前置知识

传感器测量中的概率分布

假设我们用一个激光测距仪测量某一物体的距离，测量了100次，会发现每次的测量值都不一样，统计一下分布频率的话，会发现其服从**正态分布**（Normal distribution，也叫高斯分布）。正态分布由均值 μ 和方差 σ^2 两个参数决定（如图1所示），其中方差 σ^2 是传感器本身的属性，由传感器厂家给出或人工标定。如果测量次数足够多的话，可以认为正态分布的均值 μ 就是距离的真值。反过来，如果我们只有一次测量，则可以认为真值服从**<以测量值为均值，方差仍然是 σ^2 的正态分布>**。

/** 【一个思考题：如何从贝叶斯学派观点出发，推导只有一次观测时真值的概率分布？欢迎讨论】 */

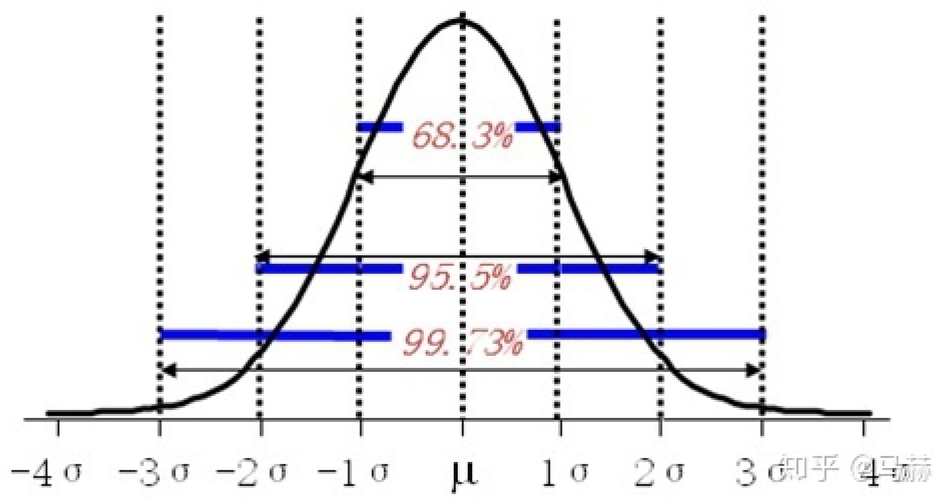

图1. 正态分布

一元变量正态分布为 $p(x) \sim N(\mu, \sigma^2)$ ， x 和 μ 都是标量， σ^2 (小sigma^2)是方差。

多元变量的正态分布 $p(x) \sim N(\mu, \Sigma)$ ， x 和 μ 为向量， Σ (大sigma)为协方差矩阵。

概率栅格地图

概率栅格地图是一种非常有用的地图形式，必修基础知识请看**这篇**。请记住其中一点，概率栅格地图通常使用对数形式的概率值。

那我们如何基于一帧激光scan来构造概率栅格地图呢？首先按照一定的分辨率把世界栅格化，然后每个激光点都会落在某个栅格中（称为击中）。由上一部分可知，**被击中的栅格的占据概率是概率分布曲线中均值 μ 对应的概率**；而对于附近没有被击中的栅格，将按照与最近的被击中栅格的距离在正态分布曲线中做概率值查询，查询出的那个概率就是这个栅格的占据概率。我们按照**这篇**中的方法对这个概率

做对数处理，就得到了算法中真正使用的栅格概率。对所有的栅格都按这种方式处理，我们就得到了这个scan的概率栅格地图。

> 这种处理类似概率机器人学中的「似然域模型」，但似然域模型考虑了更多的因素[2]。

前置知识介绍完毕。

一、Scan-Matching问题数学建模

开始进入正题～

我们要解决的是一个怎样的问题呢？

假设有一个机器人小萝卜，小萝卜上水平安装着一个二维激光雷达。当小萝卜在环境中运动的时候，会对周边产生一个观测，观测由一个个激光点组成，每个激光点即是一个平面坐标。

