

第9周：ORB_SLAM2 课程课件

本周课程重点：

1 共视图 本质图 拓展树原理

关键帧、共视图、扩展树、本质图的区别

共视图 (Covisibility Graph)

本质图 (Essential Graph)

扩展树 (spanning tree)

essential graph 和 full BA 结果对比

计算Sim3函数ComputeSim3

2 全网最详细的EPnP 算法原理详解

背景介绍

统一变量格式：

控制点如何选取？

计算控制点系数，用控制点重新表达数据

透视投影关系构建约束

求解

N=1的情况

N=2的情况

N=3的情况

N=4的情况

近似求 β 初始解

N=4的情况

N=3的情况

N=2的情况

高斯牛顿优化

ICP 求解位姿

总结

参考

第9周：ORB_SLAM2 课程课件

本课件是公众号 计算机视觉life 旗下课程 [《全网最详细的ORB-SLAM2精讲：原理推导+逐行代码分析》](#) (点击可跳转课程详情) 的课程课件。感谢各位学员的支持！

本课程对应的注释代码：https://github.com/electech6/ORBSLAM2_detailed_comments

由于源码注释和课件在持续更新，所以：

如视频课程中注释与上述GitHub中有不同，以GitHub上最新源码为准。

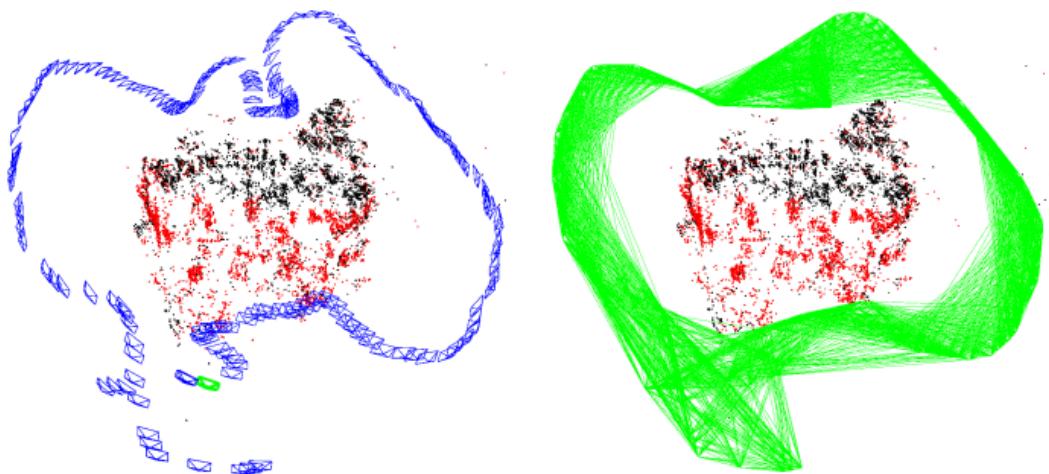
如视频课程中课件与本课件不同，以本课件为准。

本周课程重点：

1. 掌握关键帧、共视图、扩展树、本质图的原理（重要）。
2. 掌握本质图优化原理和代码（重要）。
3. 理解EPnP的原理（重要），能看懂推导过程。

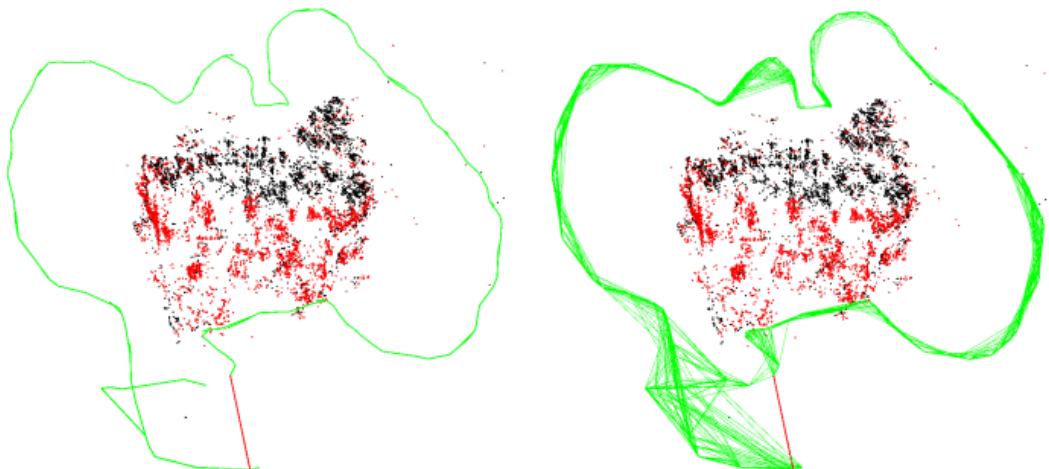
1 共视图 本质图 拓展树原理

关键帧、共视图、扩展树、本质图的区别



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

(b) Covisibility Graph

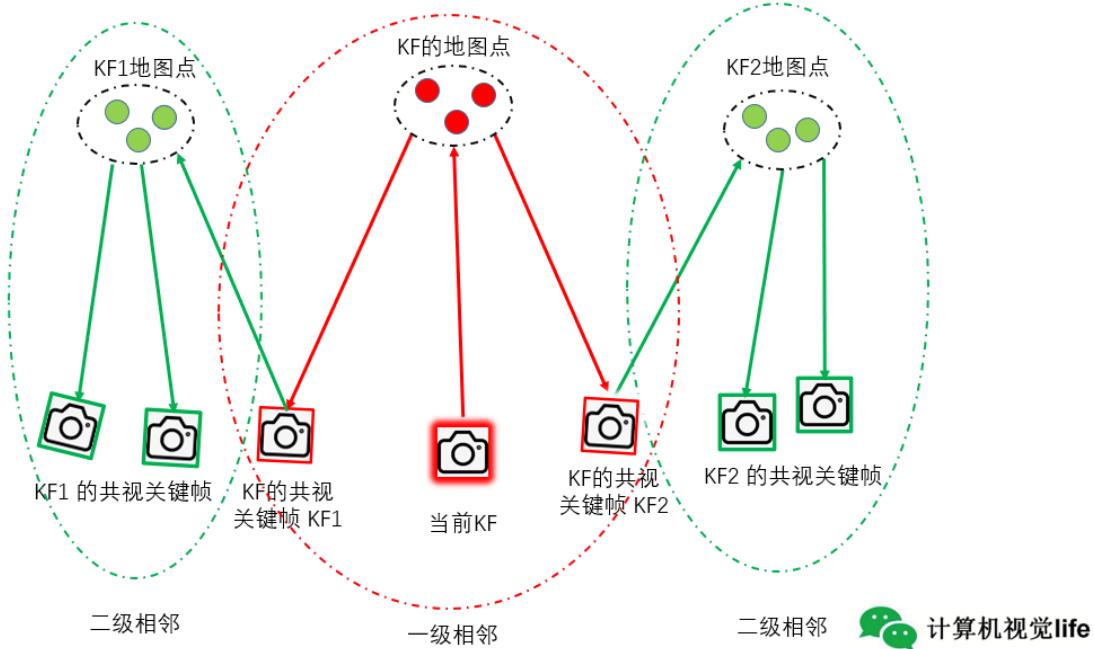


(c) Spanning Tree (green) and Loop Closure (red)

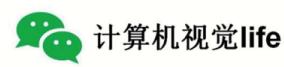
(d) Essential Graph

共视图 (Covisibility Graph)

共视图是无向加权图，每个节点是关键帧，如果两个关键帧之间满足一定的共视关系（至少15个共同观测地图点）他们就连成一条边，边的权重就是共视地图点数目



共视图的作用



- 跟踪局部地图，扩大搜索范围
 - Tracking::UpdateLocalKeyFrames()

- 局部建图里关键帧之间新建地图点
 - LocalMapping::CreateNewMapPoints()
 - LocalMapping::SearchInNeighbors()

- 闭环检测、重定位检测
 - LoopClosing::DetectLoop()、LoopClosing::CorrectLoop()
 - KeyFrameDatabase::DetectLoopCandidates
 - KeyFrameDatabase::DetectRelocalizationCandidates

- 优化
 - Optimizer::OptimizeEssentialGraph

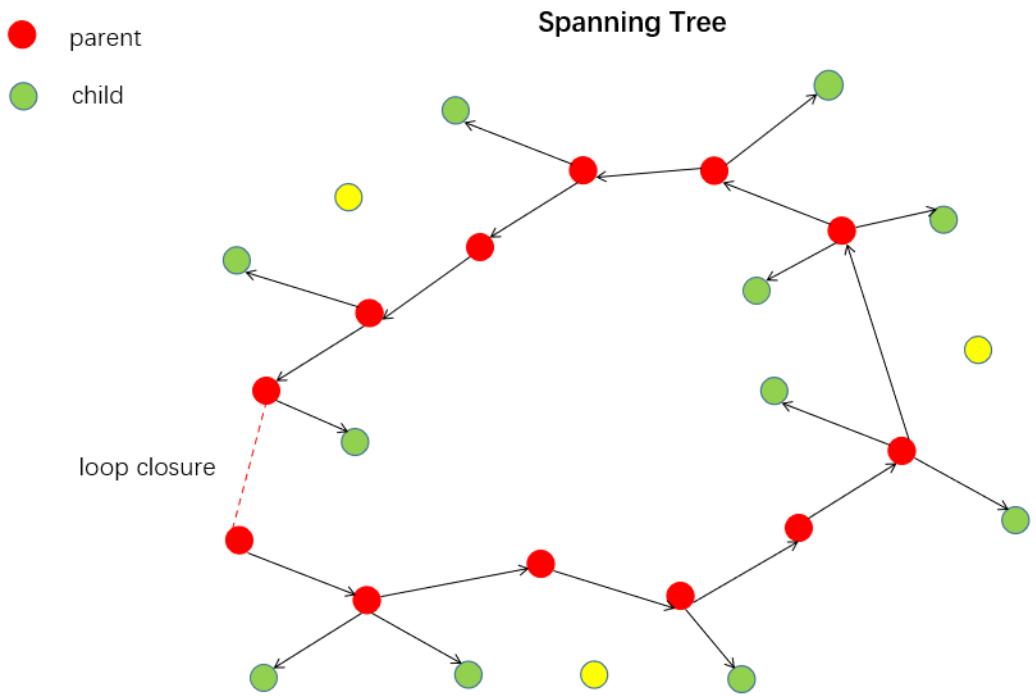
本质图 (Essential Graph)

