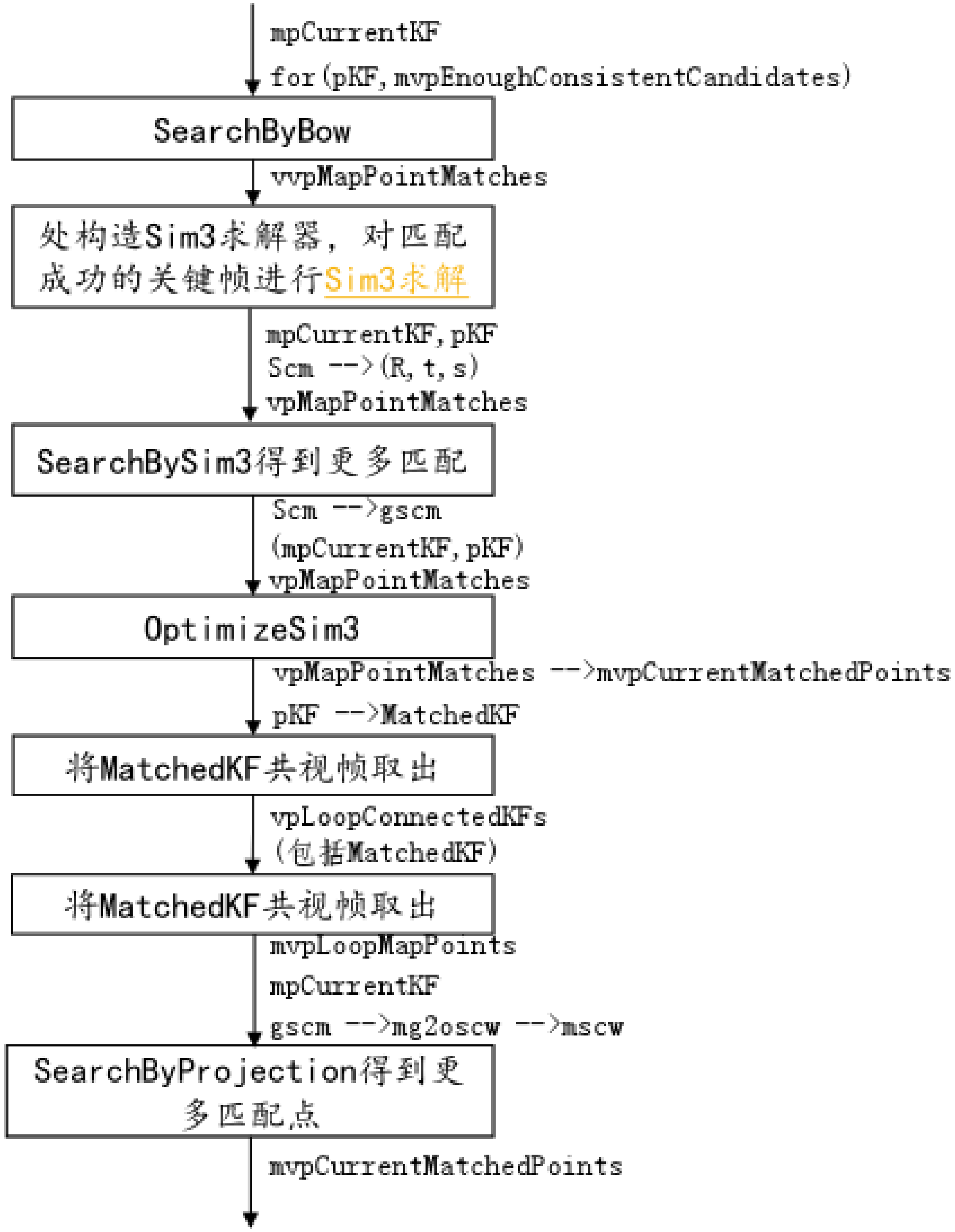
lsAccScoreAndMatch
bestAccScore

得到阈值minScoreToRetain
=0.75*bestAccScore

lsAccScoreAndMatch
minScoreToRetain

vpLoopCandidates

# LocalClosing线程(Sim3计算):



# LocalClosing线程(Sim3计算):

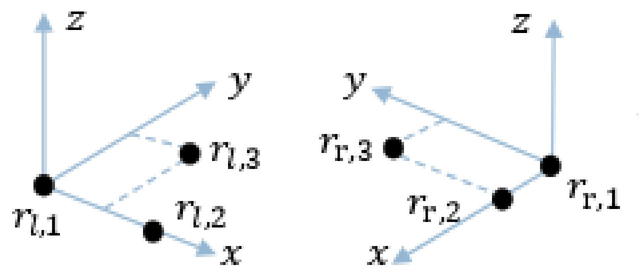
三对匹配3D，分别对左右三个3D点建立坐标系：

X轴： $\hat{x}_l = x_l / \|x_l\|$ 其中： $x_l = r_{l,2} - r_{l,1}$ (1)

Y轴： $\hat{y}_l = y_l / \|y_l\|$ 其中： $y_l = (r_{l,3} - r_{l,1}) - [(r_{l,3} - r_{l,1}) \cdot \hat{x}_l]\hat{x}_l$ (2)

Z轴： $\hat{z}_l = \hat{x}_l \times \hat{y}_l$ (3)

右坐标系同理，且令： $M_l = |\hat{x}_l \hat{y}_l \hat{z}_l|$ $M_r = |\hat{x}_r \hat{y}_r \hat{z}_r|$ (4)




如果左边坐标系有一个向量 $r_l$ ，那么： $M_l^T r_l$ 可以得到 $r_l$ 向量沿着坐标轴的值

左乘 $M_r$ 可以变换到右坐标系，故可推导出旋转：

$$r_r = M_r M_l^T r_l \Longrightarrow R = M_r M_l^T \quad (5)$$

计算平移量：

质心： $\bar{r}_l = \frac{1}{n}\sum_{i=1}^{n} r_{l,i}$ $\bar{r}_r = \frac{1}{n}\sum_{i=1}^{n} r_{r,i}$ (5)

原点移到质心： $r'_{l,i} = r_{l,i} - \bar{r}_l$ $r'_{r,i} = r_{r,i} - \bar{r}_r$ (6)

$$\sum_{i=1}^{n} r'_{l,i} = 0 \qquad \sum_{i=1}^{n} r'_{r,i} = 0$$

$$r'_{r,i} = sR(r'_{l,i}) - r'_0 \Longrightarrow r'_0 = r_0 - \bar{r}_r + sR(\bar{r}_l) \quad (7)$$