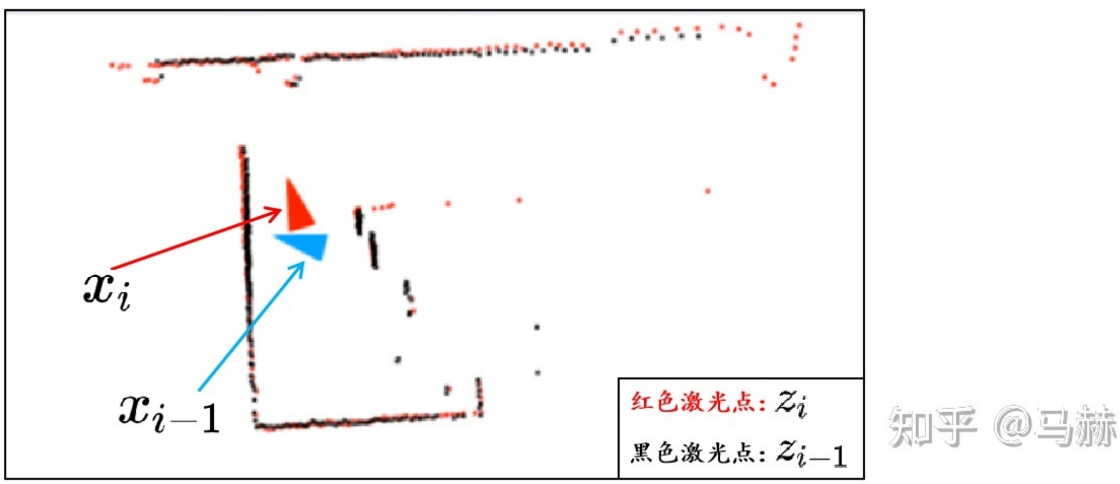

图2. 小萝卜的运动

假设在当前位置 x_i 上，小萝卜对环境产生的观测是 z_i ；在上一个相邻位置 x_{i-1} 上，小萝卜对环境的观测是 z_{i-1} 。从 x_{i-1} 到 x_i ，小萝卜自身也能感知自己移动了多少（尽管这个感知不准确），一般通过轮式里程计或者imu实现，我们记这个对自身移动的感知为 u 。**我们的问题是：已知 x_{i-1} 和 z_{i-1} ，也测量到了 u 和 z_i ，如何尽可能准确的求出 x_i ？**

这是一个最大后验概率估计问题，用数学语言来讲，**我们要使概率 $p(x_i|x_{i-1},u,z_{i-1},z_i)$ 最大化。**这个概率不是一个概率值，而是一个概率分布（更严谨一些，是关于<三元变量 x_i >的联合概率分布），当概率取得最大值时对应的那个 x_i ，即是我们对小萝卜当前位置的最优估计。

> 符号说明：以上描述中， x_{i-1} 和 x_i 是三维向量，包含一个平面坐标和一个方位角，因此准确的说， x_{i-1} 和 x_i 都是位姿，后文中将统一使用‘位姿’代替‘位置’； u 描述的是相邻位姿的变

> 化，是同样的三维向量； z_{i-1} 和 z_i 作为激光雷达观测，是一定数量的二维坐标的集合。另外， u 和 z_i 作为传感器测量值都是有噪声的。

在论文中，作者用符号 m 取代了 z_{i-1} 。 m 表示已知地图，相比于 z_{i-1} ， m 更具一般性，可以代表相邻的前一帧观测[3]（= z_{i-1} ），也可以代表之间若干观测的总和，两者各有优缺点[4]。这种替换对数学处理无影响；在代码实现中，无论 m 是取 z_{i-1} 还是取 $z_{i-1}+z_{i-2}+...$ ，都是以栅格地图的形式呈现的，因此也无影响。**下文中，将一律使用** m 。多说一句，如果 m 取 z_{i-1} ,则是scan-to-scan匹配问题；如果取 $z_{i-1}+z_{i-2}+...$ ，则是scan-to-map匹配问题。