共视图比较稠密，本质图比共视图更稀疏，这是因为本质图的作用是用在闭环矫正时，用相似变换来矫正尺度漂移，把闭环误差均摊在本质图中。本质图中节点也是所有关键帧，但是连接边更少，只保留了联系紧密的边来使得结果更精确。本质图中包含：

1. 扩展树连接关系
2. 形成闭环的连接关系，闭环后地图点变动后新增加的连接关系
3. 共视关系非常好（至少100个共视地图点）的连接关系

扩展树 (spanning tree)

子关键帧和父关键帧构成



essential graph 和 full BA 结果对比

从结果来看，

- 全局BA存在收敛问题。即使迭代100次，相对均方误差RMSE 也比较高
- essential graph 优化可以快速收敛并且结果更精确。 θ_{min} 表示被选为essential graph至少需要的共视地图点数目，从结果来看， θ_{min} 的大小对精度影响不大，但是较大的 θ_{min} 值可以显著减少运行时间
- essential graph 优化后增加全局 full BA 可以提升精度（但比较有限），但是会耗时较多

Table 4.6: Comparison of loop closing strategies in KITTI 09.

Method	Time (s)	Graph Edges	RMSE (m)
Without Loop Closure	-	-	48.77
Full BA (20 iterations)	14.64	-	49.90
Full BA (100 iterations)	72.16	-	18.82
Essential Graph ($\theta_{min} = 200$)	0.38	890	8.84
Essential Graph ($\theta_{min} = 100$)	0.48	1979	8.36
Essential Graph ($\theta_{min} = 50$)	0.59	3583	8.95
Essential Graph ($\theta_{min} = 15$)	0.94	6663	8.88
Essential Graph ($\theta_{min} = 100$) + full BA (20 iterations)	13.40	1979	7.22

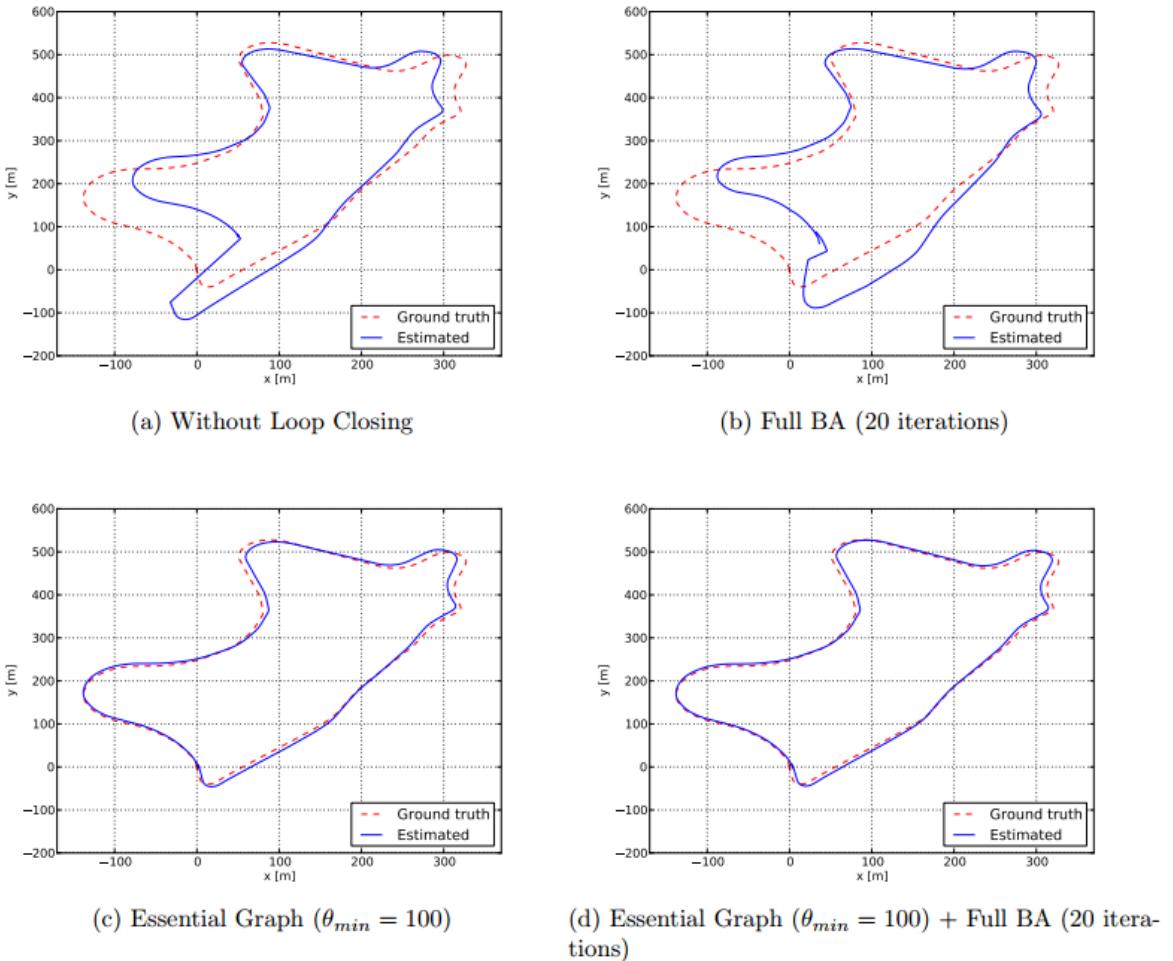
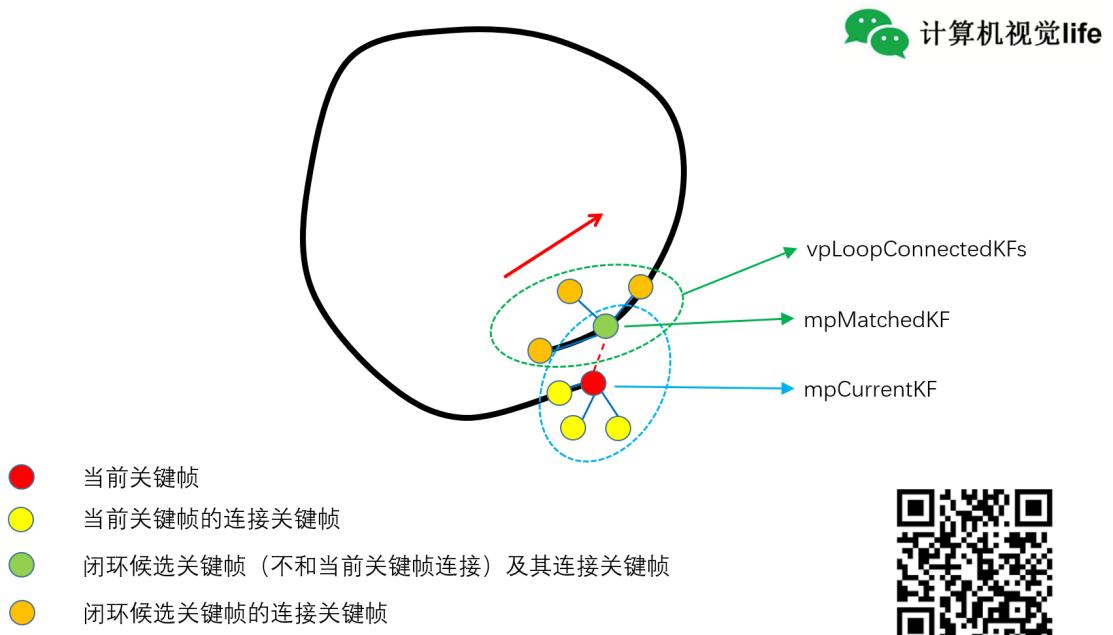


Figure 4.13: Comparison of different loop closing strategies in KITTI 09.

计算Sim3函数ComputeSim3

LoopClosing::ComputeSim3() 里查找 vpLoopConnectedKFs, mpLoopMapPoints



2 全网最详细的EPnP 算法原理详解

背景介绍

论文：

Lepetit V, Fua M N. EPnP: An Accurate $O(n)$ Solution to the PnP Problem[J]. International Journal of Computer Vision, 2009.

