深蓝学院
shenlanxueyuan.com

从零开始手写VIO_第六期

第4章作业思路分享

基于滑动窗口算法的VIO 系统

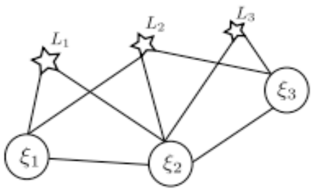
可观性和一致性

0. 作业内容

深蓝学院 shenlanxueyuan.com

作业

1. 某时刻，SLAM 系统中相机和路标点的观测关系如下图所示，其中 ξ 表示相机姿态，L 表示观测到的路标点。当路标点 L 表示在世界坐标系下时，第 k 个路标被第 i 时刻的相机观测到，重投影误差为 $\mathbf{r}(\xi_i, L_k)$。另外，相邻相机之间存在运动约束，如 IMU 或者轮速计等约束。



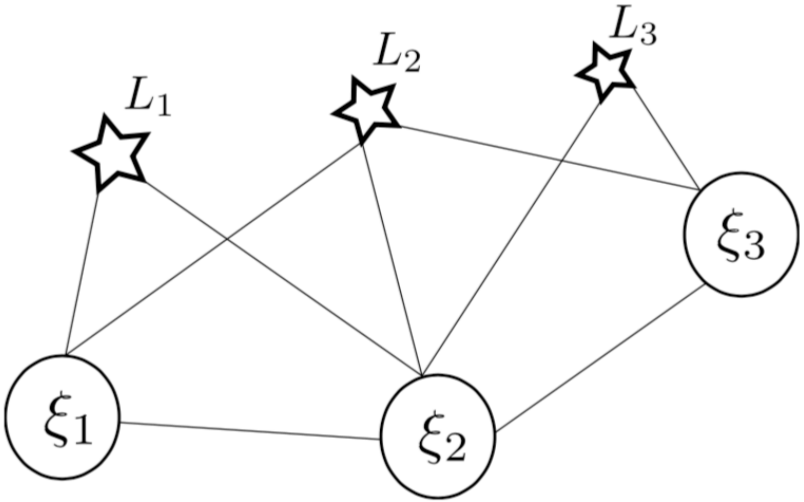
   1. 请绘制上述系统的信息矩阵 $\mathbf{\Lambda}$.
   2. 请绘制相机 ξ_1 被 marg 以后的信息矩阵 $\mathbf{\Lambda}'$.

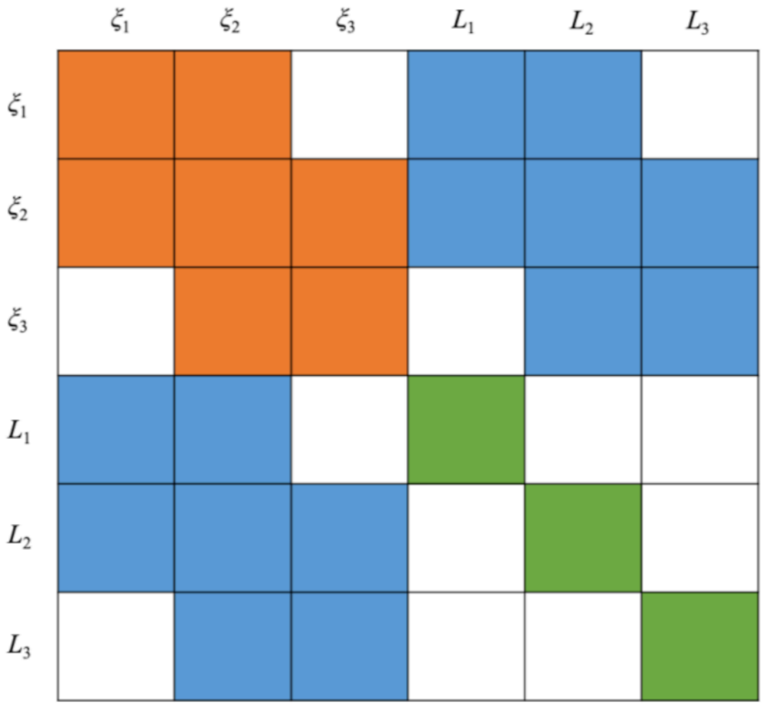
2. 阅读《Relationship between the Hessian and Covariance Matrix for Gaussian Random Variables》. 证明信息矩阵和协方差的逆之间的关系。

