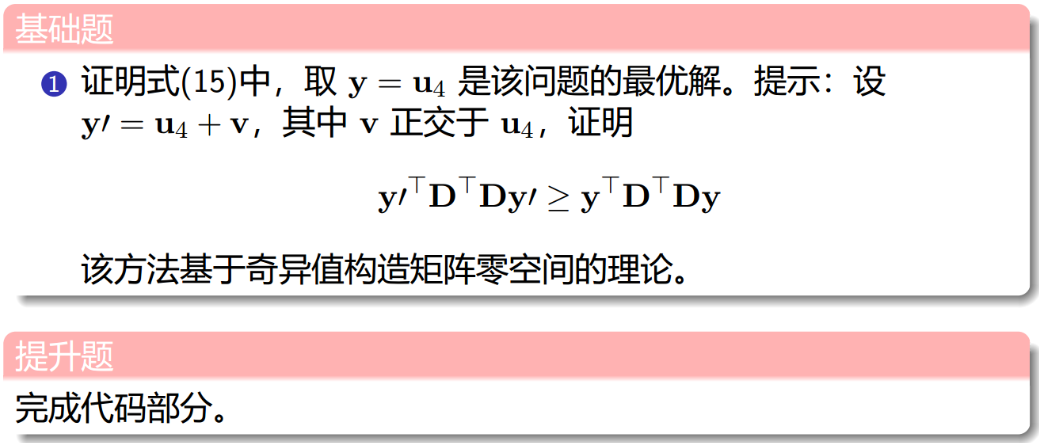
**深蓝学院VIO课程第六课作业**



**1. 证明公式**

D是14×4的矩阵，易知D的秩为3，则D存在1维零空间。的解即是D零空间中的向量，故是的最优解。也可以用公式证明：

结合公式(14)(15)有：

当 时，有：

由于半正定，有：

故有：

由于奇异值由大到小排列，故是最优解。

**2. 完成三角化代码**

补充的代码如下所示，详情可见附件triangulate.cpp文件。程序运行结果所图1所示。

|  |  |
| --- | --- |
| 25  …  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  … | int main() {  …  /\* your code begin \*/  Eigen::Matrix<double, 14, 4> D;  for (int i = start\_frame\_id; i < end\_frame\_id; ++i) { // 3~9  double ui = camera\_pose[i].uv[0];  double vi = camera\_pose[i].uv[1];  // 这里R,t是World->Camera的投影  Eigen::Matrix<double, 3, 4> P;  P.block(0, 0, 3, 3) = camera\_pose[i].Rwc.inverse();  P.block(0, 3, 3, 1) = -camera\_pose[i].Rwc.inverse()\*camera\_pose[i].twc;  int idx = 2 \* (i - start\_frame\_id);  D.block(idx, 0, 1, 4) = ui \* P.row(2) - P.row(0);  D.block(idx+1, 0, 1, 4) = vi \* P.row(2) - P.row(1);  }  Eigen::JacobiSVD<Eigen::MatrixXd> svd(D.transpose()\*D, Eigen::ComputeThinU | Eigen::ComputeThinV);  cout << "SVD singular values: \n" << svd.singularValues().transpose() << endl;  Eigen::Matrix4d U = svd.matrixU(); // 这里 U = V  P\_est = U.col(3).head(3) / U.col(3)[3];  /\* your code end \*/  … |

从图1的程序运行结果截图中可以看到，经过SVD分解后求得的奇异值分别为：468.406、7.74642、0.723255、8.2864e-16。最后一维的奇异值与前三维差距特别大，值接近于0，表明这是D的零空间。由课件资料可知，矩阵D的零空间就的解的集合。D的零空间只有最后一维，故取U的最后一列（代码88行）作为y的解。

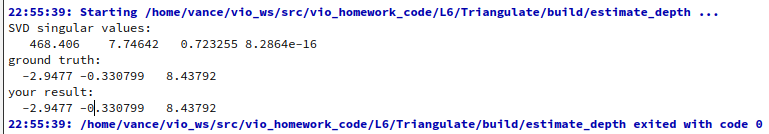


图1 estimate\_depth程序运行结果截图