输入：

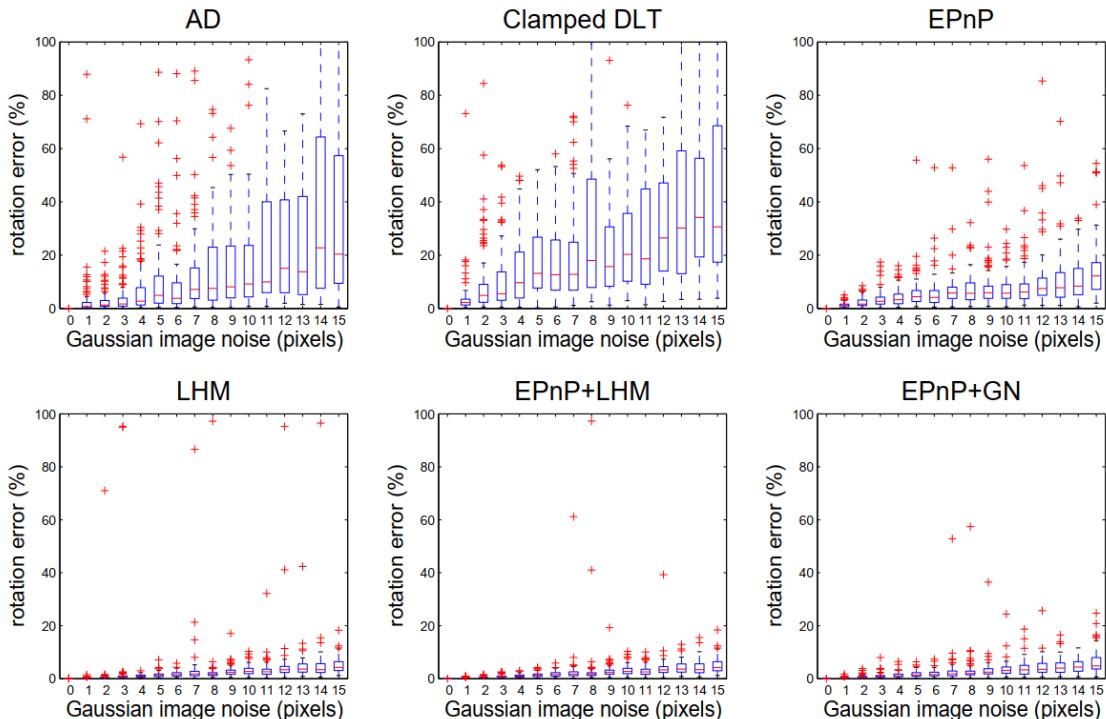
1. n 个世界坐标系下的3D点，论文中称为3D参考点
2. 这 n 个3D点投影在图像上的2D坐标
3. 相机内参矩阵 K ，包括焦距和主点

输出：相机的位姿 R, t

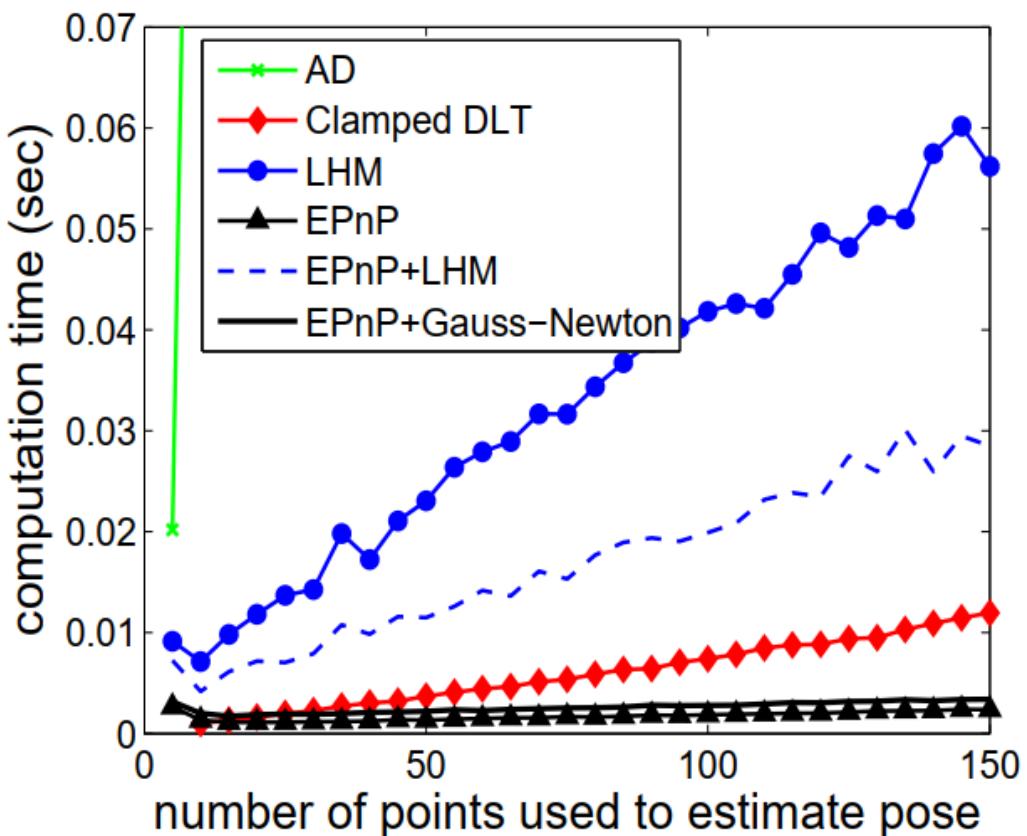
应用：特征点的图像跟踪，需要实时处理有噪声的特征点，对计算精度和效率要求比较高，只需4对匹配点即可求解。

算法的优点：

1. 只需要4对非共面点，对于平面只需要3对点
2. 闭式解，不需要迭代，不需要初始估计值。
3. 精度比较高。和迭代法里精度最高的方法LHM方法精度相当。



4. 比较鲁棒，可以处理带噪声的数据。迭代法受到初始估计的影响比较大，会不稳定
5. 线性计算复杂度为 $O(n)$



6. 平面和非平面都适用

原理和步骤：

我们目前知道 n 个世界坐标系下的3D点及其在图像上的2D投影点，还有相机内参，目的是为了求世界坐标系到相机坐标系下的位姿变换 R, t 。

EPnP的思路就是先把2D图像点通过内参变换到相机坐标系下的3D点，然后用ICP来求解3D-3D的变换就得到了位姿。那么问题的核心就转化为如何通过2D信息，加上一些约束，来得到相机坐标系下的3D点。因为我们这里的位姿变换是欧式空间下的刚体变换，所以点之间的相对距离信息在不同坐标系下是不变的。我们称之为刚体结构不变性。后面就是紧紧围绕这个特性来求解的。

- 首先我们对3D点的表达方式进行了新的定义。之前不管是世界坐标系还是相机坐标系下的3D点，它们都是相对于自己坐标系下的原点的。那么两个坐标系原点不同，坐标的量级可能差异非常大，比如相机坐标系下3D点坐标范围可能是10-100之间，世界坐标系下坐标可能是1000-10000之间，这对求解优化都是不利的。所以我们要统一一下量级。可以理解为归一化吧，这在求基础矩阵、单应矩阵时都是常规手段。

具体来说，我们对每个坐标系定义**4个控制点**，其中一个是质心（也就是各个方向均值），其他3个用PCA从三个主方向选取，这4个控制点可以认为是参考基准，类似于坐标系里的基。所有的3D点都表达为这4个参考点的线性组合。这些系数我们称之为权重，为了不改变数据的相对距离，权重必须为1。这样，我们就可以用**世界坐标系或相机坐标系下的4个控制点表示所有世界坐标系或相机坐标系下的3D点**。

- 利用投影方程将图像2D点恢复相机坐标系下3D点（未知量）。经过整理后，一组点对可以得到2个方程。我们待求的相机坐标系下3D点对应的4个控制点，每个控制点3个分量，总共12个未知数组成的一个向量。
- 用SVD分解可以求解上述向量，但是因为恢复的相机坐标系下3D点还有个**尺度因子 β** ，这里我们根据**结构信息不变性作为约束**，求解。
- 最后用高斯牛顿法优化上述求解的 β 。

统一变量格式：

首先统一下变量的定义格式：

用上标^w和^c分别表示在世界坐标系和相机坐标系中的坐标

n 个3D参考点在世界坐标系中的坐标是已知的，记为

$$\mathbf{p}_i^w, \quad i = 1, \dots, n$$

n 个3D参考点在相机坐标系下的坐标是未知的，记为

$$\mathbf{p}_i^c, \quad i = 1, \dots, n$$

n 个3D参考点在相机坐标系下对应的 n 个2D投影坐标是已知的，记为

$$\mathbf{u}_i, \quad i = 1, \dots, n$$

4个控制点在世界坐标系下的坐标为

$$\mathbf{c}_j^w, j = 1, \dots, 4$$

4个控制点在相机坐标系下的坐标是我们未知的，记为

$$\mathbf{c}_j^c, j = 1, \dots, 4$$

注意以上坐标都是非齐次坐标，后面也都是非齐次坐标。

4个控制点系数 $\alpha_{ij}, \quad i = 1, \dots, n, \quad j = 1, \dots, 4$ ，也就是论文中的homogeneous barycentric coordinates，我们翻译为齐次重心坐标。**同一3D点在世界坐标系下和相机坐标系下的控制点系数相同。** 后面会给出证明。