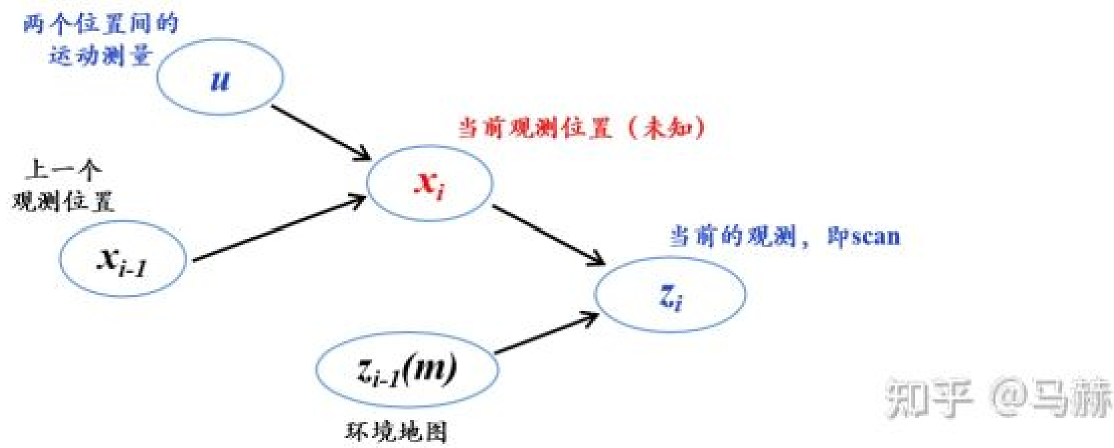

图3. 各量之间的约束关系

以上的「最大后验概率估计问题」是可以进一步拆解的。如图3所示， x_i 服从两个约束：① $x_{i-1}+u\approx x_i$ ，② $z_i|x_i\sim m$ 。第①个约束很好理解， x_i 应该位于上一个位置与运动测量之和 $x_{i-1}+u$ 的附近；第②个约束可以理解为，观测 z_i 是发生在观测位置 x_i 上的，因此 x_i 应当使得 z_i 服从已知地图。基于这两个约束，根据贝叶斯法则并移除不相关条件，我们有：

$p(x_i|x_{i-1},u,m,z_i)\propto p(z_i|x_i,m)p(x_i|x_{i-1},u)$ --------- ***公式(1)***

这样，原问题就转化为**【求解** $p(z_i|x_i,m)p(x_i|x_{i-1},u)$ **整体关于** x_i **的概率分布】**，使得 $p(z_i|x_i,m)p(x_i|x_{i-1},u)$ 取得全局最大值的那个 x_i ，即是我们对小萝卜当前位置的最优估计。

其中， $p(x_i|x_{i-1},u)$ 是我们熟悉的运动模型， $p(z_i|x_i,m)$ 是观测模型。前者是一个常规的多元变量高斯分布问题，容易求解；**在一些情况下，比如没有运动观测时，我们甚至可以略去运动模型**，仅考虑观测模型即可；在另一些情况下，比如imu或轮式里程计精度不高时，我们可以调低运动模型的权重，降低其对整体概率分布的影响。但后者的计算是一个难题，因为它容易陷入局部最优解。这篇**论文的核心贡献即在于提出了一种高效计算观测模型的方法**，既能获得鲁棒的最优解，又能对解的不确定性进行量化评价，方法如下。