优化函数： $\sum_{i=1}^{n} \|e_i\|^2 = \sum_{i=1}^{n} \|r'_{r,i} - [sR(r'_{l,i}) + r'_0]\|^2$ (8)

$$= \sum_{i=1}^{n} \|r'_{r,i} - sR(r'_{l,i})\|^2 - 2r'_0 \cdot \sum_{i=1}^{n}\left[r'_{r,i} - sR(r'_{l,i})\right] + n\|r'_0\|^2$$

$\sum_{i=1}^{n} \|r'_{r,i} - sR(r'_{l,i})\|^2$ 与优化量 $r'_0$ 无关

$$\min \sum_{i=0}^{n} \|e_i\|^2 \Longrightarrow r'_0 = 0 \Longrightarrow t = r_0 = \bar{r}_r - sR(\bar{r}_l) \quad (9)$$

# LocalClosing线程(Sim3计算)：

计算尺度：

由于 $r_0' = 0 \Longrightarrow \sum_{i=1}^{n} \|e_i\|^2 = \sum_{i=1}^{n} \|r_{r,i}' - sR(r_{l,i}')\|^2$ (10)

$$\Longrightarrow \sum_{i=1}^{n} \|e_i\|^2 = \sum_{i=1}^{n} \|r_{r,i}'\|^2 - 2s\sum_{i=1}^{n} r_{r,i}' \cdot R(r_{l,i}') + s^2 \sum_{i=1}^{n} \|R(r_{l,i}')\|^2$$

$$\Longrightarrow \sum_{i=1}^{n} \|e_i\|^2 = \sum_{i=1}^{n} \|r_{r,i}'\|^2 - 2s\sum_{i=1}^{n} r_{r,i}' \cdot R(r_{l,i}') + s^2 \sum_{i=1}^{n} \|r_{l,i}'\|^2$$

令：$\sum_{i=1}^{n} \|e_i\|^2 = S_r - 2sD + s^2 S_l = \left(s\sqrt{S_l} - D/\sqrt{S_l}\right)^2 + (S_r S_l - D^2)/S_l$ (11)

$$\Longrightarrow s = \left(\sum_{i=1}^{n} r_{r,i}' \cdot R(r_{l,i}')\right) \Big/ \sum_{i=1}^{n} \|r_{l,i}'\|^2 \quad (12)$$

根据对称性： $r_r = sR(r_l) + r_0 \qquad r_l = \bar{s}\bar{R}(r_r) + \bar{r}_0$

$$\Longrightarrow \bar{s} = 1/s \qquad \bar{r}_0 = -\frac{1}{s}R^{-1}(r_0) \qquad \bar{R} = R^{-1}$$

可是：$\bar{s} = 1/s \neq \left(\sum_{i=1}^{n} r_{l,i}' \cdot \bar{R}(r_{r,i}')\right) \Big/ \sum_{i=1}^{n} \|r_{r,i}'\|^2$

如果公式10变为： $e_i = \frac{1}{\sqrt{s}} r_{r,i}' - \sqrt{s}R(r_{l,i}')$

则公式11变为：

$$\frac{1}{s}S_r - 2D + sS_r = \left(\sqrt{s}S_L - \frac{1}{\sqrt{s}}S_r\right)^2 + 2(S_l S_r - D)$$

$$\Longrightarrow s = \left(\sum_{i=1}^{n} \|r_{r,i}'\|^2 \Big/ \sum_{i=1}^{n} \|r_{l,i}'\|^2\right)^{1/2} \quad (13)$$

# LocalClosing线程(Sim3计算):

如果对于大于三组匹配点:

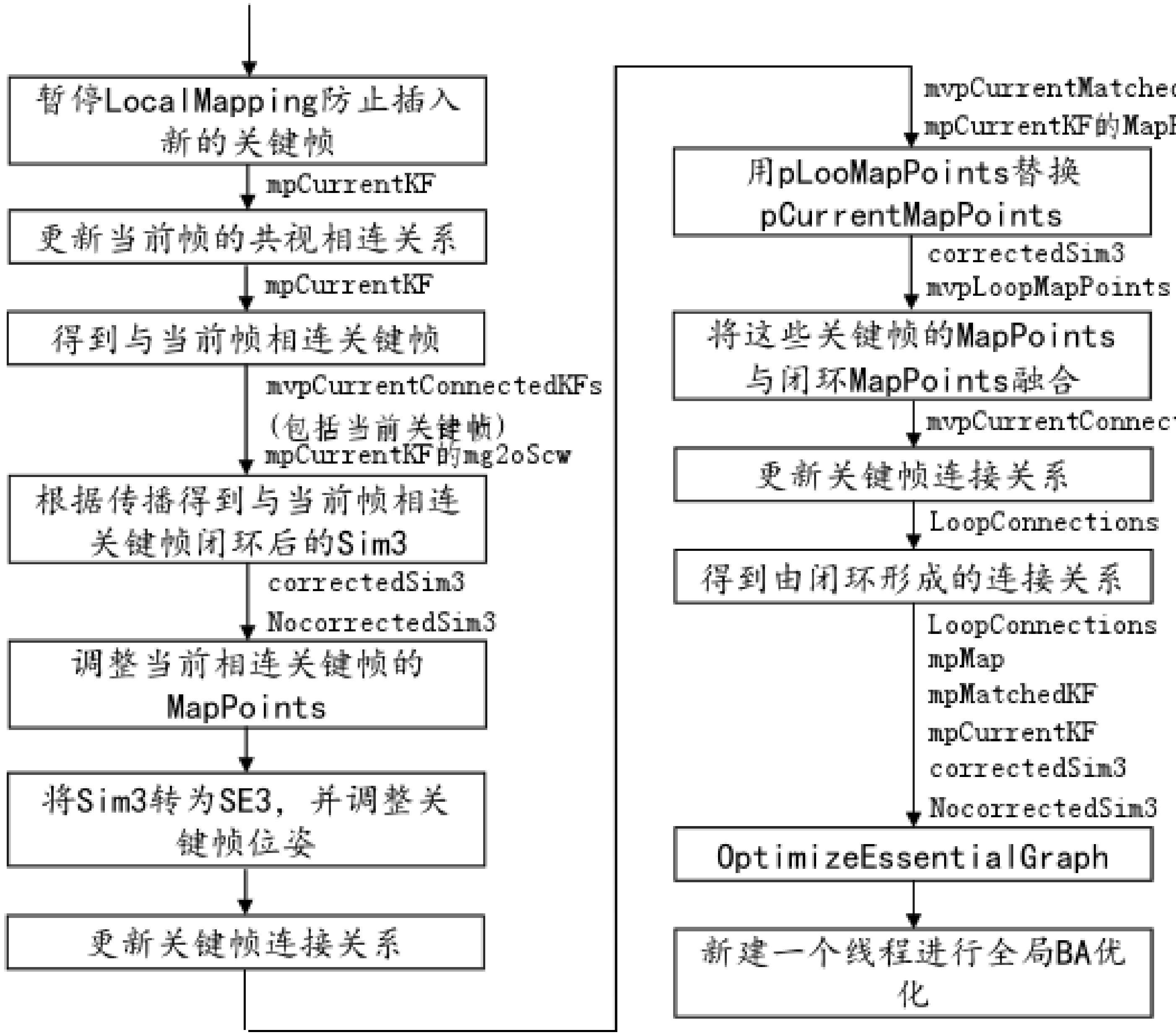
令: $$M=\begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yx} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix}=\sum_{i=1}^{n} r'_{l,i} r'^{T}_{r,i}$$

