SLAM14讲—13建图

13.1概述

建图的意义：

- 定位
- 导航
  至少需要知道**地图哪些地方不可通过，而哪些地方是可以通过的**。这至少得是一种**稠密**的地图。
- 避障
- 重建
- 交互

稀疏地图只建模感兴趣的部分，而稠密地图则是建模**所有看到过**的部分。

13.2单目稠密重建

13.2.1立体视觉

稠密重建，需要知道每一个像素点（或大部分像素点）的距离，大致上的解决方案有：

1. 使用单目相机，利用移动相机之后进行三角化，测量像素的距离
2. 使用双目相机，利用左右目的视差计算像素的距离（多目原理相同）
3. 使用RGB-D相机直接获得像素距离
   在稠密深度图估计中，我们无法把每个像素都当作特征点，计算描述子。因此，稠密深度估计问题中，匹配是很重要的一环：如何确定第一张图的某像素，出现在其他图里的位置？这需要**极线搜索**和**快匹配技术**。然后可以利用三角测量确定某个像素的深度。但在这里我们需要很多次三角测量让深度估计收敛。我们希望深度估计随着测量的增加，从一个非常不确定的量，逐渐收敛到一个稳定值。即**深度滤波器技术**。

13.2.2极线搜索与块匹配

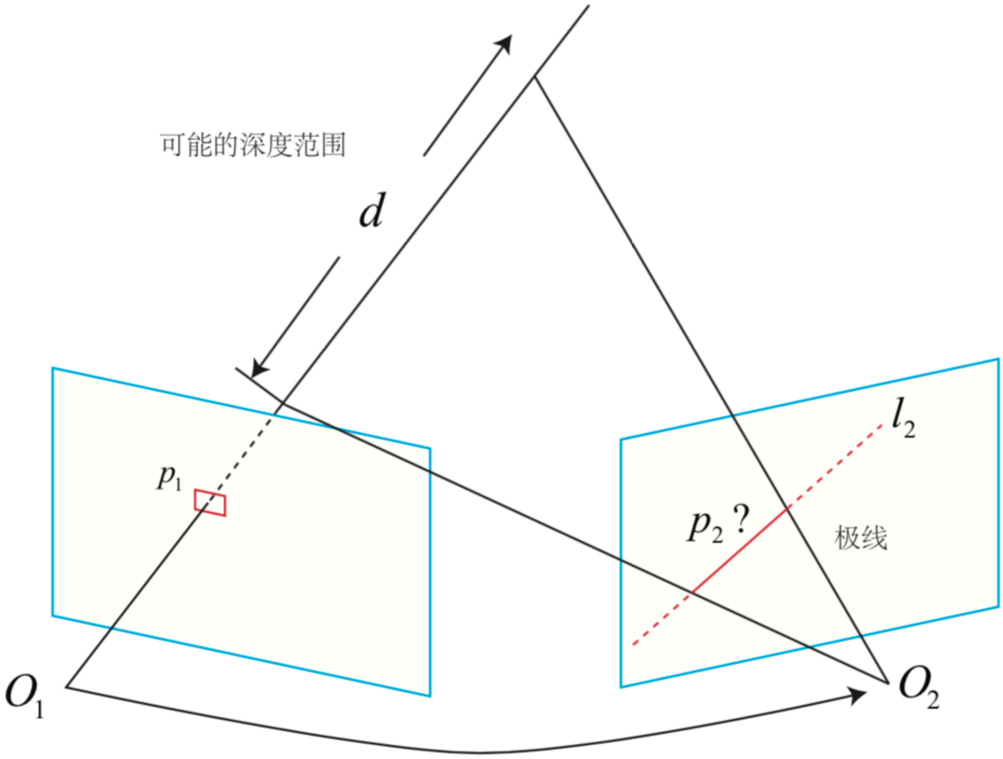

图 13-2　极线搜索的示意图。

左边的相机观测到了某个像素$\mathbf{p_1}$。由于这是一个单目相机，我们无法知道它的深度，所以假设这个深度可能在某个区域之间，即（d_{min},+∞）。故该像素对应的空间点就分布在某条线段上（本例中是射线），在另一个视角（右侧相机）看来，这条线条的投影形成图像平面上的一条线，即极线。