3. 请补充作业代码中单目 Bundle Adjustment 信息矩阵的计算，并输出正确的结果。正确的结果为：奇异值最后 7 维接近于 0，表明零空间的维度为 7.

1. 信息矩阵与边缘化

1.1

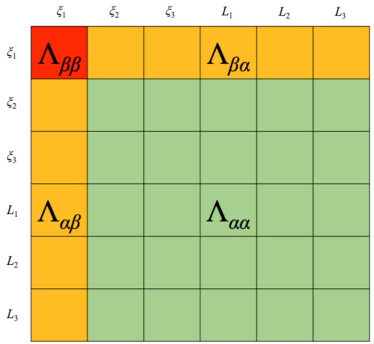




1. 信息矩阵与边缘化

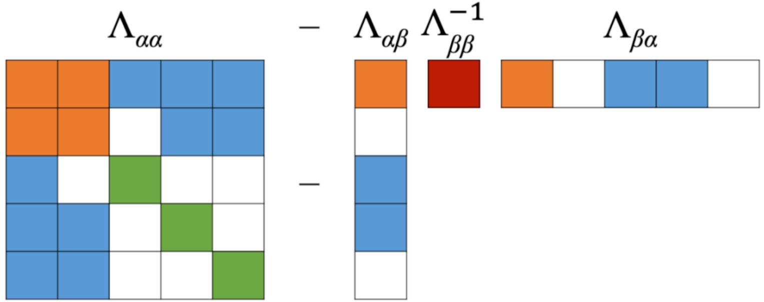
深蓝学院
shenlanxueyuan.com

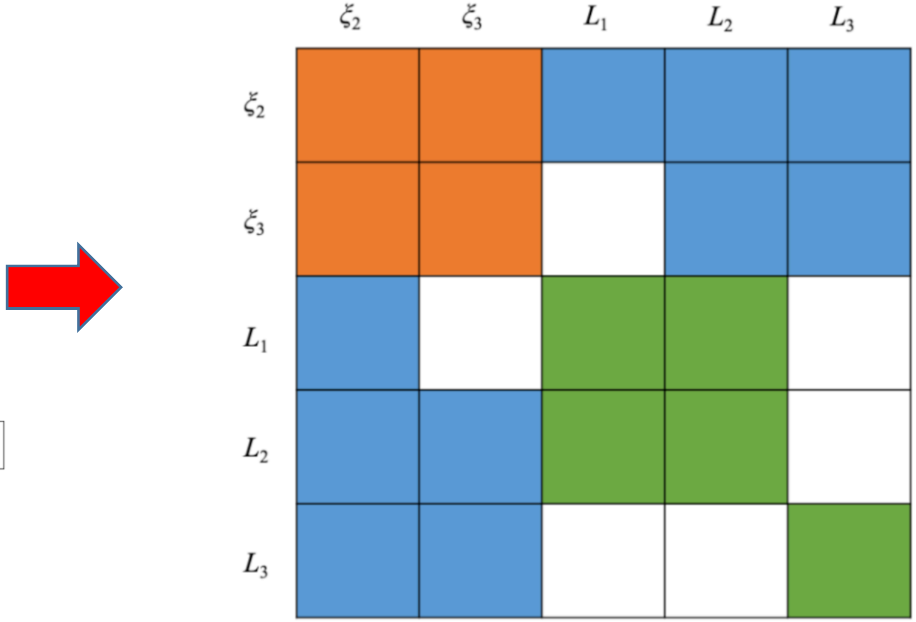
1.2



$$\Lambda' = \Lambda_{\alpha\alpha} - \Lambda_{\alpha\beta}\Lambda_{\beta\beta}^{-1}\Lambda_{\beta\alpha} \quad (1)$$