$$N=\begin{bmatrix} (S_{xx}+S_{yy}+S_{zz}) & S_{yz}-S_{zy} & S_{zx}-S_{xz} & S_{xy}-S_{yx} \\ S_{yz}-S_{zy} & (S_{xx}-S_{yy}-S_{zz}) & S_{xy}+S_{yx} & S_{zx}+S_{xz} \\ S_{zx}-S_{xz} & S_{xy}+S_{yx} & (-S_{xx}+S_{yy}-S_{zz}) & S_{yz}+S_{zy} \\ S_{xy}-S_{yx} & S_{zx}+S_{xz} & S_{yz}+S_{zy} & (-S_{xx}-S_{yy}+S_{zz}) \end{bmatrix}$$

特征值分解N矩阵: N最小特征值对应的特征向量就是待求四元数

四元数转欧拉角: $q=\cos(\theta/2)+n\sin(\theta/2)$

# LocalClosing线程(correctLoop):




共视关键帧：

有共同MapPoints观测的关键帧

Covisibility Graph:

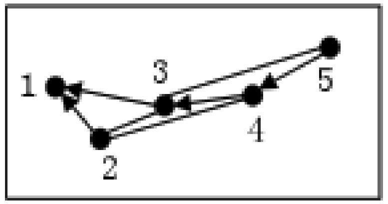
共视关键帧及它们的连接关系构成的图

Essential Graph（包含3部分）：

a. 扩展树连接关系

b. 闭环连接关系

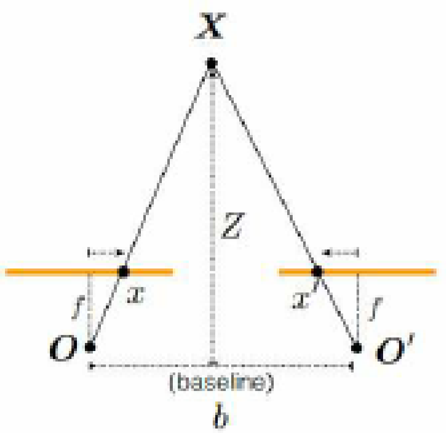
c. 共视关系非常好的连接关系



# Frame.cpp:

- 双目立体匹配
  - 为左目每个特征点建立带状区域搜索表，限定搜索区域。（已提前极线校正）
  - 通过描述子进行特征点匹配，得到每个特征点最佳匹配点scaleduR0
  - 通过SAD滑窗得到匹配修正量bestincR
  - (bestincR,dist) (bestincR-1,dist) (bestincR+1,dist)三个点拟合出抛物线，得到亚像素修正量deltaR
  - 最终匹配点位置为：scaleduR0 + bestincR + deltaR

- Disparity与Depth




$$\frac{X}{Z} = \frac{x}{f} \qquad \frac{b - X}{Z} = \frac{x'}{f}$$

$\Rightarrow$ Disparity: $d = x - x' = \frac{bf}{Z}$

# 单目Initializer.cpp:

- 特征点归一化，坐标均值为0，一阶绝对矩为1

$$mean\_x = (\sum_{i=0}^{N} u_i)/N \qquad mean\_y = (\sum_{i=0}^{N} v_i)/N \qquad (1)$$

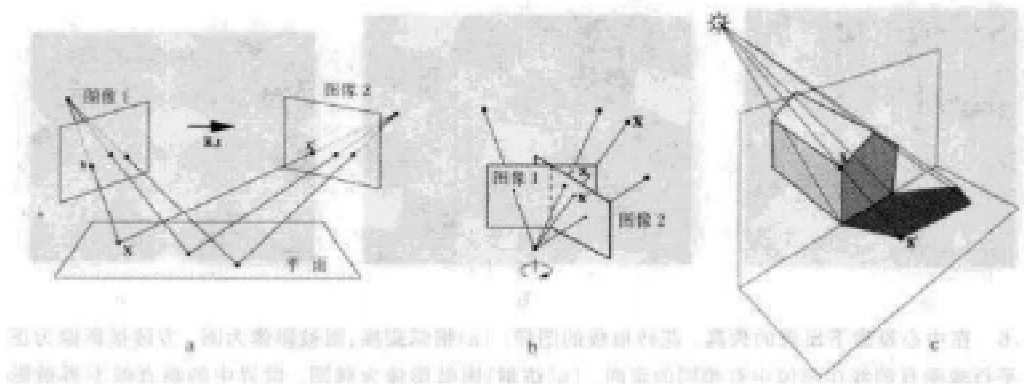
$$mean\_devx = (\sum_{i=0}^{N} |u_i - mean\_x|)/N \qquad mean\_devy = (\sum_{i=0}^{N} |v_i - mean\_y|)/N \qquad (2)$$

$$sX = 1/mean\_devx \qquad sY = 1/mean\_devy \qquad (3)$$

$$T = \begin{bmatrix} sX & 0 & -meanx * sX \\ 0 & sY & -meany * sY \\ 0 & 0 & 1 \end{bmatrix} \qquad (4)$$

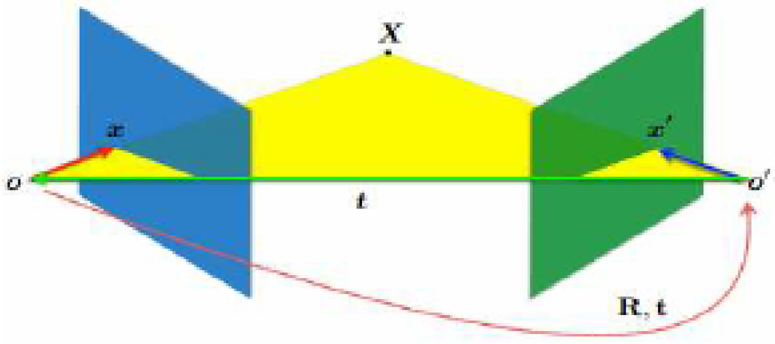
- 单应性矩阵模型（Homograph Matrix）

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}_{right} = \lambda \begin{bmatrix} h1 & h2 & h3 \\ h4 & h5 & h6 \\ h7 & h8 & h9 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}_{right} \qquad (5)$$



