**2018年第2次课习题答案**

**——Vance吴方熠**

**2. 熟悉 Eigen矩阵运算**

**设线性方程，在*A*为方阵的前提下，请回答以下问题：**

**2.1. 在什么条件下，*x*有解且唯一？**

有唯一解的充分必要条件：

① .

② 且b可由**A**的列向量线性表示.

**2.2. 高斯消元法的原理是什么？**

高斯消元法是一个古老的求解线性方程组的方法，用其解线性方程的基本思想是：用逐次消去未知数的方法把原线性方程组化为与其等价的三角形线性方程组。简单来说，就是对系数矩阵A施行一些左边换（用一些简单矩阵）将其约化为上三角矩阵。

高斯消元法定理：

如果，则可通过高斯消元法将约化为等价的三角形线性方程组：

且计算公式为：

①消元计算（）

②回代计算

高斯消元法对某些简单的矩阵可能会失败，例如.若要保证高斯消元法可用，需保证**A**的顺序主子式.

**2.3. QR分解的原理是什么？**

QR分解是计算一般矩阵（中小型矩阵）全部特征值问题的最有效方法之一，目前主要用来计算：①上海森伯格矩阵的全部特征值问题；②计算对称三对角矩阵的全部特征值问题。QR方法具有收敛快，算法稳定等特点。

一般矩阵将A正交相似变化成为Hessenberg矩阵B，然后再应用QR方法求B的特征值和特征向量。QR分解将矩阵分解成一个正规正交矩阵Q与上三角形矩阵R，所以称为QR分解法，与此正规正交矩阵的通用符号Q有关。

**2.4. Cholesky分解的原理是什么？**

Cholesky 分解是把一个对称正定的矩阵表示成一个下三角矩阵L和其转置的乘积的分解。它要求矩阵的所有特征值必须大于零，故分解的下三角的对角元也是大于零的。Cholesky分解法又称**平方根法**，是当A为实对称正定矩阵时，LU三角分解法的变形。

**2.5. 编程实现A为100\*100随机矩阵时，用QR和Cholesky分解求x的程序。**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | #include <iostream>  #include <cstdlib>  #include <ctime>  #include <Eigen/Core>  #include <Eigen/Cholesky>  #include <Eigen/QR>  **using** **namespace** std**;**  int main**(**int argc**,** char**\*\*** argv**)**  **{**  srand**((**unsigned**)**time**(NULL));**  Eigen**::**MatrixXd A **=** Eigen**::**MatrixXd**::**Random**(**100**,** 100**);**  Eigen**::**VectorXd b **=** Eigen**::**VectorXd**::**Random**(**100**);**  Eigen**::**VectorXd x1 **=** **(**A**.**adjoint**()\***A**).**ldlt**().**solve**(**A**.**adjoint**()\***b**);** // 正或负半定  Eigen**::**VectorXd x2 **=** **(**A**.**adjoint**()\***A**).**llt**().**solve**(**A**.**adjoint**()\***b**);** // 正定  Eigen**::**VectorXd x3 **=** A**.**householderQr**().**solve**(**b**);**  Eigen**::**VectorXd x4 **=** A**.**colPivHouseholderQr**().**solve**(**b**);**  Eigen**::**VectorXd x5 **=** A**.**fullPivHouseholderQr**().**solve**(**b**);**  cout **<<** "x1(using ldlt Cholesky):" **<<** endl **<<** x1 **<<** endl**;**  cout **<<** "x2(using llt Cholesky):" **<<** endl **<<** x2 **<<** endl**;**  cout **<<** "x3(using householderQr):" **<<** endl **<<** x3 **<<** endl**;**  cout **<<** "x4(using colPivHouseholderQr):" **<<** x4 **<<** endl**;**  cout **<<** "x5(using fullPivHouseholderQr):" **<<** x5 **<<** endl**;**  **return** 0**;**  **}** |