控制点如何选取？

理论上，控制点的坐标可以任意选取。但在实践中，作者发现了一种可以提高结果稳定性的控制点选择方法。具体如下

- 将参考点的质心（或者称为重心、均值中心）设置为其中一个控制点，表达式见下。这是有一定物理意义的，因为后续会使用质心对坐标点进行归一化。

$$\mathbf{c}_1^w = \frac{1}{n} \sum_{i=1}^n \mathbf{p}_i^w$$

- 剩下的3个控制点从数据的三个主方向上选取。

我们对世界坐标系下3D点集合 $\{\mathbf{p}_i^w, i = 1, \dots, n\}$ 去质心后得到

$$\mathbf{A} = \begin{bmatrix} (\mathbf{p}_1^w)^T - (\mathbf{c}_1^w)^T \\ \vdots \\ (\mathbf{p}_n^w)^T - (\mathbf{c}_1^w)^T \end{bmatrix}$$

\mathbf{A} 是一个 $n \times 3$ 的矩阵，那么 $\mathbf{A}^T \mathbf{A}$ 就是 3×3 方阵，通过对矩阵 $\mathbf{A}^T \mathbf{A}$ 进行特征值分解，得到三个特征值 $\lambda_1^w, \lambda_2^w, \lambda_3^w$ ，它们对应的特征向量为 $\mathbf{v}_1^w, \mathbf{v}_2^w, \mathbf{v}_3^w$

将剩余的3个控制点表示为

$$\mathbf{c}_2^w = \mathbf{c}_1^w + \sqrt{\frac{\lambda_1^w}{n}} v_1^w$$

$$\mathbf{c}_3^w = \mathbf{c}_1^w + \sqrt{\frac{\lambda_2^w}{n}} v_2^w$$

$$\mathbf{c}_4^w = \mathbf{c}_1^w + \sqrt{\frac{\lambda_3^w}{n}} v_3^w$$

为什么要加 \mathbf{c}_1^w ?

因为之前去了质心，现在要重新加上。

为何是 $\sqrt{\frac{\lambda_i^w}{n}}$ 的形式？

特征分解操作的矩阵为 $A^T A$ ，点的距离被作了平方，故要对特征值开方，特征值是由 n 个地图共同解出来的权重 $/n$

计算控制点系数，用控制点重新表达数据

我们将世界坐标系下3D点的坐标表示为对应控制点坐标的线性组合：

$$\mathbf{p}_i^w = \sum_{j=1}^4 \alpha_{ij} \mathbf{c}_j^w, \quad \sum_{j=1}^4 \alpha_{ij} = 1$$

在论文中， α_{ij} 称为 *homogeneous barycentric coordinates*，我们翻译为齐次重心坐标，它实际上表达的是世界坐标系下3D点在控制点坐标系下的坐标系数。当控制点 \mathbf{c}_j^w 通过第一步的方法确定后， α_{ij} 也是唯一确定的。我们来推导一下，上式展开后得到

$$\mathbf{p}_i^w = \alpha_{i1} \mathbf{c}_1^w + \alpha_{i2} \mathbf{c}_2^w + \alpha_{i3} \mathbf{c}_3^w + \alpha_{i4} \mathbf{c}_4^w$$

3D点的重心为 \mathbf{c}_1^w ，也是我们的第一个控制点。上式左右分别减去重心

$$\begin{aligned} \mathbf{p}_i^w - \mathbf{c}_1^w &= \alpha_{i1} \mathbf{c}_1^w + \alpha_{i2} \mathbf{c}_2^w + \alpha_{i3} \mathbf{c}_3^w + \alpha_{i4} \mathbf{c}_4^w - \mathbf{c}_1^w \\ &= \alpha_{i1} \mathbf{c}_1^w + \alpha_{i2} \mathbf{c}_2^w + \alpha_{i3} \mathbf{c}_3^w + \alpha_{i4} \mathbf{c}_4^w - (\alpha_{i1} + \alpha_{i2} + \alpha_{i3} + \alpha_{i4}) \mathbf{c}_1^w \\ &= \alpha_{i2} (\mathbf{c}_2^w - \mathbf{c}_1^w) + \alpha_{i3} (\mathbf{c}_3^w - \mathbf{c}_1^w) + \alpha_{i4} (\mathbf{c}_4^w - \mathbf{c}_1^w) \\ &= [\mathbf{c}_2^w - \mathbf{c}_1^w \quad \mathbf{c}_3^w - \mathbf{c}_1^w \quad \mathbf{c}_4^w - \mathbf{c}_1^w] \begin{bmatrix} \alpha_{i2} \\ \alpha_{i3} \\ \alpha_{i4} \end{bmatrix} \end{aligned}$$

→ 左乘 = 列向量的线性组合
各是一个列向量

$$\begin{bmatrix} \alpha_{i2} \\ \alpha_{i3} \\ \alpha_{i4} \end{bmatrix} = [\mathbf{c}_2^w - \mathbf{c}_1^w \quad \mathbf{c}_3^w - \mathbf{c}_1^w \quad \mathbf{c}_4^w - \mathbf{c}_1^w]^{-1} (\mathbf{p}_i^w - \mathbf{c}_1^w)$$

$$\alpha_{i1} = 1 - \alpha_{i2} - \alpha_{i3} - \alpha_{i4}$$

以上是世界坐标系下的推导，那么，在相机坐标系下， α_{ij} 满足如下的对应关系吗？

$$\mathbf{p}_i^c = \sum_{j=1}^4 \alpha_{ij} \mathbf{c}_j^c, \quad \sum_{j=1}^4 \alpha_{ij} = 1$$

这里先给出结论：同一个3D点在世界坐标系下对应控制点的系数 α_{ij} 和其在相机坐标系下对应控制点的系数相同。也就是说，我们可以预先在世界坐标系下求取控制点系数 α_{ij} ，然后将其作为已知量拿到相机坐标系下使用。

口说无凭，我们来推导一下，假设待求的相机位姿为 T ，那么

$$\begin{aligned}
\begin{bmatrix} \mathbf{p}_i^c \\ 1 \end{bmatrix} &= T \begin{bmatrix} \mathbf{p}_i^w \\ 1 \end{bmatrix} \\
&= T \begin{bmatrix} \sum_{j=1}^4 \alpha_{ij} \mathbf{c}_j^w \\ \sum_{j=1}^4 \alpha_{ij} \end{bmatrix} \\
&= \sum_{j=1}^4 \alpha_{ij} T \begin{bmatrix} \mathbf{c}_j^w \\ 1 \end{bmatrix} \\
&= \sum_{j=1}^4 \alpha_{ij} \begin{bmatrix} \mathbf{c}_j^c \\ 1 \end{bmatrix}
\end{aligned}$$

所以以下结论成立

$$\mathbf{p}_i^c = \sum_{j=1}^4 \alpha_{ij} \mathbf{c}_j^c$$

以上推导使用了对权重 α_{ij} 的重要约束条件 $\sum_{j=1}^4 \alpha_{ij} = 1$, 如果没有该约束, 那么上述结论不成立。

到目前为止, 我们已经根据世界坐标系下3D点 \mathbf{p}_i^w 求出了世界坐标系下的4个控制点 $\mathbf{c}_j^w, j = 1, \dots, 4$, 以及每个3D点对应的控制点系数 α_{ij} , 前面说过, 同一个3D点在世界坐标系下对应控制点的系数 α_{ij} 和其在相机坐标系下对应控制点的系数相同。所以如果我们能把4个控制点在相机坐标系下的坐标 $\mathbf{c}_j^c, j = 1, \dots, 4$ 求出来, 就可以得到世界坐标系下3D点在相机坐标系下的坐标 $\mathbf{c}_j^c, j = 1, \dots, 4$ 了。就可以根据ICP求解位姿了。

透视投影关系构建约束

记 w_i 为投影尺度系数, K 为相机内参矩阵, \mathbf{u}_i 为相机坐标系下3D参考点 \mathbf{p}_i^c 对应的2D投影坐标, 根据相机投影原理可得

$$w_i \begin{bmatrix} \mathbf{u}_i \\ 1 \end{bmatrix} = \mathbf{K} \mathbf{p}_i^c = \mathbf{K} \sum_{j=1}^4 \alpha_{ij} \mathbf{c}_j^c$$

记控制点 \mathbf{c}_j^c 坐标为 $[x_j^c, y_j^c, z_j^c]^T$, f_u, f_v 是焦距, u_c, v_c 是主点坐标, 上式可以化为

$$w_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_c \\ 0 & f_v & v_c \\ 0 & 0 & 1 \end{bmatrix} \sum_{j=1}^4 \alpha_{ij} \begin{bmatrix} x_j^c \\ y_j^c \\ z_j^c \end{bmatrix}$$

根据最后一行可以推出

$$w_i = \sum_{j=1}^4 \alpha_{ij} z_j^c, \quad i = 1, \dots, n$$

消去最后一行, 我们把上面矩阵展开写成等式右边为0的表达式, 所以实际上每个点对可以得到2个方程

$$\begin{aligned}
\sum_{j=1}^4 \alpha_{ij} f_u x_j^c + \alpha_{ij} (u_c - u_i) z_j^c &= 0 \\
\sum_{j=1}^4 \alpha_{ij} f_v y_j^c + \alpha_{ij} (v_c - v_i) z_j^c &= 0
\end{aligned}$$

