第七次作业

第二题：Bundle Adjustment

2.1 文献阅读

2.1.1 为何说Bundle Adjustment is slow 是不对的？

该说法忽略了H矩阵具有系数结构的特征，该特征可以使用加速技巧进行求解。

2.1.2 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。

1. 需要被参数化的地方包括

   相机位姿、相机内参、三维特征点P以及投影后的像素坐标

2. Pose 和 Point 各有那些参数化方式？有何缺点？

Pose

旋转	平移	缺点
欧拉角	三维位移向量	万向锁
四元数	三维位移向量	理解困难，不够直观
旋转矩阵	三维位移向量	占用了较大的内存

Point

三维坐标点（X,Y,Z）：简单直观，但无法描述无限远点

逆深度 ：能够建模无穷远点，在实际应用中，逆深度也具有更好的数值稳定性。

2.1.3 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。

- Intensity-based方法就是直接法的Bundle Adjustment;
- 文中提的Network Structure对应现在应用比较广泛的图优化方式;
- H的稀疏性可以实现BA实时，在07年的PTAM上实现。

2.2 BAL-dataset

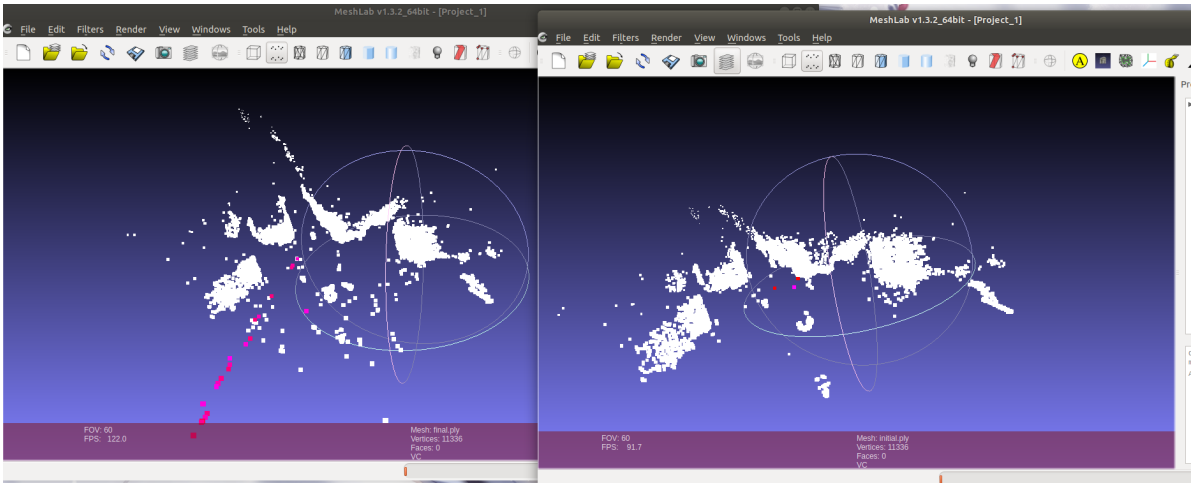
```
Vector2d project(const Vector3d &point) {
        //1.把世界坐标转换成像素坐标
        Vector3d pc = _estimate.rotation * point + _estimate.translation;
        //2.归一化坐标，BAL的投影模型比教材中介绍的多一个负号
        pc = -pc / pc[2];
        double r2 = pc.squaredNorm();
        //3.考虑归一化坐标的畸变模型
        double distortion = 1.0 + r2 * (_estimate.k1 + _estimate.k2 * r2);
        //4.根据内参模型计算像素坐标
        return Vector2d(_estimate.focal * distortion * pc[0],
                        _estimate.focal * distortion * pc[1]);
    }
```

```
//继承并重写BaseVertex类，并实现接口
class VertexPoint : public g2o::BaseVertex<3, Vector3d> {
public:
    EIGEN_MAKE_ALIGNED_OPERATOR_NEW;

    VertexPoint() {}

    virtual void setToOriginImpl() override {
        _estimate = Vector3d(0, 0, 0);
    }

    virtual void oplusImpl(const double *update) override {
        _estimate += Vector3d(update[0], update[1], update[2]);
    }

    virtual bool read(istream &in) {}

    virtual bool write(ostream &out) const {}
};
```

```
//边的定义
class EdgeProjection :
```

```
    public g2o::BaseBinaryEdge<2, Vector2d, VertexPoseAndIntrinsics, VertexPoint>
{
public:
    EIGEN_MAKE_ALIGNED_OPERATOR_NEW;

    virtual void computeError() override {
        auto v0 = (VertexPoseAndIntrinsics *) _vertices[0];
        auto v1 = (VertexPoint *) _vertices[1];
        auto proj = v0->project(v1->estimate());
        _error = proj - _measurement;
    }

    // use numeric derivatives
    virtual bool read(istream &in) {}

    virtual bool write(ostream &out) const {}

};
```