我们假设 z_i 中的各个激光点（以 $z_i^{(k)}$ 表示）位置的概率分布是彼此独立的，则：

$$p(z_i|x_i,m)=\prod_k p(z_i^{(k)}|x_i,m)$$ ------ ***公式(2)***

对上述公式两侧同样按这篇中的方式取对数，让乘法变加法，我们有：

$$p_{log}(z_i|x_i,m)=\sum_k p_{log}(z_i^{(k)}|x_i,m)$$ ------ ***公式(3)***

现在，***公式(3)***就是我们的观测模型。那么如何来计算***公式(3)***？这里就要提到我们上文介绍过的概率栅格地图了。

重点：m，就是按照上一帧观测 z_{i-1}（或若干帧）建立出来的概率栅格地图！

> 这种根据前一帧观测建立概率栅格地图的过程，在论文中叫做“Lookup-Table Rasterization”，我把它翻译为“建立栅格化的概率查询表”。

$\sum_k p_{log}(z_i^{(k)}|x_i,m)$ 如何计算？按照位姿 x_i 把当前观测 z_i 投影到 m 中，把所有被击中的栅格的概率值相加，就是！在栅格地图的基础上，***公式(3)***的求解就是这么简单。

> 所得的 $p_{log}(z_i|x_i,m)$ 即是位姿 x_i 的得分，代表着在这个位姿下，当前观测 z_i 与已知环境 m 相一致的程度（i.e., 相关性）。得分越高，表明这个位姿越靠谱[5]。

现在，我们已经知道了观测模型的求解方法，但不要忘了，我们最终想要的是对小萝卜当前位姿 x_i 的尽可能准确的估计。通常来说我们可以获得一个 x_i 的粗略预估，这个预估可能来自于IMU或轮式里程计甚至是一个定死的值，因此是不准确的。那么如何得到尽可能准确的 x_i？

论文提出：可以**在预估位姿的附近建立 x_i 的搜索空间**，如果搜索空间足够大的话，我们就认为 x_i 的真值就一定存在于这个空间中。暴力遍历搜索空间中的所有位姿，并选出得分最高者，作为对 x_i 的最优估计。

