视觉SLAM中的数学基础 第二篇 四元数

什么是四元数

相比欧拉角，四元数(Quaternion)则是一种紧凑、易于迭代、又不会出现奇异值的表示方法。它在程序中广为使用，例如ROS和几个著名的SLAM公开数据集、g2o等程序都使用四元数记录机器人的姿态。因此，理解四元数的含义与用法，对学习SLAM来说是必须的。本节我们就来讲讲四元数。

首先，请读者不要对四元数有什么神秘的感觉。四元数仅是3D姿态的一种表达方式，我们用一个单位四元数表达原本用旋转矩阵表示的三维旋转。这样做一个直接的好处是省空间。一个旋转阵有9个分量，但只有三个自由度。那么，能不能用三个数来描述呢？可以是可以的，但不可避免会出现奇异的情况，欧拉角就是一个例子。而四元数，比三维向量多了一个分量，从而可以无奇异地表示各种姿态。下面我们来详细讲讲四元数。

四元数是Hamilton找到的一种扩展的复数。一个四元数拥有一个实部和三个虚部(故事上说他原先找了很久带两个虚部的，结果怎么也找不到，最后豁然开朗找到了三虚部的四元数)：

$$= {}_0 + {}_1 + {}_2 + {}_3$$

其中，， 为四元数的三个虚部。这三个虚部满足关系式：

$$\begin{aligned} {}^2 = {}^2 = {}^2 = -1 \\ = , = - \\ = , = - \\ = , = - \end{aligned} \tag{1}$$

由于它的这种特殊表示形式，有时人们也用一个标量和一个向量来表达四元数：

$$= [,], \quad = {}_0 \in \mathbb{R}, = [{}_1, {}_2, {}_3] \in \mathbb{R}^3.$$

这里，标量 称为四元数的实部，而向量 称为它的虚部。如果一个四元数虚部为$\mathbf{0}$，称之为实四元数。反之，若它的实部为0，称之为虚四元数。该定义和复数是相似的。

四元数可以表示三维空间中任意一个旋转。与旋转矩阵中类似，我们仍假设某个旋转是绕单位向量 $= [\ ,\ ,\ ]$ 进行了角度为 的旋转，那么这个旋转的四元数形式为：

$$= [\cos \frac{}{2}, \sin \frac{}{2}, \sin \frac{}{2}, \sin \frac{}{2}] \tag{2}$$

事实上，这还是一个模长为1的四元数，称为单位四元数。反之，我们亦可通过任意一个长度为1的四元数，计算对应旋转轴与夹角：

$$\begin{cases} = 2 \arccos {}_0 \\ [\ ,\ ,\ ] = [{}_1, {}_2, {}_3] / \sin \frac{}{2} \end{cases} \tag{3}$$

若某个四元数长度不为1，我们可以通过归一化将它转换为一个模长为1的四元数。

对式2的 加上2 ，我们得到一个相同的旋转，但此时对应的四元数变成了$-$ 。因此，在四元数中，任意的旋转都可以由两个互为相反数的四元数表示。同理，取 为0，则得到一个没有任何旋转的四元数：

$${}_0 = [\pm 1, 0, 0, 0] \tag{4}$$

四元数的运算

四元数和通常复数一样，可以进行一系列的运算。常见的有四则运算、内积、求逆、共轭、求指数／对数等等。表示姿态时，它还可以进行插值。下面我们分别介绍。

现有两个四元数 ， ，它们的向量表示为$[\ ,\ ], [\ ,\ ]$，或者原始四元数表示为：

$$+ \quad + \quad + \quad , \quad + \quad + \quad + \quad .$$

那么，它们的运算可表示如下。

- 加法和减法

四元数 ， 的加减运算为：

$$\pm \quad = [\ \pm\ ,\ \pm\ ]. \tag{5}$$

- 乘法

乘法是把 的每一项与 每项相乘，最后相加，虚部要按照式~$\underline{1}$~进行：

$$\begin{aligned} = \quad - \quad - \quad - \\ +(\quad + \quad + \quad - \quad) \\ +(\quad - \quad + \quad + \quad) \\ +(\quad + \quad - \quad + \quad) \end{aligned} \tag{6}$$

虽然稍为复杂，但形式上也是整齐有序的。如果写成向量形式并利用内外积运算，该表达会更加简洁：

$$= [\quad - \quad \cdot \quad , \quad + \quad + \quad \times \quad] \tag{7}$$

这里我们就不帮读者复习什么叫外积了。在该乘法定义下，两个实的四元数乘积仍是实的，这与复数也是一致的。然而，注意到，由于最后一项外积的存在，该乘法通常是不可交换的，除非 和 在$\mathbb{R}^3$中共线。

