

# 第七节课习题

高翔

2019年11月

## 1 习题说明

- 第  $i$  节课习题所有材料打包在  $Li.zip$  中,  $\forall i = 1 \dots 8$ 。
- 习题分为若干种: **计算类**习题, 需要读者编程计算一个实际问题, 我们会附有参考答案以供自测。**操作类**习题, 会指导读者做一个具体的实验, 给出中间步骤截图或结果。**简述类**习题则提供阅读材料, 需要读者阅读材料后, 回答若干问题。
- 每个习题会有一定的分值。每次习题分值加和为 10 分。你需要获得 8 分以上才能得到“通过”的评价。带 \* 的习题为附加题, 会在总分之外再提供一定的分值, 所以总和可能超过 10 分。换句话说, 你也可以选择一道附加题, 跳过一道正常题。
- 每道习题的给分由助教评判, 简述类习题可能存在一定开放性, 所以评分也存在主观因素。
- 请利用深蓝学院系统提交习题。每次习题我们会记通过与否。提交形式为 word 或 pdf 格式报告, 如有编程习题请提交可编译的源码。
- 为方便读者, 我通常会准备一些阅读材料, 放在 books/或 papers/目录下。请读者按个人需求使用这些材料。它们多数是从网络下载的, 如果侵犯到你的权利, 请及时告诉我。
- 每个习题会标注大致用时, 但视同学个人水平可能会有出入。
- 习题的完成情况会影响你对本课程内容的掌握程度, 请认真、独立完成。**习题总得分较高的同学将获得推荐资格。**

## 2 Bundle Adjustment (5 分, 约 3 小时)

### 2.1 文献阅读 (2 分)

我们在第五讲中已经介绍了 Bundle Adjustment, 指明它可以用于解 PnP 问题。现在, 我们又在后端中说明了它可以用于解大规模的三维重构问题, 但在实时 SLAM 场合往往需要控制规模。事实上, Bundle Adjustment 的历史远比我们想象的要长。请阅读 Bill Triggs 的经典论文 Bundle Adjustment: A Modern Synthesis (见 paper/目录)<sup>1</sup>, 了解 BA 的发展历史, 然后回答下列问题:

1. 为何说 Bundle Adjustment is slow 是不对的?
2. BA 中有哪些需要注意参数化的地方? Pose 和 Point 各有哪些参数化方式? 有何优缺点。
3. \* 本文写于 2000 年, 但是文中提到的很多内容在后面十几年的研究中得到了印证。你能看到哪些方向在后续工作中有所体现? 请举例说明。

### 2.2 BAL-dataset (3 分)

BAL (Bundle Adjustment in large) 数据集 (<http://grail.cs.washington.edu/projects/bal/>) 是一个大型 BA 数据集, 它提供了相机与点初始值与观测, 你可以用它们进行 Bundle Adjustment。现在, 请你使用 g2o, 自己定义 Vertex 和 Edge (不要使用自带的顶点类型, 也不要像本书例程那边调用 Ceres 来求导), 书写 BAL 上的 BA 程序。你可以挑选其中一个数据, 运行你的 BA, 并给出优化后的点云图。

本题不提供代码框架, 请独立完成。提示:

1. 注意 BAL 的投影模型比教材中介绍的多了个负号;

---

<sup>1</sup> 本文比较长, 建议选读 1-5 节。

### 3 直接法的 Bundle Adjustment (5 分, 约 3 小时)

#### 3.1 数学模型

特征点法的 BA 以最小化重投影误差作为优化目标。相对的, 如果我们以最小化光度误差为目标, 就得到了直接法的 BA。之前我们在直接法 VO 中, 谈到了如何用直接法去估计相机位姿。但是直接法亦可用来处理整个 Bundle Adjustment。下面, 请你推导直接法 BA 的数学模型, 并完成它的 g2o 实现。注意本题使用的参数化形式与实际的直接法还有一点不同, 我们用  $x, y, z$  参数化每一个 3D 点, 而实际的直接法多采用逆深度参数化 [1]。

本题给定 7 张图片, 记为 0.png 至 6.png, 每张图片对应的相机位姿初始值为  $\mathbf{T}_i$ , 以  $\mathbf{T}_{cw}$  形式存储在 poses.txt 文件中, 其中每一行代表一个相机的位姿, 格式如之前作业那样:

$$\text{time}, t_x, t_y, t_z, q_x, q_y, q_z, q_w$$

平移在前, 旋转 (四元数形式) 在后。同时, 还存在一个 3D 点集  $P$ , 共  $N$  个点。其中每一个点的初始坐标记作  $\mathbf{p}_i = [x, y, z]^T_i$ 。每个点还有自己的固定灰度值, 我们用 16 个数来描述, 这 16 个数为该点周围  $4 \times 4$  的小块读数, 记作  $I(p)_i$ , 顺序见图 1。换句话说, 小块从  $u-2, v-2$  取到  $u+1, v+1$ , 先迭代  $v$ 。那么, 我们知道, 可以把每个点投影到每个图像中, 然后再看投影后点周围小块与原始的  $4 \times 4$  小块有多大差异。那么, 整体优化目标函数为:

$$\min \sum_{j=1}^7 \sum_{i=1}^N \sum_W \|I(\mathbf{p}_i) - I_j(\pi(\mathbf{K}\mathbf{T}_j\mathbf{p}_i))\|_2^2 \quad (1)$$