- 那如何确定极线上的哪个点是我们刚才看到的$\mathbf{p_1}$?
  单个像素的亮度没有区分性，但我们可以比较像素块。
  取$\mathbf{p_1}$周围的小块，并且在极线上也取了很多个小块。不妨把$\mathbf{p_1}$周围的小块记作**A**（大小为wxw），极线上的n个小块记作$\mathbf{B_i}$,i=1,2,...,n。
  计算小块与小块之间差异的方法有：
  1. SAD(Sum of Absolute Difference)。顾名思义，即取两个小块的差的绝对值之和：
  $$S(\boldsymbol{A}, \boldsymbol{B})_{SAD} = \sum_{i,j} |\boldsymbol{A}(i,j) - \boldsymbol{B}(i,j)|. \tag{13.1}$$
  2. SSD。SSD 并不是说大家喜欢的固态硬盘，而是 Sum of Squared Distance(SSD)（平方和）的意思：
  $$S(\boldsymbol{A}, \boldsymbol{B})_{SSD} = \sum_{i,j} (\boldsymbol{A}(i,j) - \boldsymbol{B}(i,j))^2. \tag{13.2}$$
  3. NCC(Normalized Cross Correlation)（归一化互相关）。这种方式比前两者要复杂一

些，它计算的是两个小块的相关性：

$$S(\boldsymbol{A},\boldsymbol{B})_{NCC}=\frac{\sum\limits_{i,j}\boldsymbol{A}(i,j)\boldsymbol{B}(i,j)}{\sqrt{\sum\limits_{i,j}\boldsymbol{A}(i,j)^2\sum\limits_{i,j}\boldsymbol{B}(i,j)^2}}. \tag{13.3}$$

请注意，由于这里用的是相关性，所以相关性接近 0 表示两个图像不相似，而接近 1 才表示相似。前面两种距离则是反过来的，接近 0 表示相似，而大的数值表示不相似。

- 那么对于不同图像进行极线搜索，我们估计的深度分布将发生怎样的变化?
  - 深度滤波器

13.2.3高斯分布的深度滤波器

$$P(d)=N(\mu,\sigma^2). \tag{13.4}$$

设某个像素点的深度d服从

每当新的数据到来，我们都会观测它的深度。假设这次观测是一个高斯分布

$$P(d_{obs})=N(\mu_{obs},\sigma_{obs}^2). \tag{13.5}$$

那么我们如何更新原先d的分布?

设融合后的d的分布为 $N(\mu_{fuse},\sigma_{fuse}^2)$，那么根据高斯分布的乘积，有：

$$\mu_{fuse}=\frac{\sigma_{obs}^2\mu+\sigma^2\mu_{obs}}{\sigma^2+\sigma_{obs}^2},\quad \sigma_{fuse}^2=\frac{\sigma^2\sigma_{obs}^2}{\sigma^2+\sigma_{obs}^2}. \tag{13.6}$$