- 共轭

四元数的共轭为：

$${}^* = \quad - \quad - \quad - \quad = [\ , -\ ] \tag{8}$$

即把虚部取成相反数。四元数共轭与自己本身相乘，会得到一个实四元数，其实部为模长的平方：

$$* = * = [{}^2 + \quad , \mathbf{0}] = {}^2 + \tag{9}$$

- 模长

四元数的模长定义为：

$$\| \quad \| = \sqrt{{}^2 + {}^2 + {}^2 + {}^2} = \sqrt{\quad *} \tag{10}$$

可以验证，两个四元数乘积的模即为模的乘积。这保证单位四元数相乘后仍是单位四元数。

$$\| \quad \| = \| \quad \| \| \quad \| \tag{11}$$

- 逆

一个四元数的逆为：

$${}^{-1} = {}^* / \| \|^2 \tag{12}$$

按此定义，四元数和自己的逆的乘积为实四元数的1：

$${}^{-1} = {}^{-1} = 1 \tag{13}$$

同时，乘积的逆有和矩阵相似的性质：

$$(\quad)^{-1} = {}^{-1} \quad {}^{-1} \tag{14}$$

对于单位四元数，即$\| \| = 1$ ，它的逆即是它的共轭四元数。

- 数乘与点乘

和向量相似，四元数可以与数相乘：

$$= [,] \tag{15}$$

点乘是指两个四元数每个位置上的数值分别相乘：

公告

昵称：半闲居士
园龄：8年10个月
粉丝：3062
关注：0
+加关注

< 2023年1月 >

日	一	二	三	四	五	六
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	1	2	3	4
5	6	7	8	9	10	11

搜索

找找看
谷歌搜索

常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签

我的标签

视觉SLAM(17)
机器人(14)
SLAM(13)
一起做RGB-D SLAM(7)
Kinect(4)
rgbdslam(3)
计算机视觉(2)
图像处理(2)
视觉SLAM漫谈(2)
李群(2)
更多

随笔分类

随笔(2)
一起做rgbd slam(2)

随笔档案

2016年8月(1)
2016年7月(1)
2016年6月(2)
2016年3月(2)
2016年2月(2)
2016年1月(8)
2015年12月(1)
2015年8月(4)
2015年7月(4)
2015年4月(2)
2014年6月(1)
2014年4月(1)

阅读排行榜

1. 视觉SLAM漫谈(211589)
2. 一起做RGB-D SLAM (1)(137824)
3. 一起做RGB-D SLAM (2)(118148)
4. 深入理解图优化与g2o:g2o篇(115570)
5. 视觉SLAM实战(一):RGB-D SLAM V2(110972)

评论排行榜

1. 一起做RGB-D SLAM (2)(77)
2. 一起做RGB-D SLAM (5)(66)
3. 一起做RGB-D SLAM (3)(66)
4. 一起做RGB-D SLAM (6)(64)
5. 一起做RGB-D SLAM (4)(62)

推荐排行榜

1. 视觉SLAM漫谈(71)
2. 一起做RGB-D SLAM (2)(34)
3. 一起做RGB-D SLAM (1)(31)
4. 深入理解图优化与g2o:g2o篇(25)
5. 视觉SLAM漫谈 (三): 研究点介绍(25)

最新评论

1. Re:一起做RGB-D SLAM (5)
@going_go 你好 我现在也是碰到这个问题。想问下大佬是怎么解决的...
--养朱的钊
2. Re:一起做RGB-D SLAM (4)
大家好，我在执行bin/detectfeatures时出现以下错误了oodboy@goodboy-virtual-machine:~/slam$ bin/detectFeatures bin/dete...
--啊啦啦啦看
3. Re:一起做RGB-D SLAM (1)
博主，您好。我在执行第三部分bin/detectFeatures时出现bin/detectFeatures: symbol lookup error: bin/detectFeatures: unde...
--啊啦啦啦看
4. Re:视觉SLAM实战(二):ORB-SLAM2 with Kinect2
大佬你好，看了您的文章，非常钦佩您VSLAM实战教程这方面的讲解，有兴趣合作成为我们古月居网站的

$$\cdot \quad = \quad + \quad + \quad + \tag{16}$$

用四元数表示旋转

在复数域$\mathbb{C}$，我们可以用一个复数　表示2D的旋转，类似的，3D空间也可以用单位四元数表示旋转。假设一个空间三维点 $=[,,]\in\mathbb{R}^3$ ，以及一个由旋转轴和夹角，　指定的旋转，下面讨论如何用四元数表示它们。