# 单目Initializer.cpp:

- 对极几何模型（Fundamental Matrix）




刚体旋转 $x' = R(x - t)$ (1)

共平面 $(x - t)^T(t \times x) = 0$ (2)

$\Longrightarrow (x'^T R)(t \times x) = 0 \Longrightarrow (x'^T R)([t_\times]x) = 0$

$\Longrightarrow x'^T(R[t_\times])x = 0 \Longrightarrow x'^T E x = 0$ (3)

图像坐标系转相机坐标系：

$\hat{x} = k^{-1}x \quad \hat{x}' = k^{-1}x'$ (4)

$\Longrightarrow x'^T F x = 0 \quad F = K'^{-T} E K^{-1}$ (5)

- Homograph矩阵求解（归一化4点算法）

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}_{right} = \lambda \begin{bmatrix} h1 & h2 & h3 \\ h4 & h5 & h6 \\ h7 & h8 & h9 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}_{right} \Longrightarrow x' = \lambda H x$$

DLT求解： $x' \times Hx = 0 \Longrightarrow Ah = 0$ (6)

其中： $A = \begin{bmatrix} 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \\ -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \end{bmatrix} \Longrightarrow$ 特征值分解分解： $A^T A$ 最小特征值对应的特征向量

$h = [h1 \quad h2 \quad h3 \quad h4 \quad h5 \quad h6 \quad h7 \quad h8 \quad h9]'$

# 单目Initializer.cpp:

- **Fundamental** 矩阵求解（归一化8点算法）

$$x'Fx = 0 \qquad F = \begin{bmatrix} f1 & f2 & f3 \\ f4 & f5 & f6 \\ f7 & f8 & f9 \end{bmatrix}$$

$\Rightarrow$ $Af = 0 \quad (1)$

其中： $A = [x'x \quad x'y \quad x' \quad y'x \quad y'y \quad y' \quad x \quad y \quad 1]$

$f = [f1 \quad f2 \quad f3 \quad f4 \quad f5 \quad f6 \quad f7 \quad f8 \quad f9]'$

$\Rightarrow$ 特征值分解分解：$A^TA$

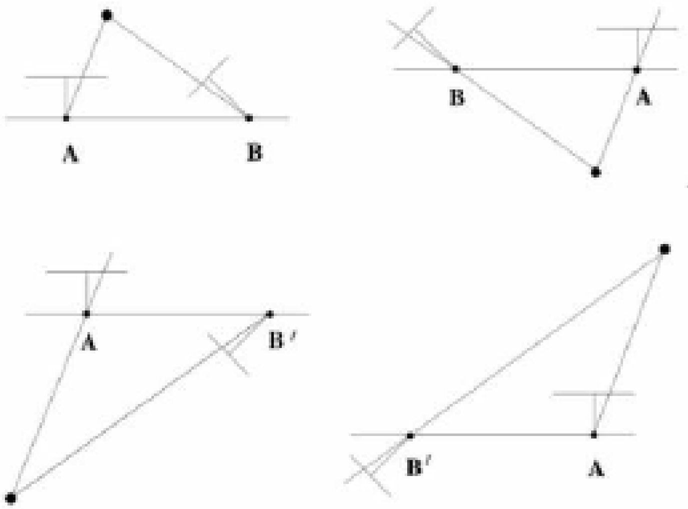
最小特征值对应的特征向量

- **Fundamental** 矩阵分解

$E = k'^T Fk$

SVD分解：$E = U\Sigma V^T$ 令：$W = R_z(\frac{\pi}{2}) = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$$E = [R|T] \begin{cases} R_1 = UWV^T \quad R_2 = UW^TV^T \\ T_1 = U_3 \quad T_2 = -U_3 \end{cases}$$



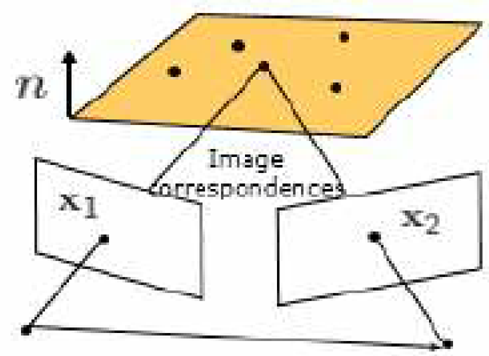

（R T）选择：3D点出现在两个相机前方最多的模型

统计四个模型中3D点在摄像头前方且投影误差小于阈值的3D点个数，以及每个模型下较大的视差角。

如果其中一个模型的视差角大于阈值，并且满足条件的3D点个数明显大于其它模型，那么这个模型就是最优选择

# 单目Initializer.cpp:

- **Homograph** 矩阵分解（ Faugeras SVD-based decomposition ）




$\{X1, X2\}$ 是相机坐标系匹配的特征点

$aX + bY + cZ = d$ 即 $\frac{1}{d} n^T X = 1$ 表示3D点共同所在的平面，N为平面法向量

$X = \lambda_1 X_1$ 表示$X_1$在平面上对应的3D点，世界坐标系与第一个相机坐标系重合

$$\lambda_2 X_2 = RX + t$$

$$= R(\lambda_1 X_1) + t$$

将所有的3D点共平面这个约束引入上式：

$$\lambda_2 X_2 = R(\lambda_1 X_1) + t \cdot \frac{1}{d} n^T (\lambda_1 X_1)$$

$$\Rightarrow \quad X_2 = \lambda \left( R + \frac{1}{d} t n^T \right) X_1$$

$$= \lambda H X_1$$

{x1,x2} 是图像坐标系匹配特征点，则：

$$x_2 = \lambda G x_1 \quad G = KHK^{-1}$$

令： $A = dR + tn^T$

对A奇异值分解： $A = U \Lambda V^T$

则： $\Lambda = U^T A V = dU^T R V + (U^T t)(V^T n)^T$

考虑： $s = \det U \det V \quad s^2 = 1$

则： $\Lambda = s^2 d U^T R V + (U^T t)(V^T n)^T$

$$= (sd)(sU^T R V) + (U^T t)(V^T n)^T$$

令： $R' = sU^T R V \quad t' = U^T t \quad n' = V^T n \quad d' = sd$

则： $\Lambda = d' R' + t' n'^T$

引入s是为了说明R’ 与R，d’ 与d存在符号取反的可能

# 单目Initializer.cpp:

- **Homograph 矩阵分解**

$$\Lambda = \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{bmatrix} = d'R' + t'n'^T \quad (1)$$

取: $e_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad e_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

