视觉 SLAM 笔记 *

易鹏

中山大学

内部版本号：V1.12.028 (内测版)

2022 年 10 月 12 日

*本笔记已经开源，可以免费下载，github 地址：https://github.com/Lovely-XPP/Notebook

gitee 分流地址：https://gitee.com/sysu_xpp/Notebook

目录

符号说明

$\boldsymbol{E}_i$	$i \times i$ 的单位矩阵 , 19
$\boldsymbol{R}_{ba}$	坐标系 S_a 中的向量分量变换到坐标系 S_b 的旋转矩阵 , 19
$\boldsymbol{C}_{ba}$	坐标系 S_a 变换为坐标系 S_b 的方向余弦矩阵（坐标系旋转矩阵） , 14
$\boldsymbol{e}$	三维向量基 $\boldsymbol{e}$, 11
$\boldsymbol{u}$	三维向量 $\boldsymbol{u}$, 11
$\|q\|$	四元数的模长（范数）
q	四元数 , 20
q^*	四元数 q 的共轭 , 24
q^{-1}	四元数的逆 , 24

第 1 章　SLAM 简介

1.1　SLAM 的目标

SLAM 的全称为 Simultaneous Localization and Mapping，中文翻译为即时定位于建图，所以 SLAM 的目标就是两个：

(1) 我在什么地方？——**定位**

(2) 周围环境怎么样？——**建图**

1.2　视觉 SLAM

1.2.1　SLAM 传感器

机器的定位有两类传感器。

(1) **机器本身携带**：轮式编码器、相机、激光传感器、IMU 等

(2) **安装于环境**：导轨、二维码标识等

由于环境中的传感器受限于环境的条件，而 GPS 在室内没有信号，SLAM 就是为了解决在任意未知环境中进行定位。这里主要讲视觉 SLAM 的工作。

1.2.2　视觉 SLAM 的传感器

视觉 SLAM 主要的传感器就是**相机**，照片本质上是场景在相机的成像平面上的**投影**。它以**二维的形式记录了三维的世界**，在这个过程中丢掉了场景的一个维度：**深度**（距离）。相机主要有三类：

(1) **单目相机**

- 组成：只使用一个摄像头。
- 原理：在单张图像中，无法确定一个物体的真实大小。它可能是一个**很大但很远**的物体，也可能是一个**很近但很小**的物体。所以如果要恢复三维结构，只能改变相机的视角。所以在单目 SLAM 中，我们必须移动相机才能估计它的**运动**（Motion），同时估计场景中物体的远近和大小，称为**结构**（Structure）。从图像的变化可以得到相机的相对运动状态，同时我们还知道：**近处的物体移动块，远处的物体移动慢，极远处（无穷远处）的物体（如太阳、月亮）看上去是不动的**。所以物体在图像上的运动就形成了**视差**（Disparity），通过视差就可以定量判断物体的距离远近。
- 缺点：由于单张图像无法确定深度，所以得到的物体远近仅仅是一个相对值。也就是说，如果把相机的运动和场景放大同样的倍数，单目相机得到的像是一样的。所以，单目 SLAM 估计的轨迹和地图将与真实的

轨迹和地图相差一个因子，即**尺度**（Scale）。由于单目 SLAM 无法仅仅通过图像确定真实尺度，所以又称为**尺度不确定性**（Scale Ambiguity）。

(2) **双目相机**

- 目的：通过某种手段测量物体与相机之间的距离，克服单目相继无法知道距离的缺点。
- 组成：两个单目相机，且两个相机之间的距离（**基线**）已知。
- 原理：和人眼相似，通过左右眼图像的差异判断物体的远近，具体定量用基线确定。
- 缺点：需要大量计算才能近似（不太可靠）估计每一个像素点的深度，且深度的量程和精度受基线和分辨率所限，视差的计算需要消耗大量的计算资源。
- 应用：室内室外

(3) **深度相机**

- 目的：通过某种手段测量物体与相机之间的距离，克服单目相继无法知道距离的缺点。
- 组成：深度相机又称 **RGB-D 相机**，由激光传感器等组成。
- 原理：通过红外结构光或 Time-of-Flight（ToF）原理，像激光传感器那样，主动向物体发射并接受返回的光，测出物体与相机的距离，此为通过物理测量手段进行测量，可以节省大量的计算资源。
- 缺点：测量范围窄、噪声大、视野小、易受日光干扰、无法测量透射物质。
- 应用：室内

1.3 经典的视觉 SLAM 框架

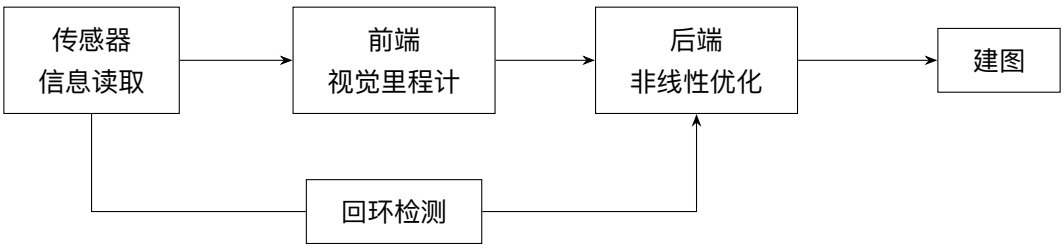

图 1.1: 经典的视觉 SLAM 框架

如图 1.1 所示，经典的视觉 SLAM 框架包括以下几个流程。

1. **传感器信息读取**：相机图像信息的读取和预处理。在机器人中，还可能包括码盘、惯性传感器等信息的读取和同步。
2. **前端视觉里程计**（Visual Odometery，**VO**）：估算相邻图像间相机运动，以及局部地图的样子。VO 又称为**前端**（Front End）。
3. **后端非线性优化**（Optimization）：接受不同时刻视觉里程计测量的相机位姿，以及回环检测的信息，对它们进行优化，得到全局一致的轨迹和地图。由于接在 VO 之后，又称为**后端**（Back End）。
4. **回环检测**（Loop Closure Detection）：回环检测判断机器人是否到达过先前的位置。如果检测到回环，它会把信息提供给后端进行处理。
5. **建图**：根据估计的轨迹，建立与任务要求对应的地图。

在静态、刚体、光照变化不明显、没有人为干扰的场景下，视觉 SLAM 技术已经相当成熟。

1.3.1 视觉里程计

视觉里程计是通过**相邻帧间的图像估计相机运动**，再通过**相机与空间点的几何关系**来估计当前时刻的相机运动，然后将相邻时刻的运动“串起来”，可以估计任意时刻的相机运动并恢复整个场景的空间结构。它称为“里程计”是因为它和实际的里程计一样，只计算相邻时刻的运动（不限于 2 帧，可以是 5 ~ 10 帧），而和过去的信息没有关联。

但是，由于视觉里程计在最简单的情况下只估计极少量图像间的运动，每个时刻的估计都产生一定的误差，导致先前的误差会传递到下一时刻，误差会随时间不断累积，这就是**累积漂移**（Accumulating Drift）。这将导致无法建立一致的地图。为了解决漂移问题，我们还需要后端优化和回环检测。回环检测负责吧“机器人回到原始位置”的事情检测出来，而后端优化则根据该信息，校正整个轨迹的形状。

1.3.2 后端优化

后端优化主要指助理 SLAM 过程中的**噪声**问题。任何传感器都会带一定的噪声，为此我们需要估计传感器带有多大的噪声，这些噪声是如何从上一时刻传递到下一时刻的，而我们又对当前的估计有多大的自信。后端优化要考虑的问题就是如何从这些带有噪声的数据中估计整个系统的状态，以及这个状态轨迹的不确定性有多大——这称为**最大后验概率估计**（Maximum-a-Posteriori, MAP）。这里的状态既包括机器人自身的轨迹，也包含地图。

前端给后端提供待优化的数据，后端只关心数据而不关心数据来自哪里。**在视觉 SLAM 中，前端和计算机视觉研究更为相关，比如图像的特征提取与匹配等，后端则主要是滤波与非线性算法**。

后端：借助状态估计理论，把定位和建图的不确定性表达出来，然后采用滤波器或非线性优化，估计状态的均值和不确定性（方差）。

1.3.3 回环检测

回环检测又称闭环检测，主要解决**位置估计随时间漂移**的问题。这需要让机器人具有**识别到过的场景**的能力。视觉 SLAM 一般采用**判断图像间的相似性**的方法来完成回环检测。所以视觉回环检测实质上是一种计算图像数据相似性的算法。机器人判断回到原点后，将数据重新更新为当时不含累积漂移的数据，同时后端根据这些新的信息，把轨迹和地图调整到符合回环检测结果的样子。所以，如果我们又充分而且正确的回环检测，则可以消除累积误差，得到全局一致的轨迹和地图。

1.3.4 建图

建图是指构件**地图**的过程。地图是对环境的描述，但这个描述并不是固定的，需要视 SLAM 的应用而定。大体上讲，地图可以分为**度量地图**和**拓扑地图**两种。

(1) **度量地图（Metric Map）**

度量地图强调精确地表示地图中物体的位置关系，通常用稀疏（Sparse）与稠密（Dense）对其分类。

- **稀疏地图**：稀疏地图进行了一定程度的抽象，并不需要表达所有物体。例如，我们选择一部分具有代表意义的东西，称之为**路标**（Landmark），那么一张稀疏地图就是由路标组成的地图，而不是路标的部分就可以忽略。

- **稠密地图**：稠密地图着重于建模所有看到的东西。稠密地图通常按照某种分辨率，由许多个小块组成，在二维度量地图中体现为许多个小**格子**（Grid），而在三维度量地图中则体现为许多小**方块**（Voxel）。通常一个小块含有占据、空间、未知三种状态，以表达该格内是否有物体。当查询某个空间位置时，地图能够给出该位置是否可以通过的信息。一般导航就是使用稠密地图。缺点：耗费大量储存空间且很多细节是不需要的，而且大规模度量地图可能会出现一致性问题，很小的误差就可以导致物体重叠，使地图失效。

(2) **拓扑地图（Topological Map）**

相比于度量地图的精确性，拓扑地图更强调地图元素之间的关系。拓扑地图是一个**图**（Graph），由节点和边组成，只考虑节点之间的连通性，而不关注节点之间是如何连通的。

优点：放松了地图对精确位置的需要，去掉了地图的细节，表达更紧凑。

缺点：不擅长表达具有复杂结构的地图，如何将实际地图转换为拓扑地图，如何使用拓扑地图进行导航与路径规划，都是比较困难的。

1.4 SLAM 问题的数学表达

假设一个机器人携带一些传感器在位置环境里面运动。由于传感器是离散采样，记采样的时刻为（连续时间离散化）$t = 1, \cdots, K$，机器人在各个时刻的位置记为 $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_K$。假设地图是由许多个**路标**组成的，传感器在各个时刻测量到一部分路标点。假设路标点一共有 N 个，记为 $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_N$。