二、多分辨率概率查询表加速搜索

暴力方法确实能够得到搜索空间范围内的全局最优解，但效率太低了。作者提出通过建立多分辨率的概率查询表来加速这个搜索过程。

> 「概率查询表」就是「概率栅格地图」，这里只是换个说法而已，以与论文保持一致。

回顾一下，构建 m 时，我们是按照某个分辨率把世界栅格化的，现在我们把这个分辨率调低（比如一倍），然后按照同样的方式构建一个新的概率栅格地图 m'，如图4所示。

可以看到， m' （红色）中的1个栅格覆盖了 m （灰色）中的4个栅格， 我们**把4个灰色栅格概率值的最大值赋给对应的红色栅格！**——这一点非常重要。

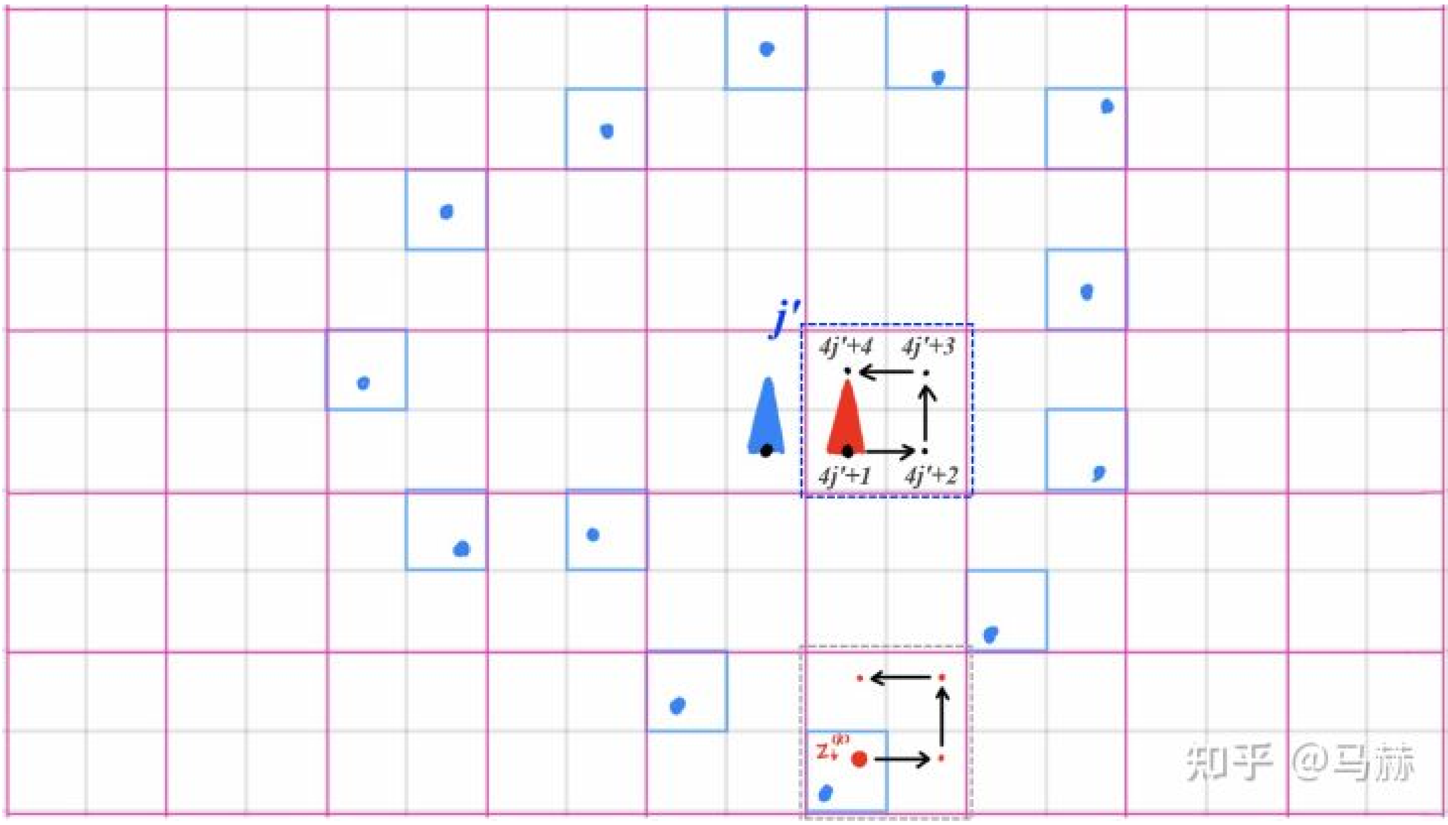

图4. 高/低分辨率查询表（灰色/红色）；图中蓝色激光点代表上一帧观测z_{i-1}，蓝色三角形代表上一个观测位置x_{i-1}。

暴力搜索时，我们需要遍历三个变量：角度，横向平移，纵向平移。现在我们把角度放在最外层，内层就只有两个平移，**这两个平移正是我们加速的对象**。

我们记 x_i 在 m 中的搜索空间为 W ，在 m' 中同等物理尺度的搜索空间为 W' 。 W 与 W' 的物理空间大小是一样的，但元素的数量却不一样！举个栗子，我们设定搜索空间的物理大小为预估位姿附近〔旋转 $\pm 10^{\circ}$ 、横向 ± 1 米、纵向 ± 1 米〕的空间，这里**只考虑平移**。假定 m 的分辨率是10cm，则 W 中元素的个数是 100 个； m' 的分辨率是20cm，则 W' 中元素的个数是 25 个。

那如何实现加速呢？我们只需对比一下，两个空间中相对应位姿的得分，就能明白。

记 W 中的第 j 个位姿为 $x_i^{(j)}$ ，其得分记作 $score_{high}(j)$ ；

记 W' 中的第 j' 个位姿为 $x_i'^{(j')}$ ，其得分记作 $score_{low}(j')$ ；

如图4蓝色虚线区域所示， W' 中的第 j' 个位姿（位姿，即投影到的栅格），对应 W 中的第 $j = 4j' + 1$ ~ $j = 4j' + 4$ 个位姿，我们不难推断出：