程序代码如下：

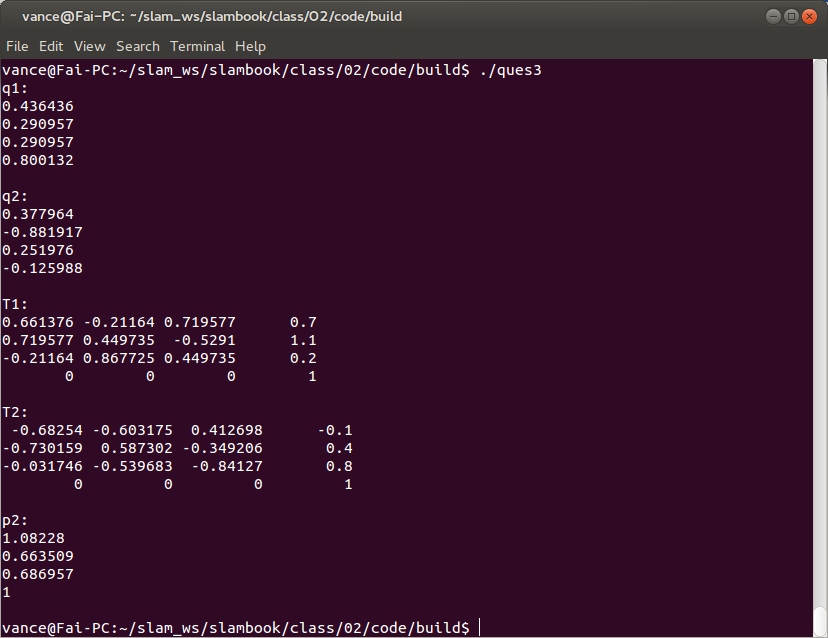
**3. 几何运算练习**

**设有小萝卜1一号和小萝卜二号位于世界坐标系中。小萝卜一号的位姿为（*q* 的第一项为实部）。这里的 *q* 和 *t* 表达的是，也就是世界到相机的变换关系。小萝卜二号的位姿为 。现在，小萝卜一号看到某个点在自身的坐标系下，坐标为**，**求该向量在小萝卜二号坐标系下的坐标。请编程实现此事，并提交你的程序。**

程序代码如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | #include <iostream>  #include <Eigen/Core>  #include <Eigen/Dense>  **using** **namespace** std**;**  int main**(**int argc**,** char**\*\*** argv**)**  **{**  Eigen**::**Quaternion**<**float**>** q1**(**0.55**,** 0.3**,** 0.2**,** 0.2**);**  Eigen**::**Quaternion**<**float**>** q2**(-**0.1**,** 0.3**,** **-**0.7**,** 0.2**);**  q1 **=** q1**.**normalized**();**  q2 **=** q2**.**normalized**();**  Eigen**::**Vector3f t1**(**0.7**,** 1.1**,** 0.2**);**  Eigen**::**Vector3f t2**(-**0.1**,** 0.4**,** 0.8**);**  Eigen**::**Matrix**<**float**,**4**,**1**>** p1**,** p2**;**  p1 **<<** 0.5**,** **-**0.1**,** 0.2**,** 1**;**  Eigen**::**Isometry3f T1 **=** Eigen**::**Isometry3f**::**Identity**();**  Eigen**::**Isometry3f T2 **=** Eigen**::**Isometry3f**::**Identity**();**  T1**.**rotate**(**q1**);** // q1.toRotationMatrix()  T1**.**pretranslate**(**t1**);**  T2**.**rotate**(**q2**);**  T2**.**pretranslate**(**t2**);**  p2 **=** T2 **\*** T1**.**inverse**()** **\*** p1**;**  cout **<<** "p2:" **<<** endl **<<** p2 **<<** endl **<<** endl**;**  **return** 0**;**  **}** |

程序运行结果截图：



结果与答案一致。

**4. 旋转的表达**

**4.1.设有旋转矩阵*R*，证明 且** 。

设一个三维坐标系发生了旋转，变成了，则旋转矩阵R可表示为：

则有：

因为，所以有:

故。

当时为瑕疵转（Improper rotation），即物体绕某轴旋转后，又对垂直于该轴的平面做反射。故这里取。

**4.2. 设有四元数 *q*，我们把虚部记为，实部记为 ，那么 。请说明 和的维度。**

：三维；：一维。

**4.3. 定义运算和 为：**

**请证明对任意单位四元数，四元数乘法可写成矩阵乘法：**

证明：

设,，其中.由《视觉SLAM十四讲》公式（3.24）可得：

|  |  |
| --- | --- |
|  | （4-1） |

由给定条件带入计算两个等式的右边，有：

|  |  |
| --- | --- |
|  | （4-2） |
|  | （4-3） |

而对于叉乘运算有：

|  |  |
| --- | --- |
|  | （4-4） |

将（4-4）带入至（4-2）与（4-3）中，可得：

|  |  |
| --- | --- |
|  | （4-5） |

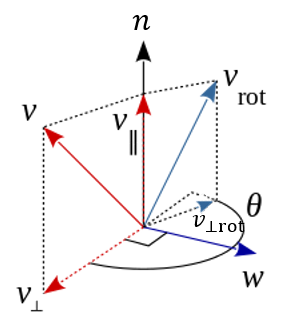
**5. 罗德里格斯公式的证明**

证明：

如下图所示，有一向量经过旋转轴的旋转后，变到了的位置。其中为单位向量，转过的角度为。向量可以由其平行于旋转轴的投影和垂直旋转轴的投影表示：**，**其中：

由于，所以：

**.**

同理，向量也可以表示为：，其中：

则联立以上各式，可得：

又,所以：

证毕。

**6. 四元素运算性质的验证**

**课程中介绍了单位四元数可以表达旋转。其中，在谈论用四元数*q* 旋转点 *p* 时，结果为：**

**我们说，此时必定为虚四元数（实部为零）。请你验证上述说法。**

**此外，上式亦可写成矩阵运算：。请根据你的推导，给出矩阵*Q*。注意此时*p*和都是四元数形式的变量，所以*Q* 为4×4的矩阵。**

（1）证明：

点可由一个虚四元数来表示。首先证明

|  |  |
| --- | --- |
|  | （6-1） |

对（6-1）有：

（6-1）得证。再根据（6-1）的性质，可将视为单位四元素（若不是单位四元素，可令其乘一常数使化为单位四元素）。则根据单位四元素的性质，有：

令函数，可知为求实部标量的函数。则有：

|  |  |
| --- | --- |
|  | （6-2） |

由于的值为标量，故式（6-2）可写成：

|  |  |
| --- | --- |
|  | （6-3） |

即与的实部值相等，均为0。故为虚四元数，得证。

（2）求解Q：

令单位四元素，其中向量为虚部，为实部。使用第4题结果，有：

从而可以导出四元数至旋转矩阵的转换⽅式：

其中：

故最后

为4×4的矩阵。

**7. \*熟悉C++11**

**设有类A，并有A类的⼀组对象，组成了⼀个vector。现在希望对这个vector进行排序，但排序的方式由*A.*index成员大小定义。那么，在C++11的语法下，程序写成：**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | #include <iostream>  #include <vector>  #include <algorithm>  **using** **namespace** std**;**  class A **{**  public**:**  A**(**const int**&** i **)** **:** index**(**i**)** **{}**  int index **=** 0**;** // C++11 面向对象增强  **};**  int main**()** **{**  A a1**(**3**),** a2**(**5**),** a3**(**9**);**  vector**<**A**>** avec**{**a1**,** a2**,** a3**};** // C++11 初始化列表  std**::**sort**(**avec**.**begin**(),** avec**.**end**(),** **[](**const A**&**a1**,** const A**&**a2**)** **{return** a1**.**index **<** a2**.**index**;});** // C++11 Lambda表达式  **for** **(** auto**&** a **:** avec **)** cout **<<** a**.**index **<<** " "**;** // C++11 区间迭代for循环  cout **<<** endl**;**  **return** 0**;**  **}** |

**请说明该程序中哪些地方用到了C++11标准的内容。**

* 第10行，面对对象增强部分
* 第15行，初始化列表
* 第16行，Lambda表达式
* 第17行，区间迭代for循环