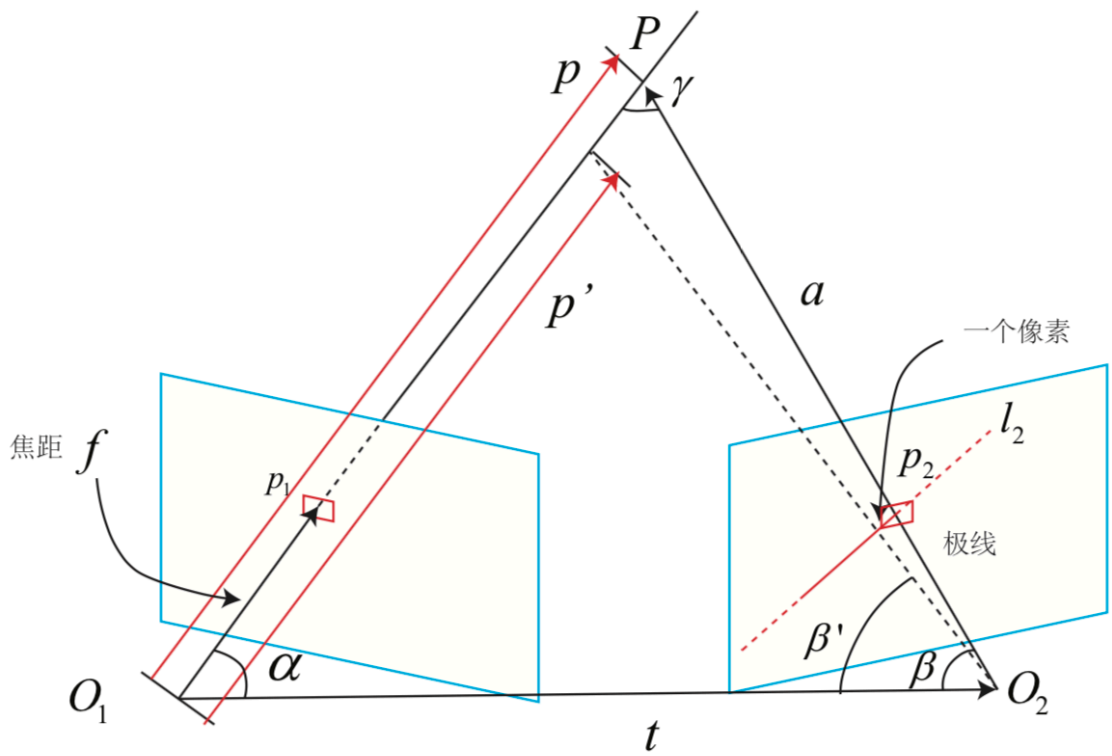

图 13-4　不确定性分析。

以图 13-4 为例。考虑某次极线搜索，我们找到了 $\boldsymbol{p}_1$ 对应的 $\boldsymbol{p}_2$ 点，从而观测到了 $\boldsymbol{p}_1$ 的深度值，认为 $\boldsymbol{p}_1$ 对应的三维点为 $\boldsymbol{P}$。从而，可记 $\boldsymbol{O}_1\boldsymbol{P}$ 为 $\boldsymbol{p}$，$\boldsymbol{O}_1\boldsymbol{O}_2$ 为相机的平移 $\boldsymbol{t}$，$\boldsymbol{O}_2\boldsymbol{P}$ 记为 $\boldsymbol{a}$。并且，把这个三角形的下面两个角记作 α, β。现在，考虑极线 l_2 上存在着一个像素大小的误差，使得 β 角变成了 β'，而 $\boldsymbol{p}$ 也变成了 $\boldsymbol{p}'$，并记上面那个角为 γ。我们要问的是，这一个像素的误差，会导致 $\boldsymbol{p}'$ 与 $\boldsymbol{p}$ 产生多大的差距呢？

根据几何关系计算可以得到：

$$\|\boldsymbol{p}'\| = \|\boldsymbol{t}\| \frac{\sin \beta'}{\sin \gamma}.$$

于是，我们可以认为极线搜索的块匹配仅有一个像素的误差，设：

$$\sigma_{obs} = \|\boldsymbol{p}\| - \|\boldsymbol{p}'\|. \qquad (13.11)$$

接下来进行深度数据融合。在实际工程中，当不确定性小于一定阈值后，就可以认为深度数据已经收敛了。
下面是估计稠密深度的一个完整的过程：

1. 假设所有像素的深度满足某个初始的高斯分布；
2. 当新数据产生时，通过极线搜索和块匹配确定投影点位置；
3. 根据几何关系计算三角化后的深度以及不确定性；
4. 将当前观测融合进上一次的估计中。若收敛则停止计算，否则返回 2。

13.3实践：单目稠密重建

13.4实验分析与讨论

13.5RGB-D稠密建图

octomap_mapping.cpp及pointcloud_mapping.cpp中第16行修改为如下代码

```
//原代码
vector<Eigen::Isometry3d> poses;
//修改代码
vector<Eigen::Isometry3d,Eigen::aligned_allocator<Eigen::Isometry3d>> poses;
```

注意一下data的存放位置，可能会影响读取

八叉树octovis的显示需要安装

```
//依赖项安装
sudo apt-get install doxygen
sudo apt-get install libqglviewer-dev-qt4
git clone https://github.com/OctoMap/octomap//然后各种编译安装
```

点云地图部分：

体素滤波器：PCL实现的VoxelGrid类通过输入的点云数据创建一个三维体素栅格（可把体素栅格想象为微小的空间三维立方体的集合），然后在每个体素（即，三维立方体）内，用体素中所有点的重心来近似显示体素中其他点，这样该体素就内所有点就用一个重心点最终表示，对于所有体素处理后得到过滤后的点云。这种方法比用体素中心来逼近的方法更慢，但它对于采样点对应曲面的表示更为准确。

```
pcl::VoxelGrid<pcl::PointXYZ> sor;
sor.setInputCloud(pointCloud_raw);
sor.setLeafSize(0.05f, 0.05f, 0.05f);//体素大小，5*5*5cm
sor.filter(*pointCloud_filter);
```