把 = 左边的 $w_i w_i$ 移到右边

这里的待求的未知数是12个相机坐标系下控制点坐标 $\{(x_j^c, y_j^c, z_j^c)\}, j = 1, \dots, 4$, 我们把 n 个匹配点对全部展开, 再写成矩阵的形式:

$$\begin{bmatrix} \alpha_{11}f_u & 0 & \alpha_{11}(u_c - u_1) & \cdots & \alpha_{14}f_u & 0 & \alpha_{14}(u_c - u_1) \\ 0 & \alpha_{11}f_v & \alpha_{11}(v_c - v_1) & \cdots & 0 & \alpha_{14}f_v & \alpha_{14}(v_c - v_1) \\ \vdots & & & & & & \\ \alpha_{i1}f_u & 0 & \alpha_{i1}(u_c - u_i) & \cdots & \alpha_{i4}f_u & 0 & \alpha_{i4}(u_c - u_i) \\ 0 & \alpha_{i1}f_v & \alpha_{i1}(v_c - v_i) & \cdots & 0 & \alpha_{i4}f_v & \alpha_{i4}(v_c - v_i) \\ \vdots & & & & & & \\ \alpha_{n1}f_u & 0 & \alpha_{n1}(u_c - u_n) & \cdots & \alpha_{n4}f_u & 0 & \alpha_{n4}(u_c - u_n) \\ 0 & \alpha_{n1}f_v & \alpha_{n1}(v_c - v_n) & \cdots & 0 & \alpha_{n4}f_v & \alpha_{n4}(v_c - v_n) \end{bmatrix} \begin{bmatrix} x_1^c \\ y_1^c \\ z_1^c \\ x_2^c \\ y_2^c \\ z_2^c \\ x_3^c \\ y_3^c \\ z_3^c \\ x_4^c \\ y_4^c \\ z_4^c \end{bmatrix} = 0$$

其中, $i = 1, \dots, n$ 表示点对的数目, 我们记左边第1个矩阵为 \mathbf{M} , 它的大小为 $2n \times 12$, 第2个矩阵 \mathbf{x} 是待求的未知量组成的矩阵, 它的大小为 12×1 , 则上式可以写为

$$\mathbf{M}\mathbf{x} = \mathbf{0}$$

求解

满足 $\mathbf{M}\mathbf{x} = \mathbf{0}$ 的所有的解 \mathbf{x} 的集合就是 \mathbf{M} 的零空间。零空间 (null space), 有时也称为核 (kernel)。

$$\mathbf{x} = \sum_{i=1}^N \beta_i \mathbf{v}_i$$

正交矩阵不会改变模 (1-Norm)

$$(U\Sigma V) \mathbf{x} = \mathbf{0}$$

最小的特征值, 及其对应的向量

特征值和特征向量分别是特征分解时的
大小和方向

其中 \mathbf{v}_i 是 \mathbf{M} 的零奇异值对应的右奇异向量, 它的维度为 12×1 。

具体求解方法是通过构建 $\mathbf{M}^T \mathbf{M}$ 组成方阵, 求解其特征值和特征向量, 特征值为0的特征向量即为 \mathbf{v}_i 。这里需要说明的是, 不论有多少点对, $\mathbf{M}^T \mathbf{M}$ 的大小永远是 12×12 , 因此计算复杂度是 $O(n)$ 的。

EPnP中选取了4组点对, 原因:

6点法就没有自由度(从随机性角度, 行满秩的概率为1)去做优化了, orb 用的4点, 这样解空间的维度至少为4(12-8, 假设各行线性无关), 用优化可以进一步提升精度。

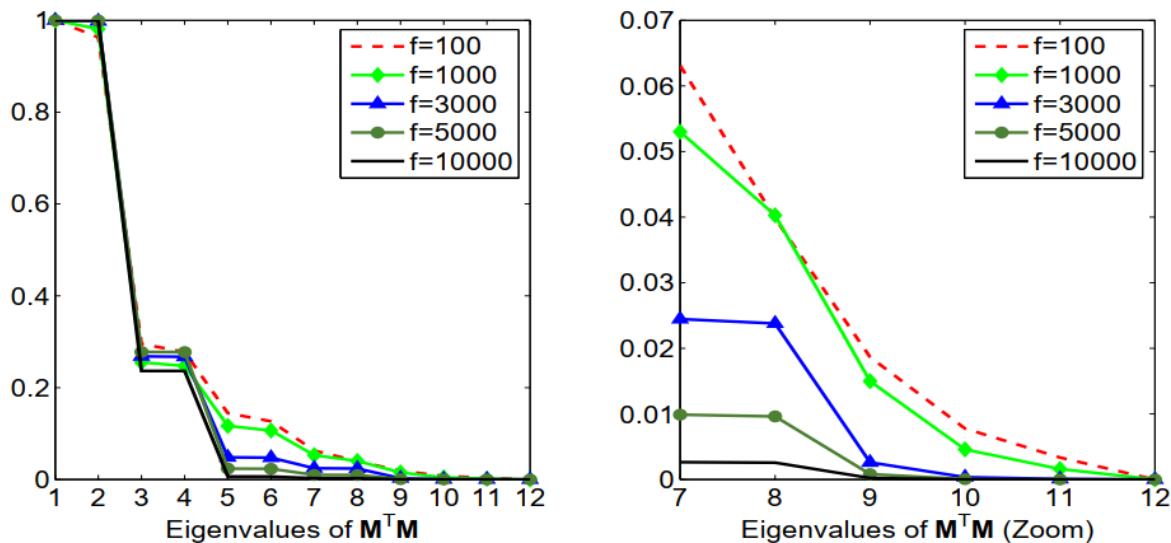
因为每个点对可以得到2个约束方程, 总共有12个未知数, 所以如果有6组点对, 我们就能直接求解, 此时 $N = 1$ 。如果相机的焦距逐渐增大, 相机模型更趋近于使用正交相机代替透视相机, 零空间的自由度就会增加到 $N = 4$ 。

我们来看论文中这张图:

横坐标表示通过 $\mathbf{M}^T \mathbf{M}$ 特征值分解得到的12个特征值的序号, 纵坐标表示对应特征值的大小。

当焦距 $f = 100$ 时, 我们看局部放大的右图, 只有1个特征值是0, 所以只要用最后一个特征向量就可以了。

当焦距 $f = 10000$ 时, 右图中可以看到第9, 10, 11, 12个特征值都是0, 也就是说只用最后一个特征向量是没有办法表示的, 要用到最后4个特征值对应的特征向量加权才行, 这就是最大 $N = 4$ 的来源。



这样，实际上 N 的取值范围是 $N = 1, 2, 3, 4$ ，ORB-SLAM 中的方法是：四种情况我们都试一遍，找出其中使得重投影误差最小的那组解作为最佳的解。

接下来就是如何求 $\{\beta_i\}, i = 1, \dots, N$ 。

这里就用到我们前面说的刚体结构不变性，就是不同坐标系下两个点的相对距离是恒定的，也就是

$$\|c_i^c - c_j^c\|^2 = \|c_i^w - c_j^w\|^2$$

后面会用到这个约束条件。

相机坐标系下两个控制点距离

下面分别讨论：

二 世界坐标系下它们的距离

N=1的情况

$\mathbf{x} = \beta \mathbf{v}$ ，未知数只有1个。记 $\mathbf{v}^{[i]}$ 是 3×1 的列向量，表示 \mathbf{v} （大小为 12×1 ）的第 i 个控制点 \mathbf{c}_i^c 所占据的3个元素组成的子向量，例如 $\mathbf{v}^{[1]} = [x_1^c, y_1^c, z_1^c]^T$ 代表 \mathbf{v} 的前3个元素，代入上述约束公式

$$\|\beta \mathbf{v}^{[i]} - \beta \mathbf{v}^{[j]}\|^2 = \|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2$$

4个控制点可以得到 β 的一个闭式解

$$\beta = \frac{\sum_{\{i,j\} \in [1:4]} \|\mathbf{v}^{[i]} - \mathbf{v}^{[j]}\| \cdot \|\mathbf{c}_i^w - \mathbf{c}_j^w\|}{\sum_{\{i,j\} \in [1:4]} \|\mathbf{v}^{[i]} - \mathbf{v}^{[j]}\|^2}$$

其中 $[i, j] \in [1 : 4]$ 表示 i, j 可以从1到4之间任意取值，也就是从4个值中任意取2个，有 $C_4^2 = 6$ 种取值。

N=2的情况

此时， $\mathbf{x} = \beta_1 \mathbf{v}_1 + \beta_2 \mathbf{v}_2$ ，带入到刚体结构不变性的约束方程

$$\left\| \left(\beta_1 \mathbf{v}_1^{[i]} + \beta_2 \mathbf{v}_2^{[i]} \right) - \left(\beta_1 \mathbf{v}_1^{[j]} + \beta_2 \mathbf{v}_2^{[j]} \right) \right\|^2 = \|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2$$

$$\|\beta_1 (\mathbf{v}_1^{[i]} - \mathbf{v}_1^{[j]}) + \beta_2 (\mathbf{v}_2^{[i]} - \mathbf{v}_2^{[j]})\|^2$$