则: $n' = \begin{bmatrix} x_1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ x_2 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ x_3 \end{bmatrix} = x_1 e_1 + x_2 e_2 + x_3 e_3$

$$x_1{}^2 + x_2{}^2 + x_3{}^2 = 1 \quad (2)$$

(1) 式变为:

$$[d_1 e_1 \quad d_2 e_2 \quad d_3 e_3] = [d'R'e_1 \quad d'R'e_2 \quad d'R'e_3] + [t'x_1 \quad t'x_2 \quad t'x_3] \quad (3)$$

(3) 式也可以拆成3个等式:

$$\begin{cases} d_1 e_1 = d'R'e_1 + t'x_1 & (4) \\ d_2 e_2 = d'R'e_2 + t'x_2 & (5) \\ d_3 e_3 = d'R'e_3 + t'x_3 & (6) \end{cases}$$

# 单目Initializer.cpp:

- **Homograph 矩阵分解**

(4) (5) (6) 中每两个式子消去t’ 可得:

$$\begin{cases} d'R'(x_2e_1 - x_1e_2) = d_1x_2e_1 - d_2x_1e_2 \\ d'R'(x_3e_2 - x_2e_3) = d_2x_3e_2 - d_3x_2e_3 \\ d'R'(x_1e_3 - x_3e_1) = d_3x_1e_3 - d_1x_3e_1 \end{cases} \quad (7)$$

因为: $\|R'X\| = \|X\|$

对 (7) 式中三个式子的左右两边同时取范数可得:

$$\begin{cases} (d'^2 - d_2{}^2)x_1{}^2 + (d'^2 - d_1{}^2)x_2{}^2 = 0 \\ (d'^2 - d_3{}^2)x_2{}^2 + (d'^2 - d_2{}^2)x_3{}^2 = 0 \\ (d'^2 - d_1{}^2)x_3{}^2 + (d'^2 - d_3{}^2)x_1{}^2 = 0 \end{cases} \quad (8)$$

对于 (8) 式如果令:

$$d'^2 - d_1{}^2 = a \qquad d'^2 - d_2{}^2 = \mathrm{b} \qquad d'^2 - d_3{}^2 = \mathrm{c}$$

则 (8) 式简写为:

$$\begin{bmatrix} b & a & 0 \\ 0 & c & b \\ c & 0 & a \end{bmatrix} \begin{bmatrix} x_1{}^2 \\ x_1{}^2 \\ x_1{}^2 \end{bmatrix} = 0 \text{ 则 } \det \begin{pmatrix} b & a & 0 \\ 0 & c & b \\ c & 0 & a \end{pmatrix} = 0$$

则 $abc = 0$

$$(d'^2 - d_1{}^2)(d'^2 - d_2{}^2)(d'^2 - d_2{}^2) = 0 \quad (9)$$

因为: $d_1 \geq d_2 \geq d_3$ (10)

对于 (9) 式，可分成以下三种情况:

$$\begin{cases} d_1 \neq d_2 \neq d_3 \\ d_1 = d_2 \neq d_3 \text{ 或 } d_1 \neq d_2 = d_3 \\ d_1 = d_2 = d_3 \end{cases}$$

这三种情况下均可以得到: $d' = \pm d_2$

现对第一种情况用反证法进行证明:

如果: $d' = \pm d_1$ 或 $d' = \pm d_3$

根据 (8) 式可得:

$$\begin{cases} x_1 = 0 \\ (d_1{}^2 - d_3{}^2)x_2{}^2 + (d_1{}^2 - d_2{}^2)x_3{}^2 = 0 \\ d_1 > d_2 > d_3 \end{cases}$$

推出: $x_1 = x_2 = x_3 = 0$

与 $x_1{}^2 + x_2{}^2 + x_3{}^2 = 1$ 矛盾

# 单目Initializer.cpp:

- **Homograph 矩阵分解**

经过上一页的说明，（3）式的解分以下几种情况：

$$
d' = d_2 > 0 \quad \begin{cases} d_1 \neq d_2 \neq d_3 & (11) \\ d_1 = d_2 \neq d_3 \text{ 或 } d_1 \neq d_2 = d_3 & (12) \\ d_1 = d_2 = d_3 & (13) \end{cases}
$$

$$
d' = -d_2 < 0 \quad \begin{cases} d_1 \neq d_2 \neq d_3 & (14) \\ d_1 = d_2 \neq d_3 \text{ 或 } d_1 \neq d_2 = d_3 & (15) \\ d_1 = d_2 = d_3 & (16) \end{cases}
$$

# 单目Initializer.cpp:

- **Homograph 矩阵分解**

对于（11）式这种情况：

根据（8）式三个方程可解得：

$$n'=\begin{cases} x_1=\varepsilon_1\sqrt{\dfrac{{d_1}^2-{d_2}^2}{{d_1}^2-{d_3}^2}} \\ x_2=0 \\ x_3=\varepsilon_2\sqrt{\dfrac{{d_2}^2-{d_3}^2}{{d_1}^2-{d_3}^2}} \\ \varepsilon_1\ \ \varepsilon_2=\pm 1 \end{cases} \qquad (17)$$

将 $n'$ 带入（5）式可得：

$$R'e_2=e_2$$

因此可以得到R’的形式为：

$$R'=\begin{bmatrix}\cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta\end{bmatrix} \qquad (18)$$

将（18）（19）带入（3）可得：

$$t'=(d_1-d_3)\begin{pmatrix}x_1\\0\\x_3\end{pmatrix} \qquad (20)$$

将（17）和（18）带入（7）式第三个可得（18）式中的：$\sin\theta$ $\cos\theta$

$$d'\begin{bmatrix}\cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta\end{bmatrix}\begin{bmatrix}-x_3\\0\\x_1\end{bmatrix}=\begin{bmatrix}-d_1x_3\\0\\d_3x_1\end{bmatrix} \Rightarrow \begin{cases} \sin\theta=\dfrac{(d_1-d_3)}{d_2}x_1x_3=\varepsilon_1\varepsilon_3\dfrac{\sqrt{({d_1}^2-{d_2}^2)({d_2}^2-{d_3}^2)}}{(d_1+d_3)d_2} \\ \cos\theta=\dfrac{d_3{x_1}^2+d_1{x_3}^2}{d_2}=\dfrac{d_1d_3+{d_2}^2}{(d_1+d_3)d_2} \end{cases} \qquad (19)$$