$score_{high}(j) \le score_{low}(j')$, $in\ which:\ 4j'+1 \le j \le 4j'+4$ ------ ***公式(4)***

也即：

$$\max_{4j'+1\le j\le 4j'+4} score_{high}(j) \le score_{low}(j')$$ ------ ***公式(5)***

为什么***公式(5)***所表示的得分关系能够成立？因为在构建低分辨率栅格时就已经决定了——“把4个灰色栅格概率值的最大值赋给对应的红色栅格！”。

> 试想，一个位姿的得分是所有单个激光点击中栅格的对数概率值之和，我们取某单个激光点 $z_i^{(k)}$ 为例（图4中灰色虚线框），当遍历 W 空间中的位姿 $x_i^{(4j'+1)}$ 到 $x_i^{(4j'+4)}$ 时，点 $z_i^{(k)}$ 也会投影到不同的高分辨率栅格（灰色），但无论投影到哪个栅格，对应的得分必然 $\le$ 红色栅格的得分（即点 $z_i^{(k)}$ 投影到对应低分辨率栅格的得分）。其它激光点的情形也都一样。

因此我们得到「**结论：低分辨率栅格的得分是对应所有高分辨率栅格得分的上界！**」

利用上述结论，我们加速搜索的做法是：

1.我们先遍历所有的低分辨率栅格，找出得分最高的那个；
2.遍历得分最高的低分辨率栅格对应的高分辨率栅格，找出得分最高的那个，记作 H_{best}；
3.遍历剩余的低分辨率栅格：{
    如果第 j' 个低分辨率栅格的得分 $L_{j'} < H_{best}$，什么也不做，继续遍历下一个；
    否则，如果 $L_{j'} > H_{best}$，则：{
        遍历该低分辨率栅格对应的所有高分辨率栅格：{
            如果某个高分辨率栅格的得分 $L_{4\times j'+\#} > H_{best}$，则 $H_{best} \leftarrow L_{4\times j'+\#}$；
            否则，什么也不做，继续遍历下一个；
        }
    }
}
（以上，H 代表*high-resolution*，L 代表*low-resolution*）

图5. 多分辨率查询表加速算法

加速的思想很简单：低分辨率栅格的得分是对应所有高分辨率栅格得分的上界，如果某个低分辨率栅格的得分低于 H_{best}，则其对应的所有高分辨率栅格的得分必然也不会高于 H_{best}，我们索性不再考虑这部分高分辨率栅格，这个过程，也叫做「**剪枝**」！我们正是通过剪掉不必考虑的分支，来实现加速的。

事实上，由于我们选择了“得分最高的低分辨率栅格里得分最高的高分辨率栅格”来作为初始 H_{best}，因此很多高分辨率栅格都会被直接“剪枝”。在论文中，高分辨率设定为3cm，低分辨率设定为30cm，也即一个低分辨率栅格可以代表100个高分辨率栅格！剪枝加速的效果更明显！

> 1：10是很常用的高低分辨率比例。

这种通过构造低分辨率查询表来建立「上界」，通过上界对比来剪枝加速的方法，我们称作**「分支定界」算法**。推荐一篇很好的文章：分支定界法及优化求解器by留德华叫兽。