展开

$$\|\beta_1 (\mathbf{v}_1^{[i]} - \mathbf{v}_1^{[j]}) + \beta_2 (\mathbf{v}_2^{[i]} - \mathbf{v}_2^{[j]})\|^2 = \|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2$$

$$\begin{bmatrix} (\mathbf{v}_1^{[i]} - \mathbf{v}_1^{[j]})^2 & (\mathbf{v}_1^{[i]} - \mathbf{v}_1^{[j]}) (\mathbf{v}_2^{[i]} - \mathbf{v}_2^{[j]}) & (\mathbf{v}_2^{[i]} - \mathbf{v}_2^{[j]})^2 \end{bmatrix} \begin{bmatrix} \beta_{11} \\ \beta_{12} \\ \beta_{22} \end{bmatrix} = \|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2$$

其中引入了三个中间变量如下：

$$\beta_{11} = \beta_1^2, \beta_{22} = \beta_2^2, \beta_{12} = \beta_1 \beta_2$$

上式就变成了线性方程。总共3个未知数。根据前面描述，4个控制点可以组合构造出6个线性方程，组成

$$\mathbf{L}\boldsymbol{\beta} = \boldsymbol{\rho}$$

其中， $\boldsymbol{\beta} = [\beta_{11}, \beta_{12}, \beta_{22}]^T$ ， \mathbf{L} 大小为 6×3 ， $\boldsymbol{\beta}$ 大小为 3×1 ， $\boldsymbol{\rho}$ 大小为 6×1 。解出 $\boldsymbol{\beta}$ 后，可以~~获得两组 β_1, β_2 的解。再加上一个条件，控制点在摄像机的前端，即 \mathbf{c}_j^c 的 z 分量要大于0，从而 β_1, β_2 唯一确定。~~

N=3的情况

与 $N = 2$ 的解法相同。此时， $\mathbf{x} = \beta_1 \mathbf{v}_1 + \beta_2 \mathbf{v}_2 + \beta_3 \mathbf{v}_3$ ，带入到刚体结构不变性的约束方程

$$\left\| \left(\beta_1 \mathbf{v}_1^{[i]} + \beta_2 \mathbf{v}_2^{[i]} + \beta_3 \mathbf{v}_3^{[i]} \right) - \left(\beta_1 \mathbf{v}_1^{[j]} + \beta_2 \mathbf{v}_2^{[j]} + \beta_3 \mathbf{v}_3^{[j]} \right) \right\|^2 = \|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2$$

这里的 $\boldsymbol{\beta} = [\beta_{11}, \beta_{12}, \beta_{13}, \beta_{22}, \beta_{23}, \beta_{33}]^T$ ，大小为 6×1 ，表示待求解未知数个数为6个， \mathbf{L} 的大小为 6×6 。

N=4的情况

我们着重来推导 $N = 4$ 的情况，因为在ORB-SLAM里面 N 是不知道的，代码里实际就是直接采用 $N = 4$ 的情况进行计算。此时， $\mathbf{x} = \beta_1 \mathbf{v}_1 + \beta_2 \mathbf{v}_2 + \beta_3 \mathbf{v}_3 + \beta_4 \mathbf{v}_4$ ，带入到刚体结构不变性的约束方程

$$\left\| \left(\beta_1 \mathbf{v}_1^{[i]} + \beta_2 \mathbf{v}_2^{[i]} + \beta_3 \mathbf{v}_3^{[i]} + \beta_4 \mathbf{v}_4^{[i]} \right) - \left(\beta_1 \mathbf{v}_1^{[j]} + \beta_2 \mathbf{v}_2^{[j]} + \beta_3 \mathbf{v}_3^{[j]} + \beta_4 \mathbf{v}_4^{[j]} \right) \right\|^2 = \|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2$$

注意上述 $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4$ 均为大小为 12×1 的特征向量，对应 $\mathbf{M}^T \mathbf{M}$ 最后4个零特征值。以特征向量 \mathbf{v}_1 为例， $\mathbf{v}_1^{[i]}, \mathbf{v}_1^{[j]}, [i, j] \in [1 : 4]$ 是上述特征向量 \mathbf{v}_1 拆成的4个大小为 3×1 的向量， $\mathbf{v}_1^{[i]}, \mathbf{v}_1^{[j]}$ 一共有6种不同的组合方式 $[\mathbf{v}_1^{[1]}, \mathbf{v}_1^{[2]}], [\mathbf{v}_1^{[1]}, \mathbf{v}_1^{[3]}], [\mathbf{v}_1^{[1]}, \mathbf{v}_1^{[4]}], [\mathbf{v}_1^{[2]}, \mathbf{v}_1^{[3]}], [\mathbf{v}_1^{[2]}, \mathbf{v}_1^{[4]}], [\mathbf{v}_1^{[3]}, \mathbf{v}_1^{[4]}]$ 。

等式左右互换进行化简：

等式右侧即 $\mathbf{L}\boldsymbol{\beta}$ 是相机坐标系下6种组合的控制点距离差

$$\begin{aligned}
 \|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2 &= \left\| \beta_1(\mathbf{v}_1^{[i]} - \mathbf{v}_1^{[j]}) + \beta_2(\mathbf{v}_2^{[i]} - \mathbf{v}_2^{[j]}) + \beta_3(\mathbf{v}_3^{[i]} - \mathbf{v}_3^{[j]}) + \beta_4(\mathbf{v}_4^{[i]} - \mathbf{v}_4^{[j]}) \right\|^2 \\
 &= \left\| \beta_1 \cdot \mathbf{d}\mathbf{v}_{1,[i,j]} + \beta_2 \cdot \mathbf{d}\mathbf{v}_{2,[i,j]} + \beta_3 \cdot \mathbf{d}\mathbf{v}_{3,[i,j]} + \beta_4 \cdot \mathbf{d}\mathbf{v}_{4,[i,j]} \right\|^2 \\
 &= \beta_1^2 \cdot \mathbf{d}\mathbf{v}_{1,[i,j]}^2 + 2\beta_1\beta_2 \cdot \mathbf{d}\mathbf{v}_{1,[i,j]} \cdot \mathbf{d}\mathbf{v}_{2,[i,j]} + \beta_2^2 \cdot \mathbf{d}\mathbf{v}_{2,[i,j]}^2 + 2\beta_1\beta_3 \cdot \mathbf{d}\mathbf{v}_{1,[i,j]} \cdot \mathbf{d}\mathbf{v}_{3,[i,j]} \\
 &\quad + 2\beta_2\beta_3 \cdot \mathbf{d}\mathbf{v}_{2,[i,j]} \cdot \mathbf{d}\mathbf{v}_{3,[i,j]} + \beta_3^2 \cdot \mathbf{d}\mathbf{v}_{3,[i,j]}^2 + 2\beta_1\beta_4 \cdot \mathbf{d}\mathbf{v}_{1,[i,j]} \cdot \mathbf{d}\mathbf{v}_{4,[i,j]} \\
 &\quad + 2\beta_2\beta_4 \cdot \mathbf{d}\mathbf{v}_{2,[i,j]} \cdot \mathbf{d}\mathbf{v}_{4,[i,j]} + 2\beta_3\beta_4 \cdot \mathbf{d}\mathbf{v}_{3,[i,j]} \cdot \mathbf{d}\mathbf{v}_{4,[i,j]} + \beta_4^2 \cdot \mathbf{d}\mathbf{v}_{4,[i,j]}^2
 \end{aligned}$$

上式等式左边记为 $\boldsymbol{\rho}_{6 \times 1}$ ，右边记为两个矩阵 $\mathbf{L}_{6 \times 10} \cdot \boldsymbol{\beta}_{10 \times 1}$ 相乘，下角标表示矩阵的维度， $[i, j] \in [1 : 4]$ ，于是我们可以得到如下结论：

四个控制点 $C_4^2 = 6$ 有6个组合

$$\begin{aligned}
 \mathbf{L}_{6 \times 10} &= [\mathbf{d}\mathbf{v}_1^2, 2\mathbf{d}\mathbf{v}_1 \cdot \mathbf{d}\mathbf{v}_2, \mathbf{d}\mathbf{v}_2^2, 2\mathbf{d}\mathbf{v}_1 \cdot \mathbf{d}\mathbf{v}_3, 2\mathbf{d}\mathbf{v}_2 \cdot \mathbf{d}\mathbf{v}_3, \mathbf{d}\mathbf{v}_3^2, 2\mathbf{d}\mathbf{v}_1 \cdot \mathbf{d}\mathbf{v}_4, 2\mathbf{d}\mathbf{v}_2 \cdot \mathbf{d}\mathbf{v}_4, 2\mathbf{d}\mathbf{v}_3 \cdot \mathbf{d}\mathbf{v}_4, \mathbf{d}\mathbf{v}_4^2]_{[i,j] \in [1:4]} \\
 \boldsymbol{\beta}_{10 \times 1} &= [\beta_1^2, \beta_1\beta_2, \beta_2^2, \beta_1\beta_3, \beta_2\beta_3, \beta_3^2, \beta_1\beta_4, \beta_2\beta_4, \beta_3\beta_4, \beta_4^2]^T \\
 \mathbf{L}_{6 \times 10} \cdot \boldsymbol{\beta}_{10 \times 1} &= \boldsymbol{\rho}_{6 \times 1}
 \end{aligned}$$

以上 $\mathbf{L}_{6 \times 10}$ 和 $\boldsymbol{\rho}_{6 \times 1}$ 是已知的，待求解的是 $\boldsymbol{\beta}_{10 \times 1}$ 。