# 单目Initializer.cpp:

- **Homograph 矩阵分解**

对于（12）式这种情况，（11）情况的特例：

根据（8）式三个方程可解得：

$$n' = \begin{cases} x_1 = 0 \\ x_2 = 0 \\ x_3 = \pm 1 \end{cases}$$

将 $n'$ 带入（5）式可得：

$$R' = \mathrm{I}$$

带入（3）式可得：

$$t' = (d_3 - d_1)n'$$

对于（13）式这种情况，（11）情况的特例：

$R' = I \quad t' = 0$

$n'$ 未定义

# 单目Initializer.cpp:

## Homograph 矩阵分解

对于（14）式这种情况:

根据（8）式三个方程可解得:

$$n'=\begin{cases} x_1=\varepsilon_1\sqrt{\dfrac{{d_1}^2-{d_2}^2}{{d_1}^2-{d_3}^2}} \\ x_2=0 \\ x_3=\varepsilon_2\sqrt{\dfrac{{d_2}^2-{d_3}^2}{{d_1}^2-{d_3}^2}} \\ \varepsilon_1\ \varepsilon_2=\pm 1 \end{cases} \quad (17)$$

将 $n'$ 带入（5）式可得:

$$R'e_2=-e_2$$

因此可以得到R’的形式为:

$$R'=\begin{bmatrix}\cos\theta & 0 & \sin\theta \\ 0 & -1 & 0 \\ \sin\theta & 0 & -\cos\theta\end{bmatrix} \quad (21)$$

将（18）（19）带入（3）可得:

$$t'=(d_1+d_3)\begin{pmatrix}x_1\\0\\x_3\end{pmatrix} \quad (23)$$

将（17）和（18）带入（7）式第三个可得（18）式中的: $\sin\theta$ $\cos\theta$

$$d'\begin{bmatrix}\cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta\end{bmatrix}\begin{bmatrix}-x_3\\0\\x_1\end{bmatrix}=\begin{bmatrix}-d_1x_3\\0\\d_3x_1\end{bmatrix} \Rightarrow \begin{cases} \sin\theta=\dfrac{(d_1+d_3)}{d_2}x_1x_3=\varepsilon_1\varepsilon_3\dfrac{\sqrt{({d_1}^2-{d_2}^2)({d_2}^2-{d_3}^2)}}{(d_1-d_3)d_2} \\ \cos\theta=\dfrac{d_3{x_1}^2-d_1{x_3}^2}{d_2}=\dfrac{d_1d_3-{d_2}^2}{(d_1-d_3)d_2} \end{cases} \quad (22)$$

# 单目Initializer.cpp:

- **Homograph 矩阵分解**

对于（15）式这种情况，（14）情况的特例：

根据（8）式三个方程可解得：

$$n' = \begin{cases} x_1 = 0 \\ x_2 = 0 \\ x_3 = \pm 1 \end{cases}$$

将 $n'$ 带入（5）式可得：

$$R' = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

带入（3）式可得：

$$t' = (d_1 + d_3) n'$$

对于（16）式这种情况，（14）情况的特例：

根据公式（1）： $\Lambda = \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{bmatrix} = d'R' + t'n'^T$

根据 $d' = d_1 = d_2 = d_3$ 可得： $\begin{bmatrix} d' & 0 & 0 \\ 0 & d' & 0 \\ 0 & 0 & d' \end{bmatrix} = d'R' + t'n'^T \Rightarrow Id' = d'R' + t'n'^T$

取垂直于法向量 $n'$ 的向量 $x$ ，带入： $Id'x = d'R' + t'n'^T$

$\Rightarrow$ $Rx = -x$ 根据household变换： $R' = -I + 2n'n'^T$

$t' = -2d'n'$

# 单目Initializer.cpp:

- Fundamental 模型评分

$$scoreF = \sum_{i=0}^{N} \rho(T_F - \|x'Fx\|^2/\sigma^2)$$

$$\rho(x)\begin{cases}0 & x \le 0 \\ x & else\end{cases} \qquad T_H = 5.99$$

- Homograph 模型评分

$$scoreH = \sum_{i=0}^{N} \rho(T_H - \|x' - Hx\|^2/\sigma^2) + \rho(T_H - \|x - H^{-1}x'\|^2/\sigma^2)$$ 对称转移误差

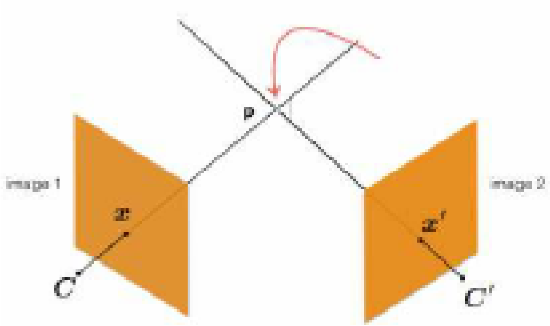
$$\rho(x)\begin{cases}0 & x \le 0 \\ x & else\end{cases} \qquad T_F = 3.84$$

- Homograph 模型与Fundamental模型选择

$$R_H = \frac{s_H}{s_H + s_F} \qquad R_H > 0.45$$

# 单目Initializer.cpp:

- 三角化恢复3D点




$(x,x')$ 为匹配特征点对

$(P,P')$ 分别为它们的投影矩阵

$\Rightarrow x = PX \quad x' = P'X$

$$\Rightarrow \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda \begin{bmatrix} p1 & p2 & p3 & p4 \\ p5 & p6 & p7 & p8 \\ p9 & p10 & p11 & p12 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

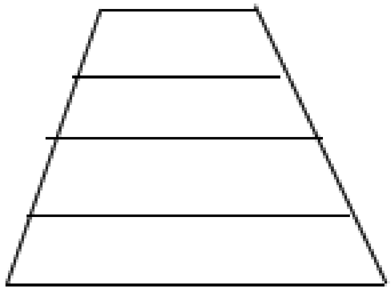
简写: $\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda \begin{bmatrix} - & P_0 & - \\ - & P_1 & - \\ - & P_2 & - \end{bmatrix} \begin{bmatrix} - \\ X \\ - \end{bmatrix}$

DLT求解: $\begin{bmatrix} vP_2 - P_1 \\ P_0 - uP_2 \\ uP_1 - vP_0 \end{bmatrix} X = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

一组匹配点: $\begin{bmatrix} vP_2 - P_1 \\ P_0 - uP_2 \\ v'P'_2 - P'_1 \\ P_0 - u'P'_2 \end{bmatrix} X = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