即有：





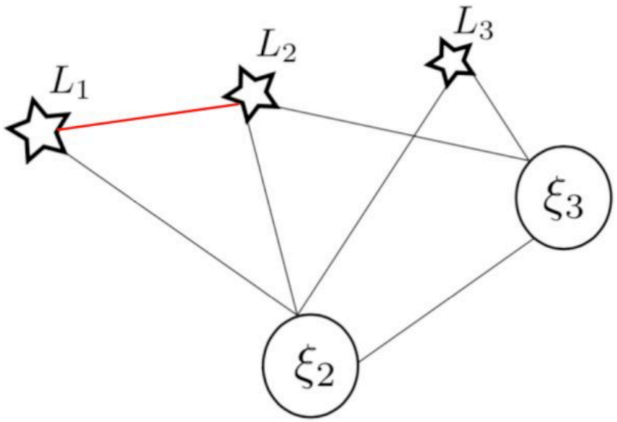
1. 信息矩阵与边缘化

深蓝学院
shenlanxueyuan.com

1.2

```cpp
#include <iostream>
#include <Eigen/Core>
#include <Eigen/Dense>

int main(int argc, char **argv) {
    Eigen::Matrix<double, 6, 6> matInfo_pre;
    Eigen::Matrix<char, 6, 6> matInfo_pre_show;
    matInfo_pre << 1, 1, 0, 1, 1, 0,
                   1, 1, 1, 1, 1, 1,
                   0, 1, 1, 0, 1, 1,
                   1, 1, 0, 1, 0, 0,
                   1, 1, 1, 0, 1, 0,
                   0, 1, 1, 0, 0, 1;
    for (int i = 0; i < matInfo_pre.rows(); ++i) {
        for (int j = 0; j < matInfo_pre.cols(); ++j) {
            if (matInfo_pre(i, j) == 0)
                matInfo_pre_show(i, j) = 'O';
            else
                matInfo_pre_show(i, j) = '#';
        }
    }
    std::cout << "Previous infomation matrix before marginalization = \n" << matInfo_pre_show.matrix() << std::endl;

    // marginalization parts
    Eigen::Matrix<double, 5, 5> mat_aa = matInfo_pre.block<5, 5>(1, 1);
    Eigen::Matrix<double, 1, 1> mat_bb = matInfo_pre.block<1, 1>(0, 0);
    Eigen::Matrix<double, 5, 1> mat_ab = matInfo_pre.block<5, 1>(1, 0);
    Eigen::Matrix<double, 1, 5> mat_ba = mat_ab.transpose();
    Eigen::Matrix<double, 5, 5> mat_abbbba = mat_ab * mat_bb.inverse() * mat_ba;

    Eigen::Matrix<double, 5, 5> matInfo_new = Eigen::Matrix<double, 5, 5>::Zero();
    Eigen::Matrix<char, 5, 5> matInfo_new_show = Eigen::Matrix<char, 5, 5>::Zero();
    for (int i = 0; i < matInfo_new.rows(); ++i) {
        for (int j = 0; j < matInfo_new.cols(); ++j) {
            if (mat_aa(i, j) != 0 || mat_abbbba(i, j) != 0) {
                matInfo_new(i, j) = 1.0;
                matInfo_new_show(i, j) = '#';
            }
            else {
                matInfo_new(i, j) = 0.0;
                matInfo_new_show(i, j) = 'O';
            }
        }
    }
    std::cout << "\nNew infomation matrix after marginalization = \n" << matInfo_new_show.matrix() << std::endl;

    return 0;
}
```

```
Previous infomation matrix before marginalization =
# # O # # O
# # # # # #
O # # O # #
# # O # O O
# # # O # O
O # # O O #

New infomation matrix after marginalization =
# # # # #
# # O # #
# O # # O
# # # # O
# # O O #
```