正常来说上面我们可以用SVD求解，但是这样存在问题：

1. 10个未知数，但只有6个方程，未知数个数超过了方程数目。
2. 同时求解这么多参数都是独立的，比如求解出来的第1,3个参数对应的 β_1 和 β_2 不一定和求解出来的第2个参数 $\beta_1\beta_2$ 相等。所以即使求出来也很难确定最终的4个 β 值。

在ORB-SLAM2代码里作者使用了一种方法：先求初始解，然后再优化得到最优解。

近似求 β 初始解

我们来介绍ORB-SLAM2代码里的实现方法。

因为我们刚开始只要求粗糙的初始解即可，所以我们可以暴力的把 $\beta_{10 \times 1}$ 中某些项置为0。

下面分 N 取不同值来讨论：

△ N=4的情况

此时待求量为： $\beta_1, \beta_2, \beta_3, \beta_4$ 。我们取 $\beta_{10 \times 1}$ 中的第1,2,4,7个元素（不是必须这样取，这里只是源码中使用的一种方法），总共4个得到如下

$$\beta_{4 \times 1} = [\beta_1^2, \beta_1\beta_2, \beta_1\beta_3, \beta_1\beta_4]^T \quad \text{只留这4个，其他置0。}$$

当然对应的 $\mathbf{L}_{6 \times 10}$ 矩阵中每行也取第1,2,4,7个对应元素得到 $\mathbf{L}_{6 \times 4}$ ，而 $\rho_{6 \times 1}$ 不变。这样我们只要用SVD求解规模更小的矩阵就行了。

$$\mathbf{L}_{6 \times 4} \cdot \beta_{4 \times 1} = \rho_{6 \times 1}$$

不是0，6排各是世界坐标下
两个控制点的距离

最后得到

$$\beta_1 = \sqrt{\beta_1^2}, \quad \beta_2 = \frac{\beta_1\beta_2}{\beta_1}, \quad \beta_3 = \frac{\beta_1\beta_3}{\beta_1}, \quad \beta_4 = \frac{\beta_1\beta_4}{\beta_1}$$

N=3的情况

此时待求量为： $\beta_1, \beta_2, \beta_3$ 。我们取 $\beta_{10 \times 1}$ 中的第1,2,3,4,5个元素，总共5个得到如下

$$\beta_{5 \times 1} = [\beta_1^2, \beta_1\beta_2, \beta_2^2, \beta_1\beta_3, \beta_2\beta_3]^T$$

当然对应的 $\mathbf{L}_{6 \times 10}$ 矩阵中每行也取第1,2,3,4,5对应元素得到 $\mathbf{L}_{6 \times 5}$ ，而 $\rho_{6 \times 1}$ 不变。这样我们只要用SVD求解规模更小的矩阵就行了。

$$\mathbf{L}_{6 \times 5} \cdot \beta_{5 \times 1} = \rho_{6 \times 1}$$

最后得到

$$\beta_1 = \sqrt{\beta_1^2}, \quad \beta_2 = \sqrt{\beta_2^2}, \quad \beta_3 = \frac{\beta_1\beta_3}{\beta_1}$$

N=2的情况

此时待求量为： β_1, β_2 。我们取 $\beta_{10 \times 1}$ 中的第1,2,3个元素，总共3个得到如下

$$\beta_{5 \times 1} = [\beta_1^2, \beta_1\beta_2, \beta_2^2]^T$$

当然对应的 $\mathbf{L}_{6 \times 10}$ 矩阵中每行也取第1,2,3对应元素得到 $\mathbf{L}_{6 \times 3}$ ，而 $\rho_{6 \times 1}$ 不变。这样我们只要用SVD求解规模更小的矩阵就行了。

$$\mathbf{L}_{6 \times 3} \cdot \beta_{3 \times 1} = \rho_{6 \times 1}$$

最后得到

$$\beta_1 = \sqrt{\beta_1^2}, \quad \beta_2 = \sqrt{\beta_2^2}$$

高斯牛顿优化

我们的目标是优化两个坐标系下控制点间距的差，使得其误差最小，如下所示

$$f(\beta) = \sum_{(i,j) \text{ s.t. } i < j} \left(\| \mathbf{c}_i^c - \mathbf{c}_j^c \|^2 - \| \mathbf{c}_i^w - \mathbf{c}_j^w \|^2 \right)$$

因为我们前面已经计算了 $N = 4$ 的情况下 $\| c_i^c - c_j^c \|^2 = \| c_i^w - c_j^w \|^2$ 的表达式为：

$$\mathbf{L}_{6 \times 10} \cdot \beta_{10 \times 1} = \rho_{6 \times 1}$$

我们记待优化目标 β 为：

$$\begin{aligned} \beta_{10 \times 1} &= [\beta_1^2 \quad \beta_1\beta_2 \quad \beta_2^2 \quad \beta_1\beta_3 \quad \beta_2\beta_3 \quad \beta_3^2 \quad \beta_1\beta_4 \quad \beta_2\beta_4 \quad \beta_3\beta_4 \quad \beta_4^2]^T \\ &= [\beta_{11} \quad \beta_{12} \quad \beta_{22} \quad \beta_{13} \quad \beta_{23} \quad \beta_{33} \quad \beta_{14} \quad \beta_{24} \quad \beta_{34} \quad \beta_{44}]^T \end{aligned}$$

所以上面的误差函数可以写为：

*相机坐标系下 由从解得 4个特征向量算得
的控制点间距*

$$f(\beta) = \mathbf{L}\beta - \rho$$

世界坐标系下 已知的控制点距离

上式两边对 β 求偏导，由于 ρ 和 β 无关，所以一阶雅克比矩阵：

$$\begin{aligned} \mathbf{J} &= \frac{\partial f(\beta)}{\beta} = \left[\frac{\partial f(\beta)}{\partial \beta_1} \quad \frac{\partial f(\beta)}{\partial \beta_2} \quad \frac{\partial f(\beta)}{\partial \beta_3} \quad \frac{\partial f(\beta)}{\partial \beta_4} \right] \\ &= \left[\frac{\partial (\mathbf{L}\beta)}{\partial \beta_1} \quad \frac{\partial (\mathbf{L}\beta)}{\partial \beta_2} \quad \frac{\partial (\mathbf{L}\beta)}{\partial \beta_3} \quad \frac{\partial (\mathbf{L}\beta)}{\partial \beta_4} \right] \end{aligned}$$

丁应该是 6×4 的矩阵

前面我们已经知道 \mathbf{L} 的维度是 6×10 ， β 的维度是 10×1 ，我们以 \mathbf{L} 第一行 \mathbf{L}^1 为例来推导

$$\begin{aligned} \mathbf{L}^1 \beta &= [L_1^1 \quad L_2^1 \quad L_3^1 \quad L_4^1 \quad L_5^1 \quad L_6^1 \quad L_7^1 \quad L_8^1 \quad L_9^1 \quad L_{10}^1] \begin{bmatrix} \beta_{11} \\ \beta_{12} \\ \beta_{22} \\ \beta_{13} \\ \beta_{23} \\ \beta_{33} \\ \beta_{14} \\ \beta_{24} \\ \beta_{34} \\ \beta_{44} \end{bmatrix} \\ &= L_1^1 \beta_{11} + L_2^1 \beta_{12} + L_3^1 \beta_{22} + L_4^1 \beta_{13} + L_5^1 \beta_{23} + L_6^1 \beta_{33} + L_7^1 \beta_{14} + L_8^1 \beta_{24} + L_9^1 \beta_{34} + L_{10}^1 \beta_{44} \end{aligned}$$

分别求偏导后得到

$$\begin{aligned} \frac{\partial (\mathbf{L}^1 \beta)}{\partial \beta_1} &= 2L_1^1 \beta_1 + L_2^1 \beta_2 + L_4^1 \beta_3 + L_7^1 \beta_4 \\ \frac{\partial (\mathbf{L}^1 \beta)}{\partial \beta_2} &= L_2^1 \beta_1 + 2L_3^1 \beta_2 + L_5^1 \beta_3 + L_8^1 \beta_4 \\ \frac{\partial (\mathbf{L}^1 \beta)}{\partial \beta_3} &= L_4^1 \beta_1 + L_5^1 \beta_2 + 2L_6^1 \beta_3 + L_9^1 \beta_4 \\ \frac{\partial (\mathbf{L}^1 \beta)}{\partial \beta_4} &= L_7^1 \beta_1 + L_8^1 \beta_2 + L_9^1 \beta_3 + 2L_{10}^1 \beta_4 \end{aligned}$$

高斯牛顿法的增量方程：

$$\begin{aligned}\mathbf{H}\Delta\mathbf{x} &= \mathbf{g} \\ \mathbf{J}^T\mathbf{J}\Delta\mathbf{x} &= -\mathbf{J}^T f(x) \\ \mathbf{J}\Delta\mathbf{x} &= -f(x)\end{aligned}$$

对应非齐次项 $-f(\beta) = \rho - \mathbf{L}\beta$

ICP 求解位姿 十四 讲有

1. 记3D点在世界坐标系下的坐标及对应相机坐标系下的坐标分别是 $\mathbf{p}_i^w, \mathbf{p}_i^c, i = 1, \dots, n$
2. 首先分别计算它们的质心:

$$\begin{aligned}\mathbf{p}_0^w &= \frac{1}{n} \sum_{i=1}^n \mathbf{p}_i^w \\ \mathbf{p}_0^c &= \frac{1}{n} \sum_{i=1}^n \mathbf{p}_i^c\end{aligned}$$