CSM帧匹配算法「分支定界」加速的完整示意图如下所示~

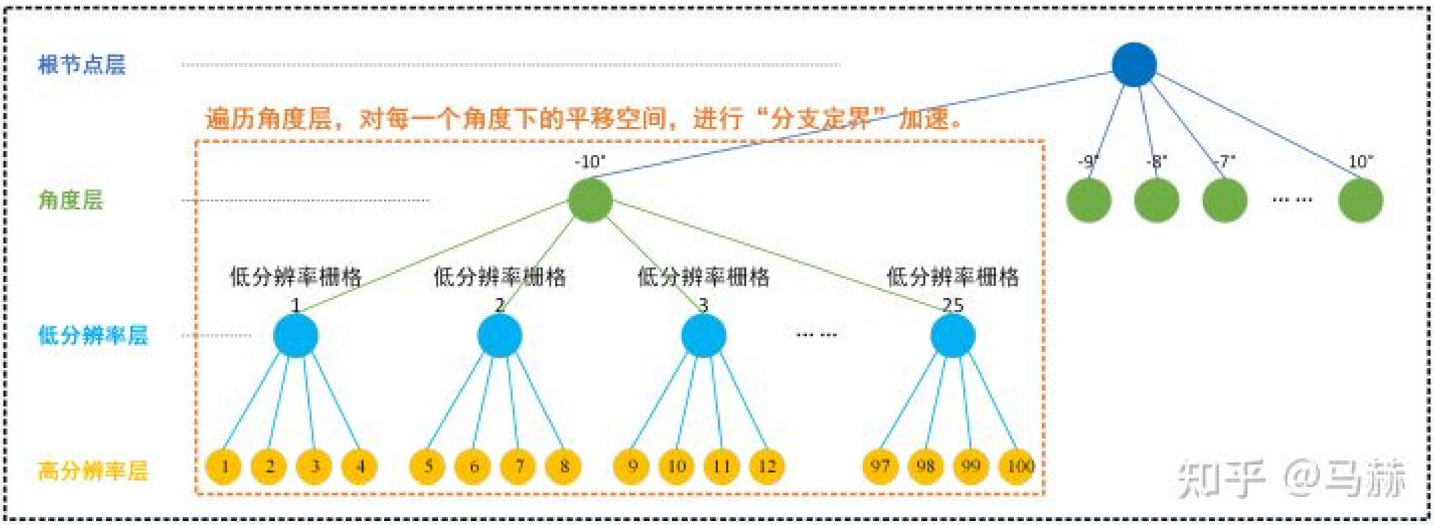

图6. 「分支定界」策略加速搜索

最外层遍历每一个角度，对各个角度下的平移空间进行「分支定界」加速搜索，获得最高得分；再对比各个角度的最高得分，获得整个搜索空间内的全局最高得分，该得分对应的那个位姿就是我们对 x_i 的最准确估计（i.e., 全局最优解）。

三、回顾问题

现在，我们回顾一下最开始的问题，以及我们做了什么：

我们最终的目的是尽可能准确的估算小萝卜的当前位姿 x_i ~
我们的方法是估算出观测模型关于 x_i 的概率分布[6]，观测模型取得最大值时对应的位置即是对 x_i 的最准确估计 ~
我们用概率栅格地图投影的方式，把观测模型的求解离散化：计算搜索空间 W 内所有离散备选位姿的得分，得分最高的那个即是对 x_i 的最准确估计 ~
为了提升搜索效率，我们应用了多分辨率查询表加速的策略（分支定界）~

以上这些，就是「**CSM帧匹配算法**」的本质。

细心的读者可以发现，我们并没有真的计算出观测模型的概率分布，体现在两点：①我们只计算了搜索空间内的离散点的得分，空间外的区域我们并没有计算；②即使在搜索空间内，我们也并没有计算所有的点，我们通过分支定界的方式跳过了很多备选位姿。

现在，如果我们想**量化的评价最优解的不确定性**，应该怎么做？答案是还是要计算出真正的观测模型概率分布才可以~

四、不确定性评价：协方差矩阵

不确定性与协方差

> 协方差矩阵能够表达一个多元变量的不确定性。

我们已经得到了小萝卜当前位置的最优估计，但这个估计结果可靠吗？有多可靠？如何量化表达其不确定性？

答案很简单，那就是计算其高斯分布的方差，方差可以反映不确定性的大小。想象一个一元变量的高斯分布，**方差越小，意味着概率分布曲线越陡峭，概率分布越集中，不确定性越小**；方差越大，则曲线越平缓和均匀，概率分布越分散，不确定性越大。如下图所示~