即最小化任意点在任意图像中投影与其本身颜色之差。其中  $\mathbf{K}$  为相机内参 (在程序内以全局变量形式给定),  $\pi$  为投影函数,  $W$  指代整个 patch。下面, 请回答:

1. 如何描述任意一点投影在任意一图像中形成的 error?
2. 每个 error 关联几个优化变量?
3. error 关于各变量的雅可比是什么?

#### 3.2 实现

下面, 请你根据上述说明, 使用 g2o 实现上述优化, 并用 pangolin 绘制优化结果。程序框架见 code/directBA.cpp 文件。实现过程中, 思考并回答以下问题:

1. 能否不要以  $[x, y, z]^T$  的形式参数化每个点?
2. 取  $4 \times 4$  的 patch 好吗? 取更大的 patch 好还是取小一点的 patch 好?
3. 从本题中, 你看到直接法与特征点法在 BA 阶段有何不同?
4. 由于图像的差异, 你可能需要鲁棒核函数, 例如 Huber。此时 Huber 的阈值如何选取?

提示:

1. 构建 Error 之前先要判断点是否在图像中, 去除一部分边界的点。
2. 优化之后, Pangolin 绘制的轨迹与地图如图 1 所示。
3. 你也可以不提供雅可比的计算过程, 让 g2o 自己计算一个数值雅可比。
4. 以上数据实际取自 DSO[1]。

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

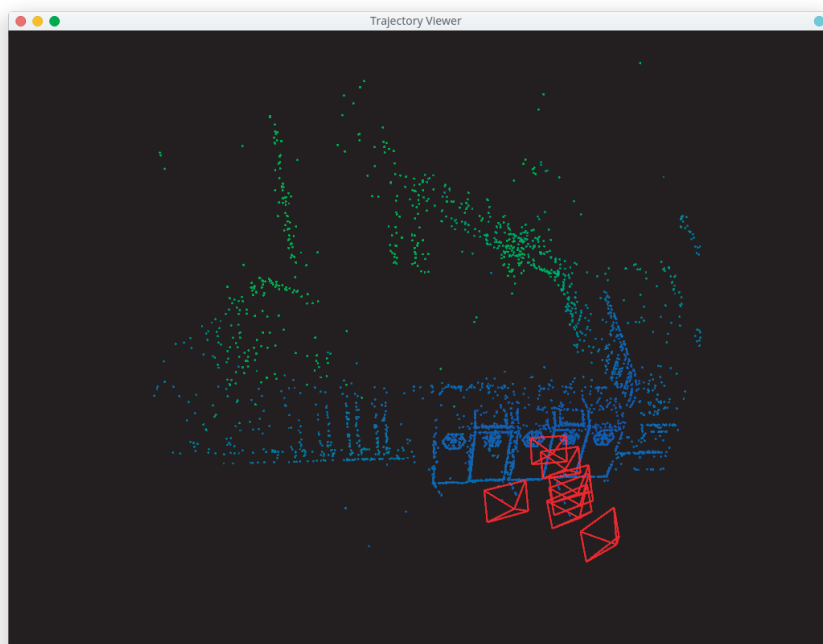
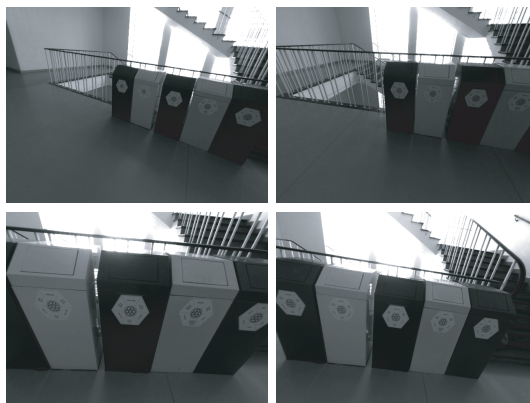


图 1: 直接法 BA 的图例。左上: 点颜色的定义顺序, 其中 11 号点是观测到的位置; 右上: 图片示例; 中间: 优化后的相机位置与点云。

## Bibliography

- [1] J. Engel, V. Koltun, and D. Cremers, “Direct sparse odometry,” *arXiv preprint arXiv:1607.02565*, 2016.