结果对比图



第三题： 直接法的 Bundle Adjustment

3.1 数学模型

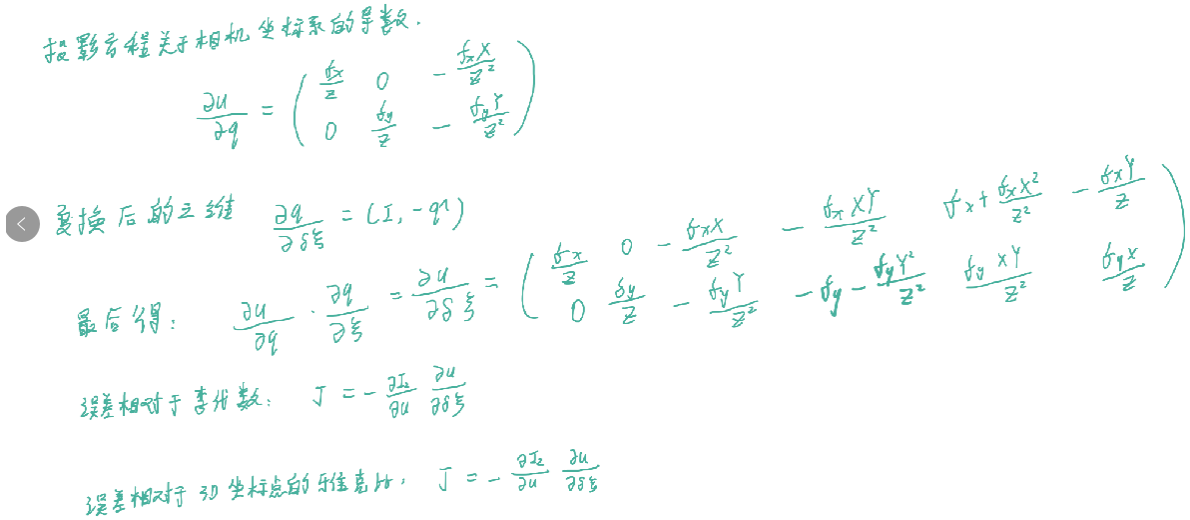
3.1.1 如何描述任意一点投影在任意一图像中形成的error?

$$\text{error} = \mathrm{I}(\mathrm{p_i}) - \mathrm{I_j}(\pi(\mathrm{K}\mathrm{T_J}\mathrm{p_i}))$$

3.1.2 每个error 关联几个优化变量？

关联两个优化变量，分别是位姿和路标点。也就是相机的李代数和三位空间坐标点P(x,y,z)。

3.1.3 error关于各变量的雅克比是什么？



3.2 实现

3.2.1 能否不要以 [x, y, z] 的形式参数化每个点？

还可以用逆深度的方法来参数化路标点，该形式能够建模无穷远点，在实际应用中，逆深度也具有更好的数值稳定性。

3.2.2 取4x4 的patch 好吗？取更大的patch 好还是取小一点的patch 好？

patch过小会导致鲁棒性低，过大会导致计算量过大，综合来看patc取 $4*4$是一个较为合适的大小

3.2.3 从本题来看，你看到直接法和特征点法在BA阶段有何不同？

直接法计算的是光度误差

特征点法计算的是重投影误差

3.2.4 由于图像的差异，你可能需要鲁棒核函数，例如Huber的阀值如何选取？

根据经验来选择

实验代码

运行结果