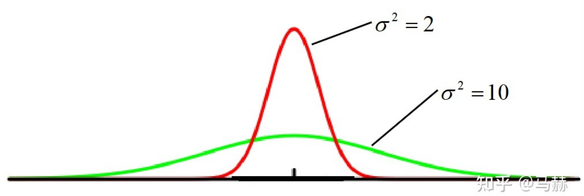

图7. 方差决定概率分布的集中程度，反映不确定性大小

对于一个多元变量来说，协方差矩阵就是它的“方差”——协方差矩阵的对角线正是由各个“元”的方差组成。协方差矩阵更直观的意义参见这篇：如何直观地理解「协方差矩阵」？。

计算协方差矩阵

先回顾一下计算协方差矩阵的数学公式~

> 对于多元变量 x ，记其均值为 $u=E(x)$ ；
> 则协方差矩阵为：
>
> $$\begin{aligned}\sum\nolimits_x &= E[(x-u)(x-u)^T] \\ &= E(xx^T-ux^T-xu^T+uu^T) \\ &= E(xx^T)-E(ux^T)-E(xu^T)+uu^T \\ &= E(xx^T)-uu^T-uu^T+uu^T \\ &= E(xx^T)-uu^T\end{aligned}$$

> 该式记作 ------ ***公式(6)***
>
> 式中，符号 $\sum_x$ 表示变量 x 的协方差矩阵。

前面说到，我们实际上跳过了搜索空间内很多离散点的计算，现在我们把所有离散点的得分都计算出来。然后，把paper中计算协方差矩阵的公式贴出来，如下：

Once the value of the cost function has been evaluated over a range of values of x_i, a multivariate Gaussian distribution can be fit to the data. Let $x_i^{(j)}$ be the j^{th} evaluation of x_i:

是离散的，所以用求和

$$K = \sum_j x_i^{(j)} x_i^{(j)T} p(x_i^{(j)} | x_{i-1}, u, m, z)$$

$$u = \sum_j x_i^{(j)} p(x_i^{(j)} | x_{i-1}, u, m, z)$$

$$s = \sum_j p(x_i^{(j)} | x_{i-1}, u, m, z)$$

协方差矩阵

$$\Sigma_{x_i} = \frac{1}{s}K - \frac{1}{s^2}uu^T$$

图8. 协方差矩阵的计算

现在，我们来证明一下，paper中的公式和 ***公式(6)*** 其实是一样的~

观察一下paper中式子的形式，我们可以发现：

K 其实就是***公式(6)***中的 $E(xx^T)$ ：蓝色部分相当于“ xx^T ”，橙色部分是“ xx^T ”出现的“概率”，两者相乘，就是均值。
u 其实就是***公式(6)***中的 $u = E(x)$ ，这个没什么好解释的。
s 则是对所有的“概率值”求和。

图8中，最终的协方差表达式 $\sum_{x_i}$ 和 ***公式(6)*** 很像，除了要除以 s 和 s^2 。原因很简单，这是一种

「**归一化**」处理，一方面，搜索空间只是整个分布域的一部分，另一方面，橙色部分并不是真正的“概率”（而是得分），搜索空间中各个位姿的“概率”之和也不为1。归一化处理之后，各个位姿的“概率”会变成真正的概率，空间中所有备选位姿的概率值之和为1。

paper中的这些处理，实质上是根据搜索空间内的得分分布，拟合出一个真正的高斯概率分布，并计算出协方差矩阵，量化地评价概率分布的不确定性。

> 工业应用中，量化评价最优解的不确定性非常重要，给出协方差矩阵计算公式也是这篇paper的贡献之一。

【完结~~】