我们需要描述的两个问题：运动与观测。

1.4.1 运动的数学表达

什么是运动？即考察 $k-1 \to k$ 时刻，机器人位置 $\boldsymbol{x}$ 的变化情况。机器人的运动状况是由上一个时刻的位置 $\boldsymbol{x}_{k-1}$ 以及运动传感器的读数或输入 $\boldsymbol{u}_k$，当然只要是传感器都会有噪声，所以噪声 $\boldsymbol{\omega}_k$ 也会对当前时刻的位置造成影响。综上，我们可以用一个函数来表示通用、抽象的数学模型。

定义 1.1 运动方程

运动方程定义如下：

$$\boldsymbol{x}_k = f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k, \boldsymbol{\omega}_k) \tag{1.1}$$

噪声的存在使得这个模型变成了随机模型。因为每次运动过程中的噪声是随机的，如果忽略噪声的影响，只根据指令来估计位置，可能与实际位置相差很远。

1.4.2 观测的数学表达

什么是观测？即考察在 k 时刻时机器人在 $\boldsymbol{x}_k$ 处观测到了某一个路标 $\boldsymbol{y}_j$，如何描述这个事件。同样地，我们可以用一个抽象的函数来表示这个数学模型。

定义 1.2 观测方程

观测方程定义如下：

$$\boldsymbol{z}_{k,j} = h(\boldsymbol{y}_j, \boldsymbol{x}_k, \boldsymbol{v}_{k,j}) \tag{1.2}$$

其中，$\boldsymbol{v}_{k,j}$ 是观测中的噪声。

1.4.3 参数化实例

对于不同的真实运动和传感器种类，存在很多**参数化**（Parameterization）方式。例如，假设机器人在平面内运动，那么它的位姿（位置和姿态）由两个位置参数和一个转角参数来描述，即

$$\boldsymbol{x}_k = \begin{bmatrix} x_1 \\ x_2 \\ \theta \end{bmatrix}_k \tag{1.3}$$

其中，x_1, x_2 为两个轴上的位置，而 θ 为转角。同时，输入的指令是由两个时间间隔位置和转角的变化量来描述，即

$$\boldsymbol{u}_k = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta\theta \end{bmatrix}_k \tag{1.4}$$

于是运动凡尘就可以具体化为

$$\begin{bmatrix} x_1 \\ x_2 \\ \theta \end{bmatrix}_k = \begin{bmatrix} x_1 \\ x_2 \\ \theta \end{bmatrix}_{k-1} + \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta\theta \end{bmatrix}_k + \boldsymbol{\omega}_k \tag{1.5}$$

在这种情况下，运动方程就是简单的线性关系。但是，不是所有输入指令都是位移和角度的变化量。例如“油门”或“控制杆”的输入就是速度或加速度量，所以运动方程是多样化的，具体需要进行动力学分析。

关于观测方程，以二维激光传感器为例。激光传感器观测 2D 路标时能够测量两个量：路标点与机器人之间的距离 r 和夹角 ϕ。记路标点，位姿和观测数据分别为

$$\boldsymbol{y}_j = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}_j \qquad \boldsymbol{x}_k = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_k \qquad \boldsymbol{z}_{k,j} = \begin{bmatrix} r_{k,j} \\ \phi_{k,j} \end{bmatrix} \tag{1.6}$$

那么观测方程可以写为

$$\begin{bmatrix} r_{k,j} \\ \phi_{k,j} \end{bmatrix} = \begin{bmatrix} \sqrt{(y_{1,j} - x_{1,k})^2 + (y_{2,j} - x_{2,k})^2} \\ \arctan\left(\dfrac{y_{2,j} - x_{2,k}}{y_{1,j} - x_{1,k}}\right) \end{bmatrix} + \boldsymbol{v}_{k,j} \tag{1.7}$$

考虑视觉 SLAM 时，传感器是相机，那么观测方程就是“对路标点拍摄后，得到图像中的像素”的过程。

所以，针对不同的传感器，运动方程和观测方程有不同的参数化形式。如果我们保持通用性，把它们取成通用的抽象形式，那么 SLAM 过程可总结为两个基本方程。

定义 1.3 SLAM 基本方程

SLAM 的两个基本方程为：

$$\begin{cases} \boldsymbol{x}_k = f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k, \boldsymbol{\omega}_k) & k = 1, \cdots, K \\ \boldsymbol{z}_{k,j} = h(\boldsymbol{y}_j, \boldsymbol{x}_k, \boldsymbol{v}_{k,j}) & (k, j) \in \mathcal{O} \end{cases} \tag{1.8}$$

其中，$\mathcal{O}$ 是一个集合，记录这在哪个时刻观察到了哪个路标，通常不是每个路标在每个时刻都能看到的——我们在单个时刻很可能只看到一小部分。

这两个方程描述了最基本的 SLAM 问题：当制导运动测量的读数 $\boldsymbol{u}$，以及传感器的读数 $\boldsymbol{z}$ 时，如何求解定位问题（估计 $\boldsymbol{x}$）和建图问题（估计 $\boldsymbol{y}$）？这是我们就把 SLAM 问题建模成了一个**状态估计问题**：如何通过带有噪声的测量数据，估计内部的、隐藏着的状态变量。

1.4.4 问题总结

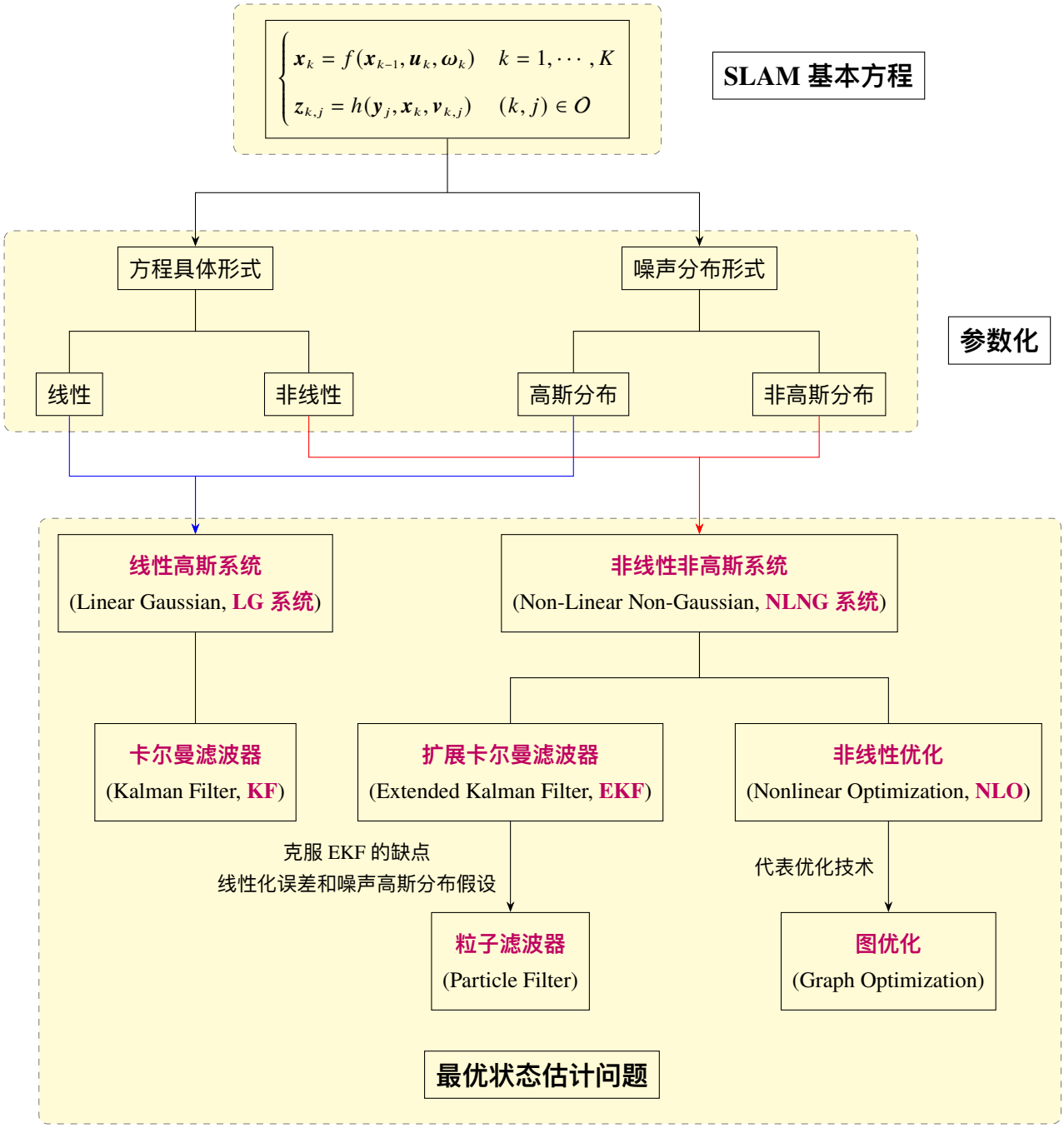

图 1.2: SLAM 问题数学表述总结图

1.5 环境配置与 CMake 基础

基本 Linux 环境：Ubuntu 20.04，安装教程略。

1.5.1 环境配置

1. git

```
sudo apt install git
```

2. vim

```
sudo apt install vim
```

3. Slambook2

```
cd ~ && git clone https://github.com/gaoxiang12/slambook2/
```

4. 编译器

```
sudo apt install gcc g++ cmake
```

5. Eigen3

```
sudo apt install libeigen3-dev
```

6. Pangolin

- 系统依赖

```
sudo apt install libglew-dev libboost-dev libboost-thread-dev libboost-fileystem-dev
```

- 克隆仓库

```
cd 3rdparty # if current dir is 3rdparty, skip this command
git clone https://github.com/stevenlovegrove/Pangolin.git
```

- CMake & Install

```
cd Pangolin
mkdir build && cd build
cmake ..
make -j4
sudo make install
```

7. fmt

- 克隆仓库

```
cd 3rdparty # if current dir is 3rdparty, skip this command
git clone https://github.com/fmtlib/fmt.git
cd fmt
git checkout b6f4ceae
```

- CMake & Install

```
cd fmt # if current dir is fmt, skip this command
mkdir build && cd build
cmake ..
make -j4
sudo make install
```

8. Sophus

- 克隆仓库

```
cd 3rdparty # if current dir is 3rdparty, skip this command
git clone https://github.com/strasdat/Sophus
```

- CMake & Install

```
cd Sophus # if current dir is Sophus, skip this command
mkdir build && cd build
cmake ..
make -j4
sudo make install
```

9. g2o

- 系统依赖

```
cd 3rdparty # if current dir is 3rdparty, skip this command
sudo apt install libqt4-dev qt5-qmake libqglviewer-dev-qt5 libsuitesparse-dev libcxsparse3
    libcholmod3
```

- 克隆仓库

```
git clone https://github.com/RainerKuemmerle/g2o
cd g2o
git checkout 9b41a4ea
```

- CMake & Install

```
cd g2o # if current dir is g2o, skip this command
mkdir build && cd build
cmake ..
make -j4
sudo make install
```

10. DBow3

- 克隆仓库

```
cd 3rdparty # if current dir is 3rdparty, skip this command
git clone https://github.com/rmsalinas/DBow3
cd DBow3
git checkout c5ae539a
```

- CMake & Install

```
cd DBow3 # if current dir is DBow3, skip this command
mkdir build && cd build
cmake ..
make -j4
sudo make install
```

11. OpenCV 3

- 系统依赖

```
sudo apt install unzip build-essential libgtk2.0-dev libvtk6-dev libjpeg-dev libtiff5-dev
    libjasper-dev libopenexr-dev libtbb-dev
```

- 克隆仓库

```
cd 3rdparty # if current dir is 3rdparty, skip this command
wget https://github.com/opencv/opencv/archive/3.4.16.zip -O opencv3.zip
unzip -q opencv3.zip
```

- CMake & Install

```
cd opencv-3.4.16
mkdir build && cd build
cmake ..
make -j4
sudo make install
```

1.5.2 CMake 基本介绍

理论上，任意一个 C++ 程序都可以用 g++ 来编译。但是当程序规模越来越大时，一个工程可能有许多个文件夹和源文件，这时输入的编译命令将越来越长。通常，一个小型 C++ 项目可能还有十几个类，各类之间还存在复杂的依赖关系。其中一部分要编译成可执行文件，另一部分编译成库文件。如果仅依靠 g++ 命令，则需要输入大量的编译指令，整个编译过程会变得异常繁琐。因此，对于 C++ 项目，使用一些工程管理工具会更加高效，我们采用 CMake 管理源代码。

在一个 CMake 工程中，我们会用 cmake 命令生成一个 makefile 文件，然后，用 make 命令根据这个 makefile 文件内容编译整个工程。CMake 主要通过 CMakeLists.txt 文件来进行管理。以下用一个示例文件来说明 CMakeLists 的语法。

```
#声明要求的cmake最低版本
cmake_minimum_required(VERSION 3.2.0)

# 指定cmake编译策略
if(POLICY CMP0048)
    cmake_policy(SET CMP0048 NEW)
endif(POLICY CMP0048)

# 声明一个cmake工程，包含版本和语言类型
project(HelloSLAM VERSION 1.0.3 LANGUAGES CXX)

# gcc 编译添加选项
# add_definitions(-std=c++11) # 指定c++格式
add_definitions(-g) # 添加其他编译选项 -g 代表debug

# 头文件目录
include_directories(${PROJECT_SOURCE_DIR}/include)

# 寻找包
find_package(OPENCV 3 REQUIRED)

# 设置输出文件目录
# 静态库输出
set(CMAKE_ARCHIVE_OUTPUT_DIRECTORY ${CMAKE_BINARY_DIR}/Build/Lib)
# 动态库输出
set(CMAKE_LIBRARY_OUTPUT_DIRECTORY ${CMAKE_BINARY_DIR}/Build/Lib)
```

```
# 可执行二进制文件输出
set(CMAKE_RUNTIME_OUTPUT_DIRECTORY ${CMAKE_BINARY_DIR}/Build)

# 需要额外的链接库文件
# link_directories(${PROJECT_SOURCE_DIR})

# 编译代码，SHARED代表动态库，STATIC代表静态库
# 编译动态库
add_library(hello_shared SHARED xxxxx) # xxxxx 为动态库源代码路径
# 编译静态库
# add_library(hello_static STATIC xxxxx) # xxxxx 为静态库源代码路径
# 库依赖连接
# add_library(hello_shared XXX)
# add_library(hello_static XXX)

# 生成可执行文件
add_executable(helloSLAM xxxxx) # xxxxx 为可执行文件源代码路径
target_link_libraries(helloSLAM hello_shared) # 链接动态库
#target_link_libraries(hello_static hello_shared) # 链接静态库
```

第 2 章　刚体的三维运动

2.1　向量及其运算的矩阵表示

2.1.1　向量的描述及向量运算的矩阵表示

对于任意一个**向量**$\boldsymbol{u}$，都可以表示为某个**向量基** $\underline{\boldsymbol{e}} = \begin{bmatrix}\boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k}\end{bmatrix}^{\mathrm{T}}$ 的**基向量**的线性组合，即

$$\boldsymbol{u} = u_s\boldsymbol{i} + u_y\boldsymbol{j} + u_z\boldsymbol{k} \tag{2.1}$$

其中，$\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ 分别称为向量 $\boldsymbol{u}$ 在基向量上的 3 个分向量，3 个标量系数 u_x, u_y, u_z 分别称为向量 $\boldsymbol{u}$ 在 3 个基向量上的坐标。这 3 个坐标构成一个标量矩阵，称为向量 $\boldsymbol{u}$ 在基向量 $\boldsymbol{e}$ 上的**坐标阵**（或**分量矩阵**），记为

$$\boldsymbol{u} = \begin{bmatrix}u_x & u_y & u_z\end{bmatrix}^{\mathrm{T}} \tag{2.2}$$

那么向量 $\boldsymbol{u}$ 可以写成矩阵乘积的形式为

$$\boldsymbol{u} = \boldsymbol{u}^{\mathrm{T}}\boldsymbol{e} = \boldsymbol{e}^{\mathrm{T}}\boldsymbol{u} \tag{2.3}$$

由基向量的正交性，可以将 $\boldsymbol{u}$ 的分量矩阵写为

$$\boldsymbol{u} = \begin{bmatrix}\boldsymbol{u}\cdot\boldsymbol{i} \\ \boldsymbol{u}\cdot\boldsymbol{j} \\ \boldsymbol{u}\cdot\boldsymbol{k}\end{bmatrix} = \boldsymbol{u}\cdot\boldsymbol{e} = \boldsymbol{e}\cdot\boldsymbol{u} \tag{2.4}$$

2.1.2　向量叉乘的矩阵表示

已知向量在三维坐标系下的表达为 $\boldsymbol{u} = u_1\boldsymbol{i} + u_2\boldsymbol{j} + u_3\boldsymbol{k}$, $\boldsymbol{v} = v_1\boldsymbol{i} + v_2\boldsymbol{j} + v_3\boldsymbol{k}$，其中，$\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ 是单位标准正交基，且其运算满足表 2.1 ，即满足右手系。

×	$\boldsymbol{i}$	$\boldsymbol{j}$	$\boldsymbol{k}$
$\boldsymbol{i}$	0	$\boldsymbol{k}$	$-\boldsymbol{j}$
$\boldsymbol{j}$	$-\boldsymbol{k}$	0	$\boldsymbol{i}$
$\boldsymbol{k}$	$\boldsymbol{j}$	$-\boldsymbol{i}$	0

表 2.1: 单位标准正交向量的叉乘计算表（左 × 上）

那么，由向量叉乘的分配律及单位标准正交基的运算法则，

$$\begin{aligned}\boldsymbol{u}\times\boldsymbol{v} &= (u_1\boldsymbol{i}+u_2\boldsymbol{j}+u_3\boldsymbol{k})\times(v_1\boldsymbol{i}+v_2\boldsymbol{j}+v_3\boldsymbol{k})\\ &= u_1v_1(\boldsymbol{i}\times\boldsymbol{i})+u_1v_2(\boldsymbol{i}\times\boldsymbol{j})+u_1v_3(\boldsymbol{i}\times\boldsymbol{k})+u_2v_1(\boldsymbol{j}\times\boldsymbol{i})+u_2v_2(\boldsymbol{j}\times\boldsymbol{j})+u_2v_3(\boldsymbol{j}\times\boldsymbol{k})\\ &+u_3v_1(\boldsymbol{k}\times\boldsymbol{i})+u_3v_2(\boldsymbol{k}\times\boldsymbol{j})+u_3v_3(\boldsymbol{k}\times\boldsymbol{k})\\ &= u_1v_2\boldsymbol{k}+u_1v_3(-\boldsymbol{j})+u_2v_1(-\boldsymbol{k})+u_2v_3\boldsymbol{i}+u_3v_1\boldsymbol{j}+u_3v_2(-\boldsymbol{i})\\ &= (u_2v_3-u_3v_2)\boldsymbol{i}+(u_3v_1-u_1v_3)\boldsymbol{j}+(u_1v_2-u_2v_1)\boldsymbol{k}=\begin{vmatrix}\boldsymbol{i}&\boldsymbol{j}&\boldsymbol{k}\\u_1&u_2&u_3\\v_1&v_2&v_3\end{vmatrix}\end{aligned}$$

将向量用矩阵表达，即

$$\begin{aligned}\boldsymbol{u}\times\boldsymbol{v} &= (u_2v_3-u_3v_2)\boldsymbol{i}+(u_3v_1-u_1v_3)\boldsymbol{j}+(u_1v_2-u_2v_1)\boldsymbol{k}\\ &= \begin{bmatrix}\boldsymbol{i}&\boldsymbol{j}&\boldsymbol{k}\end{bmatrix}\begin{bmatrix}u_2v_3-u_3v_2\\u_3v_1-u_1v_3\\u_1v_2-u_2v_1\end{bmatrix}=\begin{bmatrix}\boldsymbol{i}&\boldsymbol{j}&\boldsymbol{k}\end{bmatrix}\begin{bmatrix}0&-u_3&u_2\\u_3&0&-u_1\\-u_2&u_1&0\end{bmatrix}\begin{bmatrix}v_1\\v_2\\v_3\end{bmatrix}\end{aligned}$$

所以，我们可以得到**向量叉乘矩阵**的定义

定义 2.1 向量叉乘矩阵

已知向量 $\boldsymbol{u}=u_1\boldsymbol{i}+u_2\boldsymbol{j}+u_3\boldsymbol{k}$，$\boldsymbol{v}=v_1\boldsymbol{i}+v_2\boldsymbol{j}+v_3\boldsymbol{k}$，其在同一个坐标系下的叉乘运算可以用矩阵表达

$$\boldsymbol{u}\times\boldsymbol{v}=\begin{bmatrix}\boldsymbol{i}&\boldsymbol{j}&\boldsymbol{k}\end{bmatrix}\begin{bmatrix}0&-u_3&u_2\\u_3&0&-u_1\\-u_2&u_1&0\end{bmatrix}\begin{bmatrix}v_1\\v_2\\v_3\end{bmatrix}=\boldsymbol{e}^{\mathrm{T}}\boldsymbol{u}^{\times}\boldsymbol{v}=\boldsymbol{v}^{\mathrm{T}}(\boldsymbol{u}^{\times})^{\mathrm{T}}\boldsymbol{e} \tag{2.5}$$

同时，叉乘矩阵有一个很重要的性质，从其定义很容易可以发现

$$(\boldsymbol{u}^{\times})^{\mathrm{T}}=\begin{bmatrix}0&u_3&-u_2\\-u_3&0&u_1\\u_2&-u_1&0\end{bmatrix}=-\begin{bmatrix}0&-u_3&u_2\\u_3&0&-u_1\\-u_2&u_1&0\end{bmatrix}=-\boldsymbol{u}^{\times} \tag{2.6}$$

所以向量叉乘矩阵又称为**反对称矩阵**。

进一步，我们将叉乘运算进一步写为分量形式后可以定义**广义向量叉乘运算**如下。

定义 2.2 广义向量叉乘运算

已知向量 $\boldsymbol{u},\boldsymbol{v}$ 在坐标系 S 下的分量形式为 $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{e}$，$\boldsymbol{e}^{\mathrm{T}}\boldsymbol{v}$，则

$$\boldsymbol{u}^{\mathrm{T}}\boldsymbol{e}\times\boldsymbol{e}^{\mathrm{T}}\boldsymbol{b}=\begin{bmatrix}\boldsymbol{i}&\boldsymbol{j}&\boldsymbol{k}\end{bmatrix}\begin{bmatrix}0&-u_3&u_2\\u_3&0&-u_1\\-u_2&u_1&0\end{bmatrix}\begin{bmatrix}v_1\\v_2\\v_3\end{bmatrix}=\boldsymbol{e}^{\mathrm{T}}\boldsymbol{u}^{\times}\boldsymbol{v} \tag{2.7}$$

定义广义向量叉乘运算为

$$\boldsymbol{u}^{\mathrm{T}}\boldsymbol{e}\times\boldsymbol{e}^{\mathrm{T}}=\boldsymbol{e}^{\mathrm{T}}\boldsymbol{u}^{\times} \tag{2.8}$$

2.1.3 叉乘矩阵的坐标变换

对于向量叉乘运算 $\boldsymbol{w} = \boldsymbol{u} \times \boldsymbol{v}$，在坐标系 S_a, S_b 下的矩阵形式分别为

$$\boldsymbol{w}_a = \boldsymbol{u}_a^\times \boldsymbol{v}_a, \qquad \boldsymbol{w}_b = \boldsymbol{u}_b^\times \boldsymbol{v}_b$$

由 $\boldsymbol{w}_a = \boldsymbol{C}_{ab}\boldsymbol{w}_b$，

$$\boldsymbol{C}_{ab}\boldsymbol{w}_b = \boldsymbol{a}^\times \boldsymbol{C}_{ab}\boldsymbol{v}_b \quad \Rightarrow \quad \boldsymbol{w}_b = \boldsymbol{C}_{ba}\boldsymbol{u}_a^\times \boldsymbol{C}_{ab}\boldsymbol{v}_b$$

通过对比前后两个式子可得

定理 2.1 叉乘矩阵的坐标变换

$\boldsymbol{u}$ 在坐标系 S_a, S_b 下的叉乘矩阵的坐标变换关系为

$$\boldsymbol{u}_b^\times = (\boldsymbol{C}_{ba}\boldsymbol{u}_a) = \boldsymbol{C}_{ba}\boldsymbol{u}_a^\times \boldsymbol{C}_{ab} = \boldsymbol{C}_{ba}\boldsymbol{u}_a^\times \boldsymbol{C}_{ba}^{\mathrm{T}} \tag{2.9}$$

进一步，可以扩展得到**叉乘矩阵恒等式**

$$(\boldsymbol{C}\boldsymbol{u})^\times = \boldsymbol{C}\boldsymbol{u}^\times \boldsymbol{C}^{\mathrm{T}} \tag{2.10}$$

2.1.4 向量两边乘同一个向量运算的矩阵表示

已知向量在三维坐标系下的表达为 $\boldsymbol{u} = u_1\boldsymbol{i} + u_2\boldsymbol{j} + u_3\boldsymbol{k}$, $\boldsymbol{v} = v_1\boldsymbol{i} + v_2\boldsymbol{j} + v_3\boldsymbol{k}$，则

$$\begin{aligned}
\boldsymbol{u} \cdot \boldsymbol{v} \cdot \boldsymbol{u} &= (u_1\boldsymbol{i} + u_2\boldsymbol{j} + u_3\boldsymbol{k}) \cdot (v_1\boldsymbol{i} + v_2\boldsymbol{j} + v_3\boldsymbol{k})\boldsymbol{u} \\
&= (u_1v_1 + u_2v_2 + u_3v_3)\boldsymbol{u} \\
&= (u_1v_1 + u_2v_2 + u_3v_3)(u_1\boldsymbol{i} + u_2\boldsymbol{j} + u_3\boldsymbol{k}) \\
&= (u_1^2v_1 + u_1u_2v_2 + u_1u_3v_3)\boldsymbol{i} + (u_1u_2v_1 + u_2^2v_2 + u_2u_3v_3)\boldsymbol{j} + (u_1u_3v_1 + u_2u_3v_2 + u_3^2v_3)\boldsymbol{k}
\end{aligned}$$

用矩阵表达为

$$\begin{aligned}
\boldsymbol{u} \cdot \boldsymbol{v} \cdot \boldsymbol{u} &= (u_1^2v_1 + u_1u_2v_2 + u_1u_3v_3)\boldsymbol{i} + (u_1u_2v_1 + u_2^2v_2 + u_2u_3v_3)\boldsymbol{j} + (u_1u_3v_1 + u_2u_3v_2 + u_3^2v_3)\boldsymbol{k} \\
&= \begin{bmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \end{bmatrix} \begin{bmatrix} u_1^2v_1 + u_1u_2v_2 + u_1u_3v_3 \\ u_1u_2v_1 + u_2^2v_2 + u_2u_3v_3 \\ u_1u_3v_1 + u_2u_3v_2 + u_3^2v_3 \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \end{bmatrix} \begin{bmatrix} u_1^2 & u_1u_2 & u_1u_3 \\ u_1u_2 & u_2^2 & u_2u_3 \\ u_1u_3 & u_2u_3 & u_3^2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}
\end{aligned}$$

所以，我们可以得到

定理 2.2 向量两边乘同一个向量运算的矩阵表示

已知向量 $\boldsymbol{u} = u_1\boldsymbol{i} + u_2\boldsymbol{j} + u_3\boldsymbol{k}$, $\boldsymbol{v} = v_1\boldsymbol{i} + v_2\boldsymbol{j} + v_3\boldsymbol{k}$，则

$$\boldsymbol{u}\cdot\boldsymbol{v}\cdot\boldsymbol{u} = \begin{bmatrix}\boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k}\end{bmatrix}\begin{bmatrix}u_1\\u_2\\u_3\end{bmatrix}\begin{bmatrix}u_1 & u_2 & u_3\end{bmatrix}\begin{bmatrix}v_1\\v_2\\v_3\end{bmatrix} = \boldsymbol{e}^{\mathrm{T}}\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}}\boldsymbol{v} = \boldsymbol{v}^{\mathrm{T}}\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}}\boldsymbol{e} \tag{2.11}$$

2.2 旋转矩阵

对于坐标系原点重合的两个不同的坐标系 S_a 和 S_b，坐标基分别为 $\boldsymbol{e}_a$ 和 $\boldsymbol{e}_b$，对于矢量 $\boldsymbol{u}$ 在两个坐标系下的分解，有

$$\boldsymbol{u} = \boldsymbol{e}_b u_b = \boldsymbol{e}_a u_a \tag{2.12}$$

两边同时乘以 $\boldsymbol{e}_b$，得

$$\boldsymbol{e}_b\boldsymbol{e}_b u_b = \boldsymbol{e}_b\boldsymbol{e}_a u_a \quad \Rightarrow \quad u_b = \boldsymbol{e}_b\boldsymbol{e}_a u_a$$

为此我们定义坐标系 S_a 变换为坐标系 S_b 的**方向余弦矩阵**（**坐标系旋转矩阵**）为

$$\boldsymbol{C}_{ba} = \boldsymbol{e}_b\boldsymbol{e}_a = \begin{bmatrix}\boldsymbol{i}_b\cdot\boldsymbol{e}_a\\\boldsymbol{j}_b\cdot\boldsymbol{e}_a\\\boldsymbol{k}_b\cdot\boldsymbol{e}_a\end{bmatrix} = \begin{bmatrix}\boldsymbol{i}_b\cdot\boldsymbol{i}_a & \boldsymbol{i}_b\cdot\boldsymbol{j}_a & \boldsymbol{i}_b\cdot\boldsymbol{k}_a\\\boldsymbol{j}_b\cdot\boldsymbol{i}_a & \boldsymbol{j}_b\cdot\boldsymbol{j}_a & \boldsymbol{j}_b\cdot\boldsymbol{k}_a\\\boldsymbol{k}_b\cdot\boldsymbol{i}_a & \boldsymbol{k}_b\cdot\boldsymbol{j}_a & \boldsymbol{k}_b\cdot\boldsymbol{k}_a\end{bmatrix} = \begin{bmatrix}C_{11} & C_{12} & C_{13}\\C_{21} & C_{22} & C_{23}\\C_{31} & C_{32} & C_{33}\end{bmatrix} \tag{2.13}$$

方向余弦矩阵有以下几个特征：

1. [6 个约束方程]

(1) 模值约束

$$\begin{cases}\left|\boldsymbol{i}_b\right|^2 = C_{11}^2 + C_{12}^2 + C_{13}^2 = 1\\\left|\boldsymbol{j}_b\right|^2 = C_{21}^2 + C_{22}^2 + C_{23}^2 = 1\\\left|\boldsymbol{k}_b\right|^2 = C_{31}^2 + C_{32}^2 + C_{33}^2 = 1\end{cases} \tag{2.14}$$

证 由于 $\boldsymbol{i}_b, \boldsymbol{j}_b, \boldsymbol{k}_b$ 的模值为 1（空间绝对），所以将它们投影到坐标系 S_a 后模值仍然为 1，即

$$\begin{cases}\left|\boldsymbol{i}_b\cdot\boldsymbol{e}_a\right|^2 = \left|\boldsymbol{i}_b\right|^2 = 1\\\left|\boldsymbol{j}_b\cdot\boldsymbol{e}_a\right|^2 = \left|\boldsymbol{j}_b\right|^2 = 1\\\left|\boldsymbol{k}_b\cdot\boldsymbol{e}_a\right|^2 = \left|\boldsymbol{k}_b\right|^2 = 1\end{cases} \quad\Longrightarrow\quad \begin{cases}\left|\boldsymbol{i}_b\right|^2 = C_{11}^2 + C_{12}^2 + C_{13}^2 = 1\\\left|\boldsymbol{j}_b\right|^2 = C_{21}^2 + C_{22}^2 + C_{23}^2 = 1\\\left|\boldsymbol{k}_b\right|^2 = C_{31}^2 + C_{32}^2 + C_{33}^2 = 1\end{cases}$$

(2) 几何约束

$$\begin{cases}\boldsymbol{i}_b\cdot\boldsymbol{j}_b = C_{11}C_{21} + C_{12}C_{22} + C_{13}C_{23} = 0\\\boldsymbol{i}_b\cdot\boldsymbol{k}_b = C_{11}C_{31} + C_{12}C_{32} + C_{13}C_{33} = 0\\\boldsymbol{j}_b\cdot\boldsymbol{k}_b = C_{21}C_{31} + C_{22}C_{32} + C_{23}C_{33} = 0\end{cases} \tag{2.15}$$

证 由于 $\boldsymbol{i}_b, \boldsymbol{j}_b, \boldsymbol{k}_b$ 两两正交（空间绝对），所以将它们投影到坐标系 S_a 后仍然满足几何关系，即

$$\begin{cases} \boldsymbol{i}_b \cdot \boldsymbol{j}_b = (\boldsymbol{i}_b \cdot \boldsymbol{e}_a) \cdot (\boldsymbol{j}_b \cdot \boldsymbol{e}_a) = 0 \\ \boldsymbol{i}_b \cdot \boldsymbol{k}_b = (\boldsymbol{i}_b \cdot \boldsymbol{e}_a) \cdot (\boldsymbol{k}_b \cdot \boldsymbol{e}_a) = 0 \\ \boldsymbol{j}_b \cdot \boldsymbol{k}_b = (\boldsymbol{j}_b \cdot \boldsymbol{e}_a) \cdot (\boldsymbol{k}_b \cdot \boldsymbol{e}_a) = 0 \end{cases} \Longrightarrow \begin{cases} \boldsymbol{i}_b \cdot \boldsymbol{j}_b = C_{11}C_{21} + C_{12}C_{22} + C_{13}C_{23} = 0 \\ \boldsymbol{i}_b \cdot \boldsymbol{k}_b = C_{11}C_{31} + C_{12}C_{32} + C_{13}C_{33} = 0 \\ \boldsymbol{j}_b \cdot \boldsymbol{k}_b = C_{21}C_{31} + C_{22}C_{32} + C_{23}C_{33} = 0 \end{cases}$$

2. [坐标变换矩阵是正交矩阵]

由于

$$\begin{cases} \boldsymbol{e}_b \cdot \boldsymbol{e}_b^T = \boldsymbol{e}_a \cdot \boldsymbol{e}_a^{\mathrm{T}} = \boldsymbol{E}_3 \\ \boldsymbol{e}_b \cdot \boldsymbol{e}_b^T = \boldsymbol{C}_{ba}\boldsymbol{e}_a \cdot (\boldsymbol{C}_{ba}\boldsymbol{e}_a)^{\mathrm{T}} = \boldsymbol{C}_{ba}\boldsymbol{e}_a\boldsymbol{e}_a^{\mathrm{T}}\boldsymbol{C}_{ba}^{\mathrm{T}} \end{cases} \Rightarrow \quad \boldsymbol{E}_3 = \boldsymbol{C}_{ba}(\boldsymbol{e}_a\boldsymbol{e}_a^{\mathrm{T}})\boldsymbol{C}_{ba}^{\mathrm{T}} = \boldsymbol{C}_{ba}\boldsymbol{C}_{ba}^{\mathrm{T}}$$

所以可以得到

$$\boldsymbol{C}_{ba}^{-1} = \boldsymbol{C}_{ba}^{\mathrm{T}} \tag{2.16}$$

且有

$$\boldsymbol{C}_{ab} = \boldsymbol{e}_a\boldsymbol{e}_b^{\mathrm{T}} = \boldsymbol{e}_a\boldsymbol{e}_a^{\mathrm{T}}\boldsymbol{C}_{ba}^{\mathrm{T}} = \boldsymbol{C}_{ba}^{-1} \tag{2.17}$$

3. [坐标变换矩阵的行列式为 1]

由于矩阵乘积的行列式等于行列式的乘积且矩阵的转置的行列式等于矩阵的行列式，所以

$$\det(\boldsymbol{C}_{ba})\det(\boldsymbol{C}_{ba}) = \left[\det(\boldsymbol{C}_{ba})\right]^2 = \det(\boldsymbol{C}_{ba}\boldsymbol{C}_{ba}) = 1 \quad \Rightarrow \quad \det(\boldsymbol{C}_{ba}) = \pm 1 \tag{2.18}$$

而因为

$$\boldsymbol{C}_{ba} = \mathrm{adj}(\boldsymbol{C}_{ba}) \quad \Rightarrow \quad \boldsymbol{C}_{ba}^{-1} = \frac{\mathrm{adj}(\boldsymbol{C}_{ba})}{\det(\boldsymbol{C}_{ba})} = \frac{\boldsymbol{C}_{ba}}{\det(\boldsymbol{C}_{ba})} = \frac{\boldsymbol{C}_{ba}^{-1}}{\det(\boldsymbol{C}_{ba})}$$

因此

$$\det(\boldsymbol{C}_{ba}) = +1 \tag{2.19}$$

4. [相继运动的坐标变换矩阵]

对于坐标系原点重合的三个不同的坐标系 S_a，S_b 和 S_c，有

$$\begin{cases} \boldsymbol{e}_b \cdot \boldsymbol{e}_a = \boldsymbol{C}_{ba} \\ \boldsymbol{e}_c \cdot \boldsymbol{e}_b = \boldsymbol{C}_{cb} \\ \boldsymbol{e}_c \cdot \boldsymbol{e}_a = \boldsymbol{C}_{ca} \end{cases} \Rightarrow \quad \boldsymbol{e}_c = \boldsymbol{C}_{ca}\boldsymbol{e}_a = \boldsymbol{C}_{cb}\boldsymbol{e}_b = \boldsymbol{C}_{cb}\boldsymbol{C}_{ba}\boldsymbol{e}_a$$

因此

$$\boldsymbol{C}_{ca} = \boldsymbol{C}_{cb}\boldsymbol{C}_{ba} \tag{2.20}$$

2.3 欧拉角

定义 2.3 基元旋转矩阵

基元旋转矩阵 任何一个坐标变换可以看成是绕三个基本轴的旋转，这三个基本轴的坐标转换矩阵为基元旋转矩阵，如图 2.1, 2.2, 2.3 所示，绕各个轴旋转的角度称为**欧拉角**。每个坐标轴对应的基元旋转矩阵为

$$\boldsymbol{C}_x(\varphi)=\begin{bmatrix}1&0&0\\0&\cos\varphi&\sin\varphi\\0&-\sin\varphi&\cos\varphi\end{bmatrix}\quad \boldsymbol{C}_y(\theta)=\begin{bmatrix}\cos\theta&0&-\sin\theta\\0&1&0\\\sin\theta&0&\sin\theta\end{bmatrix}\quad \boldsymbol{C}_z(\psi)=\begin{bmatrix}\cos\psi&\sin\psi&0\\-\sin\psi&\cos\psi&0\\0&0&1\end{bmatrix}\tag{2.21}$$

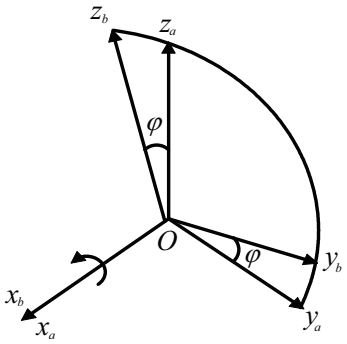

图 2.1: 绕 x 轴旋转

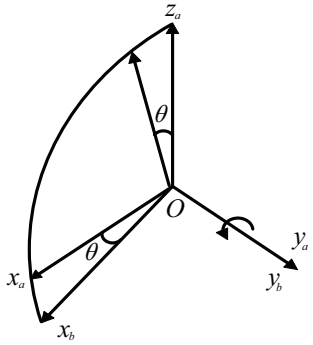

图 2.2: 绕 y 轴旋转

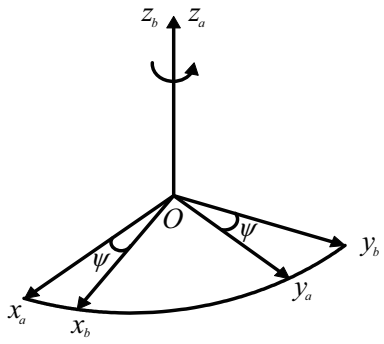

图 2.3: 绕 z 轴旋转

下面给出两种基元旋转矩阵表示的坐标变换。

1. [ZXZ 旋转顺序]

如图 2.4 所示，方向余弦矩阵和 ZXZ 顺序欧拉角的关系为

$$\boldsymbol{C}_{ba}=\boldsymbol{C}_z(\varphi)\boldsymbol{C}_x(\theta)\boldsymbol{C}_z(\psi)=\begin{bmatrix}\cos\varphi\cos\psi-\sin\varphi\cos\theta\sin\psi&\cos\varpi\sin\psi+\sin\varphi\cos\theta\cos\psi&\sin\varphi\sin\theta\\-\sin\varphi\cos\psi-\cos\varphi\cos\theta\sin\psi&-\sin\varphi\sin\psi+\cos\varphi\cos\theta\cos\psi&\cos\varphi\sin\theta\\\sin\theta\sin\psi&-\sin\theta\cos\psi&\cos\theta\end{bmatrix}\tag{2.22}$$

通过与方向余弦矩阵的对应项进行对比，可以计算得到

$$\begin{cases}\psi=-\tan^{-1}\left(\dfrac{C_{31}}{C_{32}}\right)\\\theta=\cos^{-1}\left(C_{33}\right)\\\varphi=\tan^{-1}\left(\dfrac{C_{13}}{C_{23}}\right)\end{cases}\tag{2.23}$$

由公式 (2.23) 可知，若欧拉角 $\theta=0^\circ$，则欧拉转动处于奇异状态，欧拉角 ψ,φ 不能唯一确定。因此，θ 的取值范围为 $0^\circ<\theta<180^\circ$。

2. [ZXY 旋转顺序]

如图 2.5 所示，方向余弦矩阵和 ZXY 顺序欧拉角的关系

$$\boldsymbol{C}_{ba}=\boldsymbol{C}_y(\theta)\boldsymbol{C}_x(\varphi)\boldsymbol{C}_z(\psi)=\begin{bmatrix}\cos\theta\cos\psi-\sin\varphi\sin\theta\sin\psi&\cos\theta\sin\psi+\sin\varphi\sin\theta\cos\psi&-\cos\varphi\sin\theta\\-\cos\varphi\sin\psi&\cos\varphi\cos\psi&\sin\varphi\\\sin\theta\cos\varphi+\sin\varphi\cos\theta\sin\psi&\sin\theta\sin\psi-\sin\varphi\cos\theta\cos\psi&\cos\varphi\cos\theta\end{bmatrix}\tag{2.24}$$

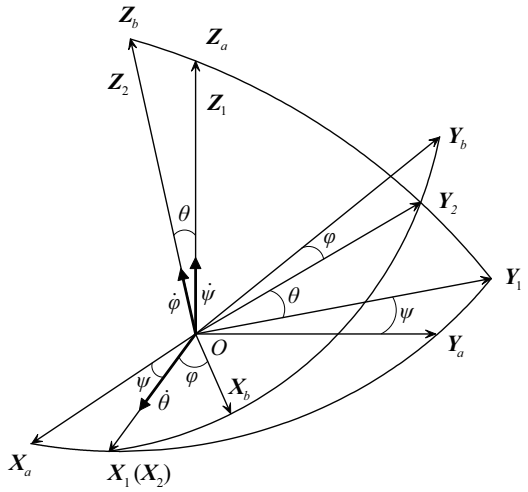

图 2.4: ZXZ 顺序欧拉角旋转

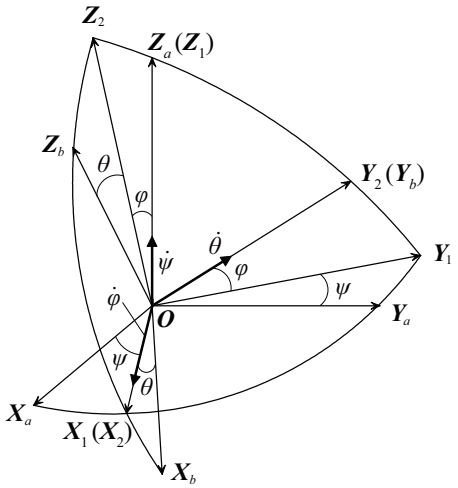

图 2.5: ZXY 顺序欧拉角旋转

通过与方向余弦矩阵的对应项进行对比，可以计算得到

$$\begin{cases} \psi = -\tan^{-1}\left(\dfrac{C_{21}}{C_{22}}\right) \\ \theta = \sin^{-1}\left(C_{23}\right) \\ \varphi = \tan^{-1}\left(\dfrac{C_{13}}{C_{33}}\right) \end{cases} \tag{2.25}$$

由公式 (2.25) 可知，若欧拉角 $\theta = \pm 90°$，则欧拉转动处于奇异状态，欧拉角 ψ, φ 在同一平面转动，不能唯一确定。

2.4 欧拉轴角

定义 2.4 欧拉轴 / 角

坐标系 S_b 相对坐标系 S_a 的姿态参数可以用单位矢量 $\boldsymbol{e}$ 在参考坐标系 S_a 的三个分量 e_x, e_y, e_z 以及绕此转轴的转角 Φ 这 4 个参数来描述，称为**欧拉轴 / 角**参数。矢量 $\boldsymbol{e}$ 称为**欧拉轴**，Φ 称为**欧拉转角**。

2.4.1 欧拉轴 / 角与方向余项矩阵的相互转换

1. [欧拉轴 / 角求方向余弦矩阵]

方向余项矩阵 $\boldsymbol{C}_{ba}$ 可由欧拉轴 / 角参数 $\boldsymbol{e}, \Phi$ 得到。

如图 2.6 所示，设矢量 $\boldsymbol{a}$ 是固定在坐标系 S_a 的任意矢量，矢量 $\boldsymbol{b}$ 是固定于坐标系 S_b 中的矢量。将矢量 $\boldsymbol{a}$ 绕轴 $\boldsymbol{e}$ 旋转一个角度 Φ 后得到矢量 $\boldsymbol{b}$。首先将矢量 $\boldsymbol{a}$，矢量 $\boldsymbol{b}$ 沿轴 $\boldsymbol{e}$ 方向和垂直于轴 $\boldsymbol{e}$ 方向分解，其平行分量 $\boldsymbol{a}_{\parallel} = \boldsymbol{b}_{\parallel} = \boldsymbol{e}$ 相等，即旋转前后不变，可以发现旋转仅与垂直分量 $\boldsymbol{a}_{\perp}, \boldsymbol{b}_{\perp}$ 有关。

将垂直分量 $\boldsymbol{a}_{\perp}, \boldsymbol{b}_{\perp}$ 投影到垂直于轴 $\boldsymbol{e}$ 的平面 Π 上，如图 2.7 所示。定义单位正交矢量 $\boldsymbol{u}, \boldsymbol{v}$

$$\boldsymbol{v} = \boldsymbol{a}_{\perp} = \frac{\boldsymbol{e} \times \boldsymbol{a}}{|\boldsymbol{e} \times \boldsymbol{a}|} = \frac{1}{a \sin\theta}(\boldsymbol{e} \times \boldsymbol{a})$$

$$\boldsymbol{u} = \boldsymbol{v} \times \boldsymbol{e} = \frac{1}{a \sin\theta}(\boldsymbol{e} \times \boldsymbol{a}) \times \boldsymbol{e} = \frac{1}{a \sin\theta}\left[\boldsymbol{a} - (\boldsymbol{e} \cdot \boldsymbol{a})\boldsymbol{e}\right]$$

将矢量 $\boldsymbol{b}_{\perp}$ 分解，得

$$\boldsymbol{b}_{\perp} = \cos\Phi\boldsymbol{u} + \sin\Phi\boldsymbol{v} = \cos\Phi(\boldsymbol{a}_{\perp} \times \boldsymbol{e}) + \sin\Phi\boldsymbol{a}_{\perp}$$

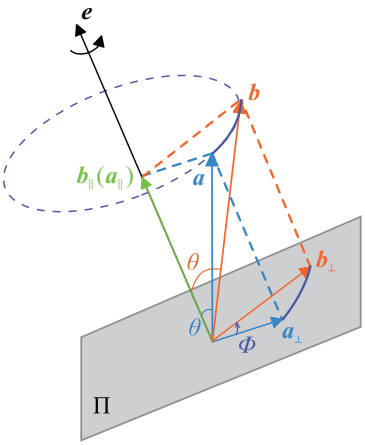

图 2.6: 欧拉轴旋转分解图

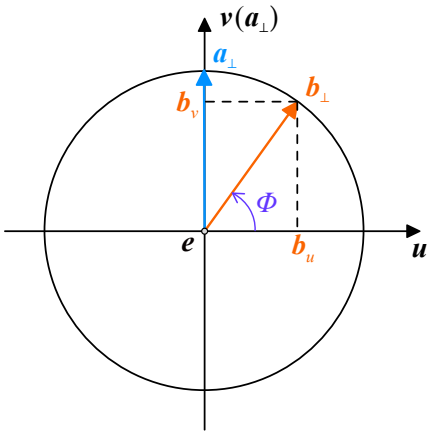

图 2.7: 欧拉轴旋转投影图

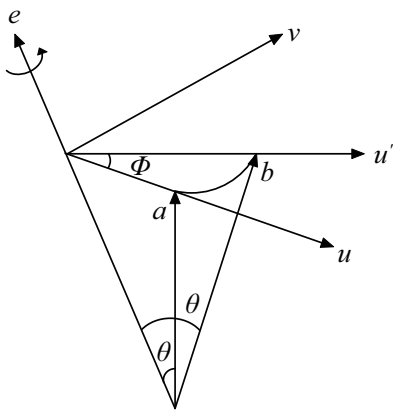

图 2.8: 欧拉轴 / 角的坐标变换图

将矢量 $\boldsymbol{a}, \boldsymbol{b}$ 用上面的矢量表示为（注：$\boldsymbol{a}, \boldsymbol{b}$ 模长相等，即 $a = b$）

$$\boldsymbol{a} = a\left(\cos\theta \boldsymbol{a}_{\parallel} + \sin\theta \boldsymbol{a}_{\perp}\right) = a\left(\cos\theta \boldsymbol{e} + \sin\theta \boldsymbol{v}\right) \tag{2.26}$$

$$\boldsymbol{b} = b\left(\cos\theta \boldsymbol{b}_{\parallel} + \sin\theta \boldsymbol{b}_{\perp}\right) = a\left(\cos\theta \boldsymbol{e} + \sin\theta \boldsymbol{b}_{\perp}\right) \tag{2.27}$$

将 $\boldsymbol{b}_{\perp}$ 的表达式反代，可以得到

$$\boldsymbol{b} = \cos\Phi \boldsymbol{a} + (1 - \cos\Phi)(\boldsymbol{e} \cdot \boldsymbol{a})\boldsymbol{e} + \sin\Phi(\boldsymbol{e} \times \boldsymbol{a}) \tag{2.28}$$

为了写成矩阵形式，我们利用把向量（在坐标系 S_a 的分量）的计算转换为矩阵的形式[1]，即

$$\boldsymbol{e} \cdot \boldsymbol{a} \cdot \boldsymbol{e} = \begin{bmatrix} \boldsymbol{i}_a & \boldsymbol{j}_a & \boldsymbol{k}_a \end{bmatrix} \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} \begin{bmatrix} e_x & e_y & e_z \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \boldsymbol{e}_a^{\mathrm{T}} \boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} \boldsymbol{a}, \quad \boldsymbol{e} \times \boldsymbol{a} = \begin{bmatrix} \boldsymbol{i}_a & \boldsymbol{j}_a & \boldsymbol{k}_a \end{bmatrix} \begin{bmatrix} 0 & -e_z & e_y \\ e_z & 0 & -e_x \\ -e_y & e_x & 0 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \boldsymbol{e}_a^{\mathrm{T}} \boldsymbol{e}^{\times} \boldsymbol{a}$$

将 $\boldsymbol{a}, \boldsymbol{b}$ 在坐标系 S_a 下分解，得

$$\boldsymbol{e}_a^{\mathrm{T}} \boldsymbol{b} = \cos\Phi \boldsymbol{e}_a^{\mathrm{T}} \boldsymbol{a} + (1 - \cos\Phi) \boldsymbol{e}_a^{\mathrm{T}} \boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} \boldsymbol{a} + \sin\Phi \boldsymbol{e}_a^{\mathrm{T}} \boldsymbol{e}^{\times} \boldsymbol{a} \tag{2.29}$$

两边同时左乘 $\left[\boldsymbol{e}_a^{\mathrm{T}}\right]^{-1}$，消去 $\boldsymbol{e}_a^{\mathrm{T}}$，可以得到

$$\begin{aligned} \boldsymbol{b} &= \cos\Phi \boldsymbol{a} + (1 - \cos\Phi) \boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} \boldsymbol{a} + \sin\Phi \boldsymbol{e}_a^{\mathrm{T}} \boldsymbol{e}^{\times} \boldsymbol{a} \\ &= \left[\cos\Phi \boldsymbol{E}_3 + (1 - \cos\Phi) \boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} + \sin\Phi \boldsymbol{e}_a^{\mathrm{T}} \boldsymbol{e}^{\times}\right] \boldsymbol{a} \end{aligned} \tag{2.30}$$

可以得到

[1] 关于 $\boldsymbol{e}\boldsymbol{e}^{\mathrm{T}}$ 和叉乘矩阵 $\boldsymbol{e}^{\times}$ 的说明及向量运算矩阵表示的证明，详见前面的小节：2.1 向量运算的矩阵表示，Page 11

定理 2.3　欧拉轴 / 角参数下的向量旋转矩阵

在同一坐标系 S_a 下将向量 $\boldsymbol{a}$ 绕轴 $\boldsymbol{e}$ 旋转 Φ 角度得到向量 $\boldsymbol{b}$，它们的关系为

$$\boldsymbol{b} = \boldsymbol{R}_{ba}\boldsymbol{a} \tag{2.31}$$

其中，**向量旋转矩阵**定义为

$$\boldsymbol{R}_{ba} = \cos\Phi\boldsymbol{E}_3 + (1-\cos\Phi)\boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} + \sin\Phi\boldsymbol{e}^{\times} \tag{2.32}$$

$$= \begin{bmatrix} \cos\Phi + e_x^2(1-\cos\Phi) & e_xe_y(1-\cos\Phi) - e_z\sin\Phi & e_xe_z(1-\cos\Phi) + e_y\sin\Phi \\ e_xe_y(1-\cos\Phi) + e_z\sin\Phi & \cos\Phi + e_y^2(1-\cos\Phi) & e_ye_z(1-\cos\Phi) - e_x\sin\Phi \\ e_xe_z(1-\cos\Phi) - e_y\sin\Phi & e_ye_z(1-\cos\Phi) + e_x\sin\Phi & \cos\Phi + e_z^2(1-\cos\Phi) \end{bmatrix} \tag{2.33}$$

而由向量旋转与坐标系旋转的对应关系，可以知道向量旋转矩阵和坐标系旋转矩阵（方向余弦矩阵）互为转置（逆），所以可以得到[2]

定理 2.4　欧拉轴 / 角参数下的坐标系旋转矩阵（方向余弦矩阵）

欧拉轴 / 角参数下的坐标系旋转矩阵（方向余弦矩阵）为

$$\begin{aligned} \boldsymbol{C}_{ba} &= (\boldsymbol{R}_{ba})^{\mathrm{T}} = \left[\cos\Phi\boldsymbol{E}_3 + (1-\cos\Phi)\boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} + \sin\Phi\boldsymbol{e}^{\times}\right]^{\mathrm{T}} \\ &= \cos\Phi\boldsymbol{E}_3 + (1-\cos\Phi)(\boldsymbol{e}\boldsymbol{e}^{\mathrm{T}})^{\mathrm{T}} + \sin\Phi(\boldsymbol{e}^{\times})^{\mathrm{T}} \\ &= \cos\Phi\boldsymbol{E}_3 + (1-\cos\Phi)\boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} - \sin\Phi\boldsymbol{e}^{\times} \end{aligned} \tag{2.34}$$

$$= \begin{bmatrix} \cos\Phi + e_x^2(1-\cos\Phi) & e_xe_y(1-\cos\Phi) + e_z\sin\Phi & e_xe_z(1-\cos\Phi) - e_y\sin\Phi \\ e_xe_y(1-\cos\Phi) - e_z\sin\Phi & \cos\Phi + e_y^2(1-\cos\Phi) & e_ye_z(1-\cos\Phi) + e_x\sin\Phi \\ e_xe_z(1-\cos\Phi) + e_y\sin\Phi & e_ye_z(1-\cos\Phi) - e_x\sin\Phi & \cos\Phi + e_z^2(1-\cos\Phi) \end{bmatrix} \tag{2.35}$$

2. [由方向余弦矩阵确定欧拉轴 / 角参数]

若已知方向余弦矩阵 $\boldsymbol{C}_{ba}$，可以计算得到欧拉轴 / 角参数，得

$$\cos\Phi = \frac{\mathrm{tr}\boldsymbol{C}_{ba} - 1}{2} \tag{2.36}$$

$$\boldsymbol{e} = \frac{1}{2\sin\Phi}\begin{bmatrix} C_{23} - C_{32} \\ C_{31} - C_{13} \\ C_{12} - C_{21} \end{bmatrix} \tag{2.37}$$

其中，$\mathrm{tr}\boldsymbol{C}_{ba}$ 是方向余弦矩阵的迹，$\mathrm{tr}\boldsymbol{C}_{ba} = C_{11} + C_{22} + C_{33}$。绕任意轴转动相同的 Φ 角，方向余项矩阵的迹不变。将公式展开，对应得到以下 3 组解：

(1)　当 $\mathrm{tr}\boldsymbol{C}_{ba} \neq 3, -1$ 时，可以直接计算分量 e_x, e_y, e_z。

(2)　当 $\mathrm{tr}\boldsymbol{C}_{ba} = 3$ 时，此时对应转角 $\Phi = 0, \pm 2\pi, \pm 4\pi, \cdots$，方向余弦矩阵 $\boldsymbol{C}_{ba}$ 为单位矩阵，e_x, e_y, e_z 无法确定。这种情况相当于没有发生转动。

(3)　当 $\mathrm{tr}\boldsymbol{C}_{ba} = -1$ 时，对应转角 $\Phi = \pm\pi, \pm 3\pi, \pm 5\pi, \cdots$，此时 $\boldsymbol{C}_{ba} = 2\boldsymbol{e}\boldsymbol{e} - \boldsymbol{E}_3$，则有

$$\begin{cases} e_x = \pm\sqrt{\dfrac{1+C_{11}}{2}}, & e_y = \pm\sqrt{\dfrac{1+C_{22}}{2}}, & e_z = \pm\sqrt{\dfrac{1+C_{33}}{2}} \\ e_xe_y = \dfrac{1}{2}C_{12}, & e_ye_z = \dfrac{1}{2}C_{23}, & e_ze_x = \dfrac{1}{2}C_{31} \end{cases} \tag{2.38}$$

[2]证明详见第 3 章：3.3 向量旋转矩阵和坐标系旋转矩阵的关系，Page 45

其中，使用后三个方程来判断前三个方程的符号。

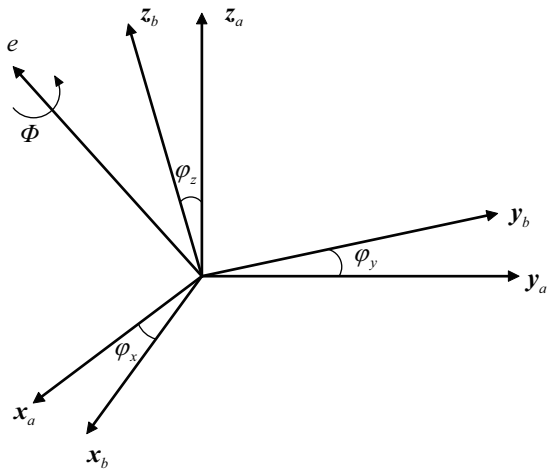

图 2.9: 欧拉转角与两个坐标系之间的几何关系

2.4.2 欧拉转角的几何意义

如图 2.9 所示，令 $\varphi_x, \varphi_y, \varphi_z$ 分别为两个坐标系对应轴之间的夹角，方向余弦矩阵的对焦线上的元素即为这几个角的余弦值，所以

$$2\cos\Phi = \mathrm{tr}\boldsymbol{C}_{ba} - 1 = \cos\varphi_x + \cos\varphi_y + \cos\varphi_z - 1 \tag{2.39}$$

利用半角公式 $\cos\alpha = 1 - 2\sin^2\frac{\alpha}{2}$，则

$$\sin^2\frac{\Phi}{2} = \frac{1}{2}\left(\sin^2\frac{\varphi_x}{2} + \sin^2\frac{\varphi_y}{2} + \sin^2\frac{\varphi_z}{2}\right) \tag{2.40}$$

当偏角较小时，有

$$\Phi \approx \frac{\sqrt{2}}{2}\sqrt{\varphi_x^2 + \varphi_y^2 + \varphi_z^2} \tag{2.41}$$

这个公式对于评价旋转误差是很有用的。

2.5 四元数

2.5.1 四元数的定义

定义 2.5 四元数

四元数的定义和复数类似，唯一的区别就是四元数一共有三个虚部，而复数只有一个。所有的四元数 $q \in \mathbb{H}$（$\mathbb{H}$ 代表四元数的发现者 William Rowan Hamilton）都可以写成下面这种形式

$$q = a + bi + cj + dk \quad (a, b, c, d \in \mathbb{R}) \tag{2.42}$$

其中，

$$i^2 = j^2 = k^2 = ijk = -1 \tag{2.43}$$

四元数可以写成向量形式

$$q = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \tag{2.44}$$

同时，通常将四元数的实部与虚部分开，并用一个三维的向量来表示虚部，将它表示为**标量向量有序对**形式

$$q = [s, \boldsymbol{v}], \quad \boldsymbol{v} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (s, x, y, z \in \mathbb{R}) \tag{2.45}$$

2.5.2 四元数的基本运算

1. [四元数的模长]

定义 **2.6** 四元数的模长

四元数的模长（范数）定义为

$$\|q\| = \sqrt{a^2 + b^2 + c^2 + d^2} \tag{2.46}$$

用标量向量有序对表示为

$$\|q\| = \sqrt{s^2 + \|\boldsymbol{v}\|^2} = \sqrt{s^2 + \boldsymbol{v} \cdot \boldsymbol{v}} \tag{2.47}$$

注：由于四元数是四维向量，其没有明确的物理几何意义。

2. [四元数的加减运算]

四元数的加减运算和复数一致，设两个四元数为 $q_1 = a + bi + cj + dk$, $q_2 = e + fi + gj + hk$，则

$$\begin{aligned} q_1 \pm q_2 &= a + bi + cj + dk \pm (e + fi + gj + hk) \\ &= (a \pm e) + (b \pm f)i + (c \pm g)j + (d \pm h)k \end{aligned} \tag{2.48}$$

标量向量对形式的加减与标量和向量的加减运算一致，设两个四元数为 $q_1 = [s, \boldsymbol{v}]$, $q_2 = [t, \boldsymbol{u}]$，则

$$q_1 \pm q_2 = [s \pm t, \boldsymbol{v} \pm \boldsymbol{u}] \tag{2.49}$$

3. [四元数的数乘]

一个四元数 $q = a + bi + cj + dk$ 和标量 s 的乘积为

$$\begin{aligned} sq &= s(a + bi + cj + dk) \\ &= sa + sbi + scj + sdk \end{aligned} \tag{2.50}$$

四元数的数乘运算满足交换律，即 $sq = qs$.

4. [四元数的乘法]

四元数的乘法和矩阵乘法类似，不遵守交换律，即在一般情况下 $q_1 q_2 \neq q_2 q_1$。所以，四元数乘法和矩阵乘法一样有左乘和右乘的区别。即

(1) $q_1 q_2$ q_1 **左乘** q_2 或 q_2 **右乘** q_1.

(2) $q_2 q_1$ q_1 **右乘** q_2 或 q_2 **左乘** q_1.

四元数的乘法满足结合律和分配律。

那么，如果有两个四元数 $q_1 = [s, \boldsymbol{v}]$, $q_2 = [t, \boldsymbol{u}]$，则

$$\begin{aligned} q_1 q_2 &= (a + bi + cj + dk)(e + fi + gj + hk) \\ &= ae + afi + agj + ak \\ &+ bei + bfi^2 + bgij + bhik \\ &+ cej + cfji + cgj^2 + chjk \\ &+ dek + dfki + dgkj + dhk^2 \end{aligned} \tag{2.51}$$

利用四元数的性质 $i^2 = j^2 = k^2 = ijk = -1$，可以得到

$$ijk = -1 \xrightarrow{\text{等式两边同时左乘 } i} iijk = -i \quad \Rightarrow \quad jk = i \tag{2.52}$$

$$ijk = -1 \xrightarrow{\text{等式两边同时右乘 } k} ijkk = -k \quad \Rightarrow \quad ij = k \tag{2.53}$$

$$jk = i \xrightarrow{\text{等式两边同时左乘 } j} jjk = ji \quad \Rightarrow \quad ji = -k \tag{2.54}$$

$$jk = i \xrightarrow{\text{等式两边同时右乘 } k} jkk = ik \quad \Rightarrow \quad ik = -j \tag{2.55}$$

$$ij = k \xrightarrow{\text{等式两边同时左乘 } i} iij = ik \quad \Rightarrow \quad ik = -j \tag{2.56}$$

$$ij = k \xrightarrow{\text{等式两边同时右乘 } j} ijj = kj \quad \Rightarrow \quad kj = -i \tag{2.57}$$

将上面的性质整理为表格如表 2.2 所示。

$\times$	1	i	j	k
1	1	i	j	k
i	i	-1	k	$-j$
j	j	$-k$	-1	i
k	k	j	$-i$	-1

表 2.2: 四元数基本向量的乘法计算表（左 $\times$ 上）

评注

记忆方法：类似于向量的叉乘，将 i, j, k 理解为三维右手坐标系，则 $i \times j = k, j \times i = -k$，其余类似。

利用表 2.2 ，可以将四元数乘法的结果化简为

$$\begin{aligned} q_1 q_2 &= ae + afi + agj + ak \\ &+ bei + bfi^2 + bgij + bhik \\ &+ cej + cfji + cgj^2 + chjk \\ &+ dek + dfki + dgkj + dhk^2 \\ &= (ae - bf - cg - dh) \\ &+ (be - af - dg - ch)i \\ &+ (ce - df - ag - bh)j \\ &+ (de - cf - bg - ah)k \end{aligned} \tag{2.58}$$

写成矩阵为

$$q_1 q_2 = \begin{bmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix} \tag{2.59}$$

同理可得，右乘的结果为

$$q_2 q_1 = \begin{bmatrix} a & -b & -c & -d \\ b & a & d & c \\ c & -d & a & b \\ d & c & -b & a \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix} \tag{2.60}$$

5. [Grassmann 积]

为了将四元数的结果写成标量向量有序对，重新整理结果

$$\begin{aligned} q_1 q_2 = &(ae - (bf + cg + dh)) \\ &+ (be + af + ch - dg)i \\ &+ (ce + ag + df - bh)j \\ &+ (de + ah + bg - cf)k \end{aligned}$$

令 $\boldsymbol{v} = \begin{bmatrix} b \\ c \\ d \end{bmatrix}$, $\boldsymbol{u} = \begin{bmatrix} f \\ g \\ h \end{bmatrix}$，那么

$$\begin{aligned} \boldsymbol{v} \cdot \boldsymbol{u} &= bf + cg + dh \\ \boldsymbol{v} \times \boldsymbol{u} &= \begin{vmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ b & c & d \\ f & g & h \end{vmatrix} = (ch - dg)\boldsymbol{i} + (df - bh)\boldsymbol{j} + (bg - cf)\boldsymbol{k} \end{aligned}$$

所以，$q_1 q_2$ 的结果可以用向量点乘和叉乘的形式表示出来[1]

$$q_1 q_2 = \left[ae - \boldsymbol{v} \cdot \boldsymbol{u}, a\boldsymbol{u} + e\boldsymbol{v} + \boldsymbol{v} \times \boldsymbol{u}\right] \tag{2.61}$$

这个结果称为**Grassmann 积**，一般来说

定理 2.5 Grassmann 积

对于任意四元数 $q_1 = [s, \boldsymbol{v}]$, $q_2 = [t, \boldsymbol{u}]$，$q_1 q_2$ 的结果为

$$q_1 q_2 = \left[st - \boldsymbol{v} \cdot \boldsymbol{u}, s\boldsymbol{u} + t\boldsymbol{v} + \boldsymbol{v} \times \boldsymbol{u}\right] \tag{2.62}$$

6. [四元数的逆]

因为四元数不遵守交换律，所以类似于矩阵除法，四元数相除等价于乘以它的逆。同样地，也有左除和右除之分，即 pq^{-1} 或 $q^{-1}p$，它们的结果一般不同。

[1] 其实按照历史的顺序来说，这里第一次提出叉乘的概念。

定义 2.7 四元数的逆

如果

$$qq^{-1} = q^{-1}q = 1 \quad (q \neq 0) \tag{2.63}$$

那么，我们称 q^{-1} 为四元数 q 的**逆**。

直接求解一个四元数的逆 q^{-1} 是非常困难的，但是我们可以使用四元数共轭的性质来求 q^{-1}。

7. [共轭四元数]

定义 2.8 共轭四元数

一个四元数 $q = a + bi + cj + dk$ 的**共轭**为

$$q^* = a - bi - ci - dk \tag{2.64}$$

如果用标量向量有序对的形式来定义，则 $q = [s, \boldsymbol{v}]$ 的**共轭**为

$$q^* = [s, -\boldsymbol{v}] \tag{2.65}$$

共轭四元数的一个非常有用的性质就是

$$\begin{aligned} qq^* &= [s, \boldsymbol{v}] \cdot [s, -\boldsymbol{v}] \\ &= [s^2 - \boldsymbol{v} \cdot \boldsymbol{v}, s(-\boldsymbol{v}) + s\boldsymbol{v} + \boldsymbol{v} \times (-\boldsymbol{v})] \\ &= [s^2 + \boldsymbol{v} \cdot \boldsymbol{v}, 0] \\ &= s^2 + x^2 + y^2 + z^2 = \|q\|^2 \quad (\boldsymbol{v} = [x, y, z]) \end{aligned} \tag{2.66}$$

因为 $(q^*)^* = [s, -(-\boldsymbol{v})] = [s, \boldsymbol{v}] = q$，则

$$q^*q = (q^*)(q^*)^* = \|q^*\|^2 = s^2 + x^2 + y^2 + z^2 = \|q\|^2 = qq^* \tag{2.67}$$

这说明，$q^*q = qq^*$。这个特殊的乘法是遵守交换律的。

下面利用四元数的性质求四元数的逆。

$$qq^{-1} = 1 \xrightarrow{\text{等式两边同时左乘 } q^*} q^*qq^{-1} = 1 \quad \Rightarrow \quad (q^*q)q^{-1} = q^* \xrightarrow{q^*q = \|q\|^2} \|q\|^2 \cdot q^{-1} = q^*$$

由此可知

定理 2.6 四元数逆的求解

四元数的逆可以表示为

$$q^{-1} = \frac{q^*}{\|q\|^2} \tag{2.68}$$

用这种方法寻找一个四元数的逆会非常高效，我们只需要将一个四元数的共轭除以它的模长的平方就可以得到四元数的逆。特别地对于**单位四元数**$\|q\| = 1$ 来说，它的逆为

$$q^{-1} = \frac{q^*}{1^2} = q^* \tag{2.69}$$

8. [纯四元数]

定义 2.9　纯四元数

如果一个四元数实部为 0，仅有虚部，即

$$v = [0, \boldsymbol{v}] \tag{2.70}$$

那么我们则称这个四元数 v 为一个**纯四元数**。因为纯四元数仅由虚部的三维向量决定，我们可以将任意的三维向量转换为纯四元数。

纯四元数有一个很重要的特性：两个纯四元数 $v = [0, \boldsymbol{v}], u = [0, \boldsymbol{u}]$ 的乘积为

$$vu = [0 - \boldsymbol{v} \cdot \boldsymbol{u}, 0 + \boldsymbol{v} \times \boldsymbol{u}] = [-\boldsymbol{v} \cdot \boldsymbol{u}, \boldsymbol{v} \times \boldsymbol{u}] \tag{2.71}$$

2.5.3　四元数与三维旋转

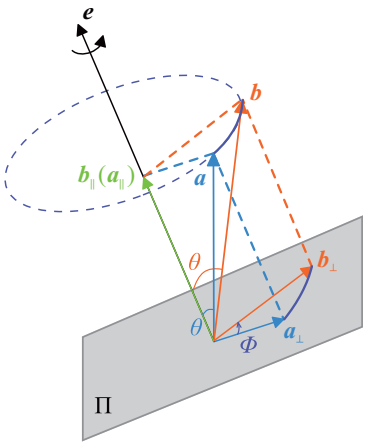

图 2.10: 向量旋转分解图

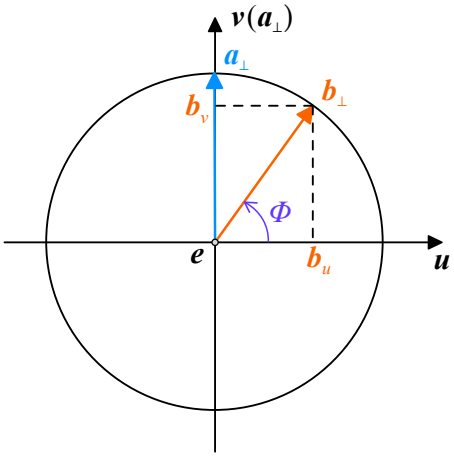

图 2.11: 向量旋转投影图

在上个章节 2.4 中我们知道：如果我们需要将一个向量 $\boldsymbol{a}$ 沿着一个用单位向量 $\boldsymbol{e}$ 所定义的旋转轴旋转 Φ 度，那么我们可以将这个向量 $\boldsymbol{a}$ 可以分解为平行于旋转轴的分量 $a_{\parallel}$ 和垂直于旋转轴的分量 $a_{\perp}$。旋转后得到的向量 $\boldsymbol{b}$ 可以表示为这两个分量分别旋转后之和，即 $\boldsymbol{b} = \boldsymbol{b}_{\parallel} + \boldsymbol{b}_{\perp}$。

我们定义这些向量为纯四元数

$$a = [0, \boldsymbol{a}] \qquad b = [0, \boldsymbol{b}]$$
$$a_{\perp} = [0, \boldsymbol{a}_{\perp}] \qquad b_{\perp} = [0, \boldsymbol{b}_{\perp}]$$
$$a_{\parallel} = [0, \boldsymbol{a}_{\parallel}] \qquad b_{\parallel} = [0, \boldsymbol{b}_{\parallel}]$$
$$e = [0, \boldsymbol{e}]$$

那么我们可以得到

$$a = a_{\parallel} + a_{\perp} \qquad b = b_{\parallel} + b_{\perp}$$

如图 2.10 所示，我们知道，旋转前后平行分量 $\boldsymbol{a}_{\parallel} = \boldsymbol{b}_{\parallel} = \boldsymbol{e}$ 相等，即旋转前后不变，下面考虑垂直分量 $\boldsymbol{a}_{\perp}$ 的旋转。在上个章节 2.4 中我们推导过垂直分量 $\boldsymbol{a}_{\perp}$ 的旋转公式

$$\boldsymbol{b}_{\perp} = \cos\Phi \boldsymbol{a}_{\perp} + \sin\Phi(\boldsymbol{e} \times \boldsymbol{a}_{\perp})$$

我们可以很容易将 $a'_\perp$ 和 $a_\perp$ 写成四元数的形式，而对于 $\boldsymbol{a}_\perp \times \boldsymbol{e}$，由纯四元数的重要性质，由第 25 页的公式 (2.71) 可知，纯四元数 $a = [0, \boldsymbol{a}_\perp]