首先，我们把三维空间点用一个虚四元数来描述：

$$=[0,,,]=[0,].$$

然后，参照式2，用另一个四元数　表示这个旋转：

$$=[\cos\frac{}{2},\ \sin\frac{}{2}].$$

那么，旋转后的点 $'$ 即可表示为这样的乘积：

$$= \quad ^{-1} \tag{17}$$

可以验证，计算结果的实部为 $(\times)=0$ ，故计算结果为纯虚四元数。其虚部的三个分量表示旋转后3D点的坐标。

四元数到旋转矩阵的转换

由于任意单位四元数都可表示为一个3D旋转，即(3) 中的元素，我们可以找到一个旋转矩阵与之对应。最简单的方式是由四元数 解出旋转角 和旋转轴 ，但那样要计算一个arccos函数，代价较大。实际上这个计算是可以通过一定的计算技巧绕过的。为省略篇幅，我们直接给出四元数到旋转矩阵的转换方式。

设四元数 $= \ _0 + \ _1 + \ _2 + \ _3$ ，对应的旋转矩阵 为：

$$= \begin{matrix} 1-2\ _2^2-2\ _3^2 & 2\ _1\ _2+2\ _0\ _3 & 2\ _1\ _3-2\ _0\ _2 \\ 2\ _1\ _2-2\ _0\ _3 & 1-2\ _1^2-2\ _3^2 & 2\ _2\ _3+2\ _0\ _1 \\ 2\ _1\ _3+2\ _0\ _2 & 2\ _2\ _3-2\ _0\ _1 & 1-2\ _1^2-2\ _2^2 \end{matrix} \tag{18}$$

反之，由旋转矩阵到四元数的转换如下。假设矩阵为 $=\{\ \},,\in[1,2,3]$ ，其对应的四元数 由下式给出：

$$_0=\frac{\sqrt{\ ()+1}}{2},\ _1=\frac{_{23}-\ _{32}}{4\ _0},\ _2=\frac{_{31}-\ _{13}}{4\ _0},\ _3=\frac{_{12}-\ _{21}}{4\ _0} \tag{19}$$

值得一提的是，由于和一 表示同一个旋转，事实上一个 的四元数表示并不是惟一的。存在其他三种与上式类似的计算方式，而本书省略了。实际编程中，当 $_0$ 接近0时，其余三个分量会非常大，导致解不稳定，此时会考虑使用剩下的几种方式计算。

其他几种变换

3D空间中的变换，除了欧氏变换之外，还存在其他几种变换（事实上欧氏变换是最简单的）。它们有一部分和测量几何有关，我们之后的讲解中会提到，在此先罗列出来。

- 相似变换

相似变换比欧氏变换多了一个自由度，它允许物体进行自由地缩放。

$$= \begin{bmatrix} \mathbf{0} & 1 \end{bmatrix} \tag{20}$$

注意到旋转部分多了一个缩放因子 ，它在,, 三个坐标上形成均匀的缩放。类似的，相似变换的乘法也构成群，称为(3) 。由于含有缩放，相似变换不再保持图形的面积不变。

- 仿射变换

仿射变换的矩阵形式如下：

$$= \begin{bmatrix} \mathbf{0} & 1 \end{bmatrix} \tag{21}$$

与欧氏变换不同的是，仿射变换只要求 是一个可逆矩阵，而不必是正交矩阵。在仿射变换下，直线的夹角会发生改变，但平行性质不变。这即是说，仿射变换把平行四边形变为平行四边形。

- 射影变换

射影变换是最一般的变换，它的矩阵形式为：

$$= \begin{bmatrix} \quad \end{bmatrix} \tag{22}$$

它左上角为可逆矩阵 ，右上为平移 ，左下缩放 。由于采用齐坐标，当 $\neq 0$ 时，我们可以对整个矩阵除以 得到一个右下角为1的矩阵；否则，则得到右下角为0的矩阵。因此，这个矩阵在2D中一共有8个自由度，而在3D中一共有15个自由度，是现在提到的变换中最为一般的。

下表总结了目前讲到的几种变换的性质。注意在"不变性质"中，从上到下是有包含关系的。例如，欧氏变换除了保体积之外，也具有保平行、相交等性质。

表 3-1　常见变换性质比较

变换名称	矩阵形式	自由度	不变性质
欧氏变换	$\begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}$	6 自由度	体积
相似变换	$\begin{bmatrix} sR & t \\ 0^T & 1 \end{bmatrix}$	7 自由度	体积
仿射变换	$\begin{bmatrix} A & t \\ 0^T & 1 \end{bmatrix}$	12 自由度	平行性、体积比
射影变换	$\begin{bmatrix} A & t \\ a^T & v \end{bmatrix}$	15 自由度	接触平面的相交和相切

« 上一篇：视觉SLAM的数学基础 第一篇 3D空间的位置表示
» 下一篇：视觉SLAM中的数学基础 第三篇 李群与李代数

· C#开发的磁吸屏幕类库 - 开源研究系列文章
· SQLSERVER 居然也能调 C# 代码 ?
· MongoDB从入门到实战之.NET Core使用MongoDB开发ToDoList系统(2)-Sw