3. 计算 $\{\mathbf{p}_i^w\}_{i=1,\dots,n}$ 去质心 \mathbf{p}_0^w 后的矩阵 A

$$A = \begin{bmatrix} \mathbf{p}_1^{w^T} - \mathbf{p}_0^{w^T} \\ \dots \\ \mathbf{p}_n^{w^T} - \mathbf{p}_0^{w^T} \end{bmatrix}$$

4. 计算 $\{\mathbf{p}_i^c\}_{i=1,\dots,n}$ 去质心 \mathbf{p}_0^c 后的矩阵 B :

$$\begin{aligned}\mathbf{p}_0^c &= \frac{1}{n} \sum_{i=1}^n \mathbf{p}_i^c \\ B &= \begin{bmatrix} \mathbf{p}_1^{c^T} - \mathbf{p}_0^{c^T} \\ \dots \\ \mathbf{p}_n^{c^T} - \mathbf{p}_0^{c^T} \end{bmatrix}\end{aligned}$$

5. 得到矩阵 H :

$$H = B^T A$$

6. 计算 H 的SVD分解:

$$H = U\Sigma V^T$$

7. 计算位姿中的旋转 R

$$R = UV^T$$

8. 计算位姿中的平移 t :

$$\mathbf{t} = \mathbf{p}_0^c - R\mathbf{p}_0^w$$

总结

1. 根据计算出来的 β, \mathbf{v} 得到相机坐标系下的4个控制点坐标 $\{\mathbf{c}_j = (x_j^c, y_j^c, z_j^c)\}, j = 1, \dots, 4$

$$\mathbf{x} = \sum_{i=1}^N \beta_i \mathbf{v}_i$$
2. 根据相机坐标系下控制点坐标 \mathbf{c}_j 和控制点系数 α_{ij} (通过世界坐标系下3D点计算得到), 得到相机坐标系下3D点坐标 \mathbf{p}_i^c

$$\mathbf{p}_i^c = \sum_{j=1}^4 \alpha_{ij} \mathbf{c}_j^c$$

3. 现在已经有3D点在世界坐标系下的坐标 \mathbf{p}_i^w 以及对应相机坐标系下的坐标 \mathbf{p}_i^c , 用 ICP 求解 R, t 即可。

参考

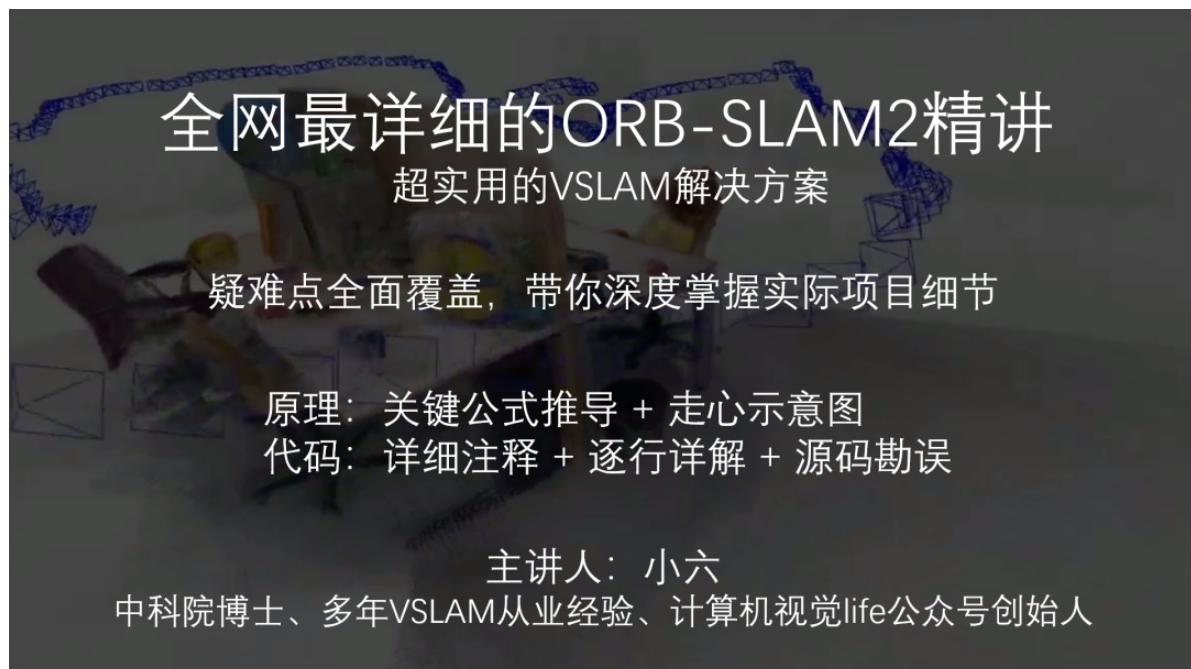
Lepetit V , Fua M N . EPnP: An Accurate O(n) Solution to the PnP Problem[J]. International Journal of Computer Vision, 2009.

小葡萄 <https://zhuanlan.zhihu.com/p/59070440>

Jessie <https://blog.csdn.net/jessecw79/article/details/82945918>

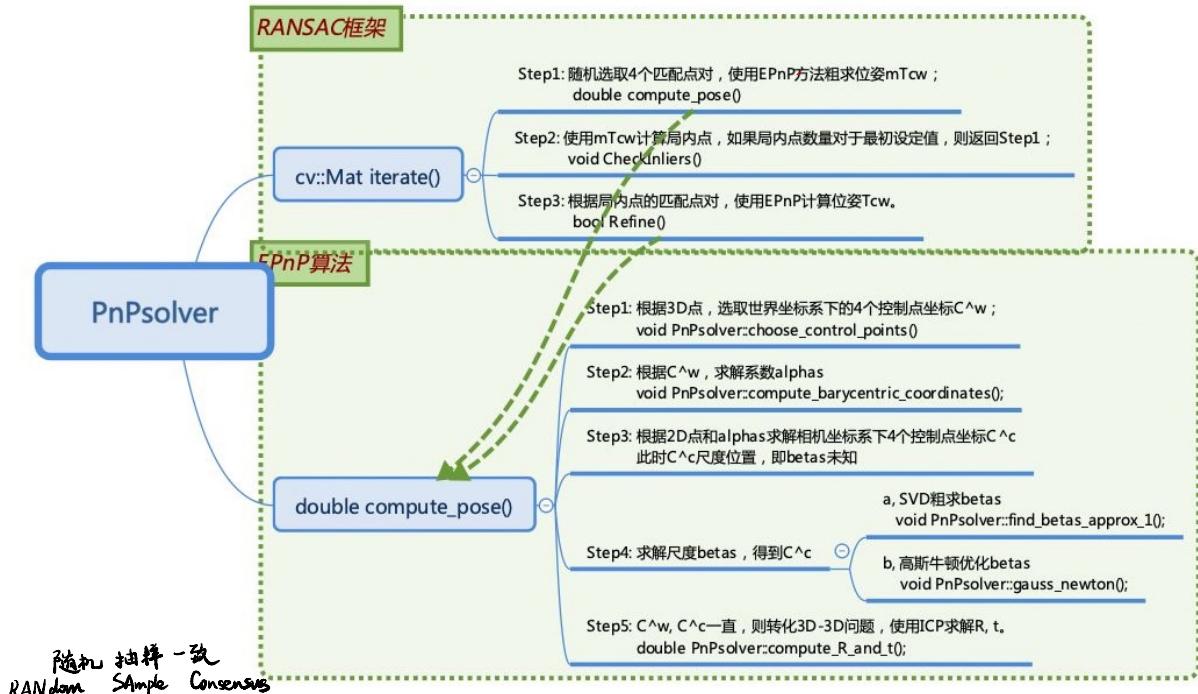
代码: <https://github.com/cvlab-epfl/EPnP>

[《全网最详细的ORB-SLAM2精讲：原理推导+逐行代码分析》](#) (点击可跳转课程详情)



长按或扫描二维码查看课程介绍和购买方式：





RANSAC

- ① 有一个模型适用于假设的局内点
- ② 用该模型测试其它数据，若其他数据也适用于该模型，则认为它也是局内点
- ③ 若有足够的局内点适用于该模型，那么估计的模型就合理
- ④ 用所有假设的局内点去重新估计模型 → 为了速度可以不重新估计，但容易受噪声影响
- ⑤ 用局内点和模型的错误率来评估模型

参数定义：

k	迭代次数
P	选取的点均为局内点的概率
w	一次从数据集中选取一个局内点的概率

$$w = \frac{\text{局内点的数目}}{\text{总数}}$$

估计模型需要选定 n 个点，那么 n 个点都是内点的概率是 w^n ， n 个点至少有一个不是内点的概率是 $1 - w^n$
 k 次迭代中，永远不会出现 n 个选择的点全不是内点的概率是 $(1 - w^n)^k$

$$1 - P = (1 - w^n)^k$$

$$\begin{aligned} k &= \log \frac{1 - P}{1 - w^n} \\ \xrightarrow{\text{换底公式}} k &= \frac{\log(1 - P)}{\log(1 - w^n)} \end{aligned}$$

参考：

<https://zhuanlan.zhihu.com/p/149017276>