2. 阅读论文与证明

对于多维 Gaussian 分布，其概率密度函数 (Probability Density Function, PDF) 如下式：

$$P(x) = \frac{1}{\sqrt{(2\pi)^N \det(\Sigma)}} \exp(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)) \qquad x \sim N(\mu, \Sigma)$$

因此对其求最大估计时等效于对其负对数求最小估计，亦即：

$$-\ln(P(x)) = \frac{1}{2}\ln\left((2\pi)^N \det(\Sigma)\right) + \frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)$$

同时，目标函数（即原 Gaussian 分布负对数）的 Hessian 矩阵（亦即目标函数的信息矩阵）为：

$$H = H^{(l,l')}(\theta^*) = J^T J = \left.\frac{\partial^2 J(\boldsymbol{\theta})}{\partial\theta_l \partial\theta_{l'}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}^*} = (\boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1})^{(l,l')}$$

故有：以 Gaussian 分布负对数为目标函数的 Hessian 矩阵（信息矩阵）等于原 Gaussian 分布的协方差矩阵的逆，如下：

$$\mathcal{H}(\boldsymbol{\theta}^\star) = \boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1}$$

此结论对于 Gaussian 分布是有效的，对于其他分布形式未必成立。而当 Gaussian 分布均值为状态的非线性函数时，可利用线性化将其展开近似为线性函数，同理可利用原 Gaussian 分布协方差矩阵的逆近似估计其负对数的 Hessian 矩阵（信息矩阵）。

3. nullspace_test

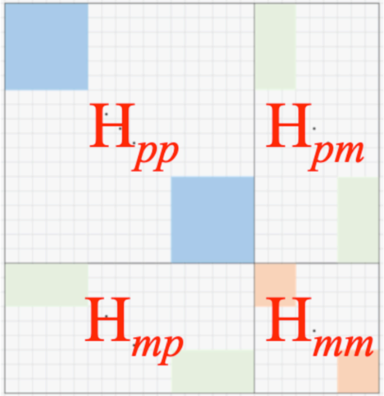
a. 残差对位姿的 Jacobian：

$$\frac{\partial e}{\partial \delta\xi} = -\begin{bmatrix} \frac{f_x}{Z'} & 0 & -\frac{f_x X'}{Z'^2} & -\frac{f_x X' Y'}{Z'^2} & f_x + \frac{f_x X^2}{Z'^2} & -\frac{f_x Y'}{Z'} \\ 0 & \frac{f_y}{Z'} & -\frac{f_y Y'}{Z'^2} & -f_y - \frac{f_y Y'^2}{Z'^2} & \frac{f_y X' Y'}{Z'^2} & \frac{f_y X'}{Z'} \end{bmatrix}$$

b. 残差对路标点坐标的 Jacobian：

$$\frac{\partial e}{\partial P} = -\begin{bmatrix} \frac{f_x}{Z'} & 0 & -\frac{f_x X'}{Z'^2} \\ 0 & \frac{f_y}{Z'} & -\frac{f_y Y'}{Z'^2} \end{bmatrix} R$$

```
// information matrix (hessian matrix)
/// 请补充完整作业信息矩阵块的计算
H.block(i*6,i*6,6,6) += jacobian_Ti.transpose() * jacobian_Ti;
H.block(i*6, 6*poseNums + j*3, 6, 3) += jacobian_Ti.transpose() * jacobian_Pj;
H.block(6*poseNums + j*3, i*6, 3, 6) += jacobian_Pj.transpose() * jacobian_Ti;
H.block(6*poseNums + j*3, 6*poseNums + j*3, 3, 3) += jacobian_Pj.transpose() * jacobian_Pj;
```




```
0.00230246
0.00172459
0.000422374
3.21708e-17
2.06732e-17
1.43188e-17
7.66992e-18
6.08423e-18
6.05715e-18
3.94363e-18
```

7

一文看尽4种SLAM中零空间的维护方法：https://zhuanlan.zhihu.com/p/341322063

深蓝学院
shenlanxueyuan.com

感谢各位聆听！

Thanks for Listening