- 尺度与距离

Nearer



Farther

Level:n-1 --> dmin

Level:m --> d

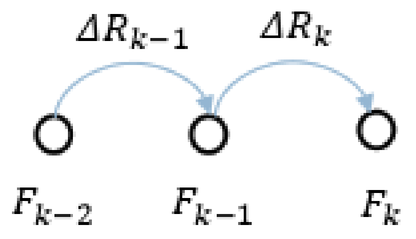
Level:0 --> dmax

d/dmin = 1.2^(n-1-m)

dmax/d = 1.2^m

# Tracking. cpp:

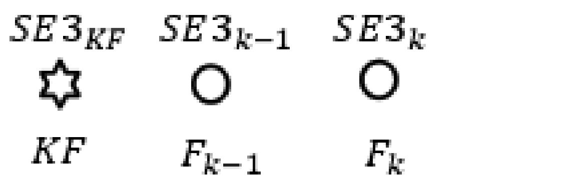
- **TrackWithMotionModel**




恒速模型： $\Delta R \approx \Delta R_{k-1}$

这里是不是可以引入IMU来测量相对旋转呢

- **TrackReferenceKeyFrame**




跟踪参考帧模型： $SE3_k \approx SE3_{KF}$

- **Relocalization**

EPnP求解：

世界坐标系下有N个3D点： $p_i^w \quad i = 1,\ldots,n$

选择四个控制点： $c_j^w \quad j = 1,\ldots,4$ 质心，三个主方向

对每个3D点，可以找到4个 $\alpha_j$ ，使得： $p_i^w = \sum_{j=1}^{4} \alpha_{ij} c_j^w$ ，且： $\sum_{j=1}^{4} \alpha_{ij} = 1 \qquad (1)$

对于同样的 $\alpha_j$ ，可以使得： $p_i^c = \sum_{j=1}^{4} \alpha_{ij} c_j^c \qquad (2)$

$p_i^c \quad c_j^c$ 为待求量

# Tracking. cpp:

根据投影模型： $\lambda_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = P \cdot \sum_{j=1}^{4} \alpha_{ij} c_j^c = \begin{bmatrix} f_u & 0 & u_c \\ 0 & f_v & v_c \\ 0 & 0 & 1 \end{bmatrix} \sum_{j=1}^{4} \alpha_{ij} \begin{bmatrix} x_j^c \\ y_j^c \\ z_j^c \end{bmatrix}$ (3)

展开可得： $\sum_{j=1}^{4} \alpha_{ij} f_u x_j^c + \alpha_{ij}(u_c - u_i) z_j^c = 0 \quad \sum_{j=1}^{4} \alpha_{ij} f_v y_j^c + \alpha_{ij}(v_c - v_i) z_j^c = 0$ (4)

将（4）写成矩阵形式： $Mx = 0$ 其中待求量为四个控制点： $x = [c_1^{cT}, c_2^{cT}, c_3^{cT}, c_4^{cT}]$ (5)

特征值分解M矩阵： $x = \sum_{i=1}^{N} \beta_i v_i$ (6)

由于公式5中待求量是四个控制点，共有12个未知数，由公式4可知，每对3D-2D对应点可以形成两个约束，故理论上来讲需要6组3D-2D对应点，可EPnP只需要至少4组即可，Why?

对于投影相机模型，公式（6）中N等于1，因为只有一个尺度变量；
对于正交相机模型，公式（6）中N等于4，因为每个参考点的深度变化后仍满足约束；
因此，当相机焦距比较小时，N为1。当相机焦距更大，相机接近于正交相机时，$M^T M$ 将有4个接近于0的特征值。

# Tracking. cpp:

四个参考点两两组合，可以得到6个距离，根据尺度不变性，可以得到如下约束：

N=1： $\|\beta v^{[i]} - \beta v^{[j]}\|^2 = \|c_i^w - c_j^w\|^2$

$$\Longrightarrow \quad \beta = \frac{\Sigma_{\{i,j\}\in[1;4]}\|v^{[i]} - v^{[j]}\| \cdot \|c_i^w - c_j^w\|}{\Sigma_{\{i,j\}\in[1;4]}\|v^{[i]} - v^{[j]}\|^2}$$

N=2： $\left\|\left(\beta_1 v_1^{[i]} + \beta_2 v_2^{[i]}\right) - \left(\beta_1 v_1^{[j]} + \beta_2 v_2^{[j]}\right)\right\|^2 = \|c_i^w - c_j^w\|^2$

………

Gauss-Newton优化：

$$Error(\beta) = \sum_{(i,j)\, s.t.\, i<j} \left(\|c_i^c - c_j^c\|^2 - \|c_i^w - c_j^w\|^2\right) \qquad c_i^c = \sum_{j=1}^{4} \beta_j v_j^{[i]}$$

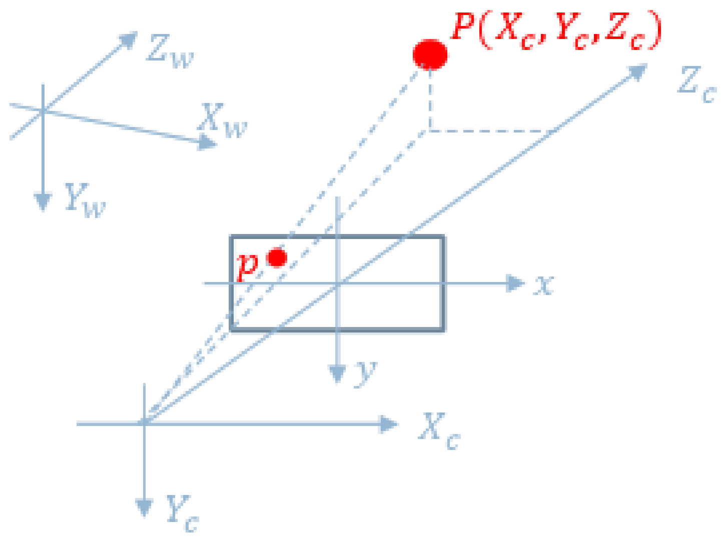
# ORBmatcher.cpp:

1、ORB-SLAM2中特征点匹配均采用了各种技巧减小特征点匹配范围。

2、ORB-SLAM2中特征点通过描述子匹配后会进行旋转一致性检测。并且最佳匹配特征点要明显优于次优匹配点。

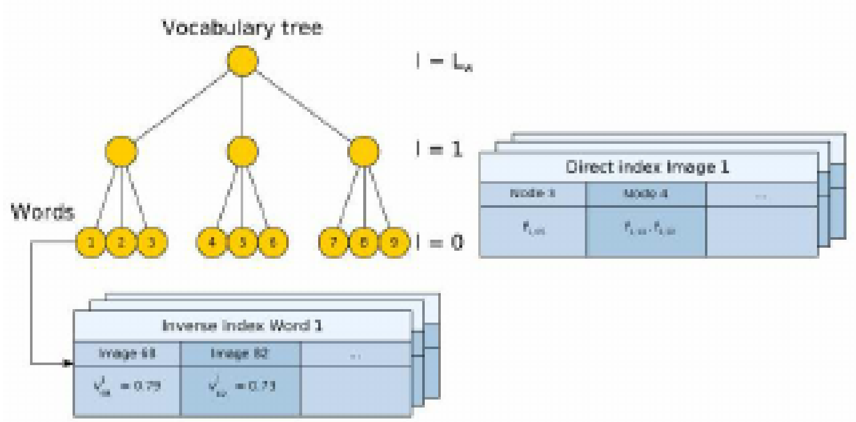
3、ORB-SLAM2中特征点提取仍然是非常耗时的地方。

- SearchByProjection与SearchBySim3




SearchByProjection函数和SearchBySim3函数利用将相机坐标系下的MapPoints投影到图像坐标系，在投影点附近搜索匹配

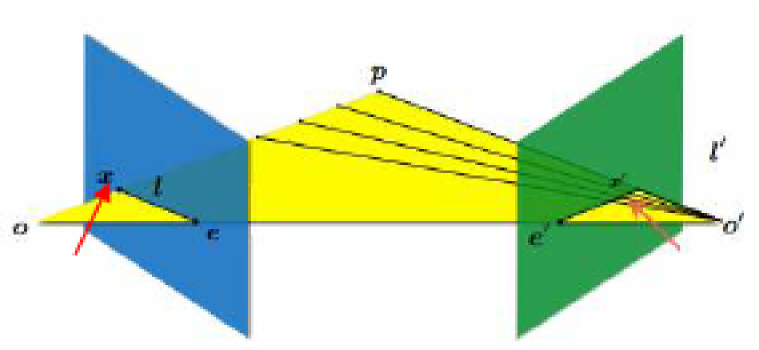
- SearchByBoW




SearchByBoW函数通过判断特征点对应的word的node是否相同可以加速匹配过程

# ORBmatcher.cpp:

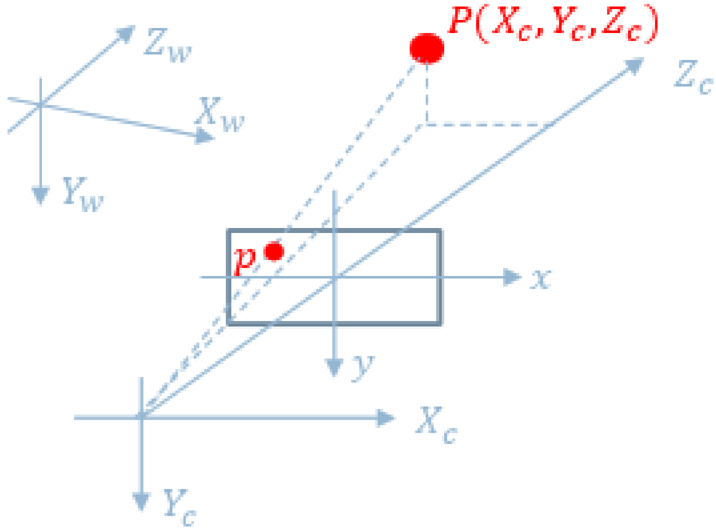
- **SearchForTriangulation**




SearchByProjection函数利用对极几何约束：左目一个点对应右目一条线。

将左图像的每个特征点与右图像同一node节点的所有特征点依次检测，判断是否满足对极几何约束，满足约束就是匹配的特征点

- **Fuse**




和SearchByProjection函数差不多，只不过是判断特征点p的MapPoint是否与MapPoint点P冲突

# Optimizer.cpp:

- **GlobalBundleAdjustemnt与LocalBundleAdjustment**

3D-2D 最小化重投影误差 e = (u,v) - project(Tcw*Pw)

Vertex: g2o::VertexSE3Expmap(), 即当前帧的Tcw

g2o::VertexSBAPointXYZ(), MapPoint的mWorldPos

Edge: g2o::EdgeSE3ProjectXYZ(), BaseBinaryEdge

Vertex: 待优化当前帧的Tcw

Vertex: 待优化MapPoint的mWorldPos

measurement: MapPoint在当前帧中的二维位置(u,v)

InfoMatrix: invSigma2(与特征点所在的尺度有关)

✓ Map中所有的MapPoints和关键帧做bundle adjustment优化

Global BA优化在ORB-SLAM2中有两个地方使用:

a.单目初始化: CreateInitialMapMonocular函数

b.闭环优化: RunGlobalBundleAdjustment函数

✓ LocalBundleAdjustment会在LocalMapping线程处理完队列中最后一个关键时进行

# Optimizer.cpp:

- **PoseOptimization**

3D-2D 最小化重投影误差 e = (u,v) - project(Tcw*Pw)

只优化Frame的Tcw，不优化MapPoints的坐标

Vertex: g2o::VertexSE3Expmap()，即当前帧的Tcw

Edge: g2o::EdgeSE3ProjectXYZOnlyPose()，BaseUnaryEdge

Vertex：待优化当前帧的Tcw

measurement：MapPoint在当前帧中的二维位置(u,v)

InfoMatrix: invSigma2(与特征点所在的尺度有关)

g2o::EdgeStereoSE3ProjectXYZOnlyPose()，BaseUnaryEdge

Vertex：待优化当前帧的Tcw

measurement：MapPoint在当前帧中的二维位置(ul,v,ur)

InfoMatrix: invSigma2(与特征点所在的尺度有关)

# Optimizer.cpp:

- **OptimizeEssentialGraph**

  Vertex: g2o::VertexSim3Expmap，Essential graph中关键帧的位姿

  Edge:　g2o::EdgeSim3()，BaseBinaryEdge

  　　Vertex：关键帧的Tcw，MapPoint的Pw

  　　measurement：经过CorrectLoop函数步骤2，Sim3传播校正后的位姿

  　　InfoMatrix：单位矩阵

  ✓ OptimizeEssentialGraph会在成功进行闭环检测后，全局BA优化前进行

# Optimizer.cpp:

- OptimizeSim3

Vertex: g2o::VertexSim3Expmap()，两个关键帧的位姿

g2o::VertexSBAPointXYZ()，两个关键帧共有的MapPoints

Edge: g2o::EdgeSim3ProjectXYZ()，BaseBinaryEdge

Vertex：关键帧的Sim3，MapPoint的Pw

measurement：MapPoint在关键帧中的二维位置(u,v)

InfoMatrix：invSigma2(与特征点所在的尺度有关)

g2o::EdgeInverseSim3ProjectXYZ()，BaseBinaryEdge

Vertex：关键帧的Sim3，MapPoint的Pw

measurement：MapPoint在关键帧中的二维位置(u,v)

InfoMatrix：invSigma2(与特征点所在的尺度有关)

✓ OptimizeSim3会在筛选闭环候选帧时用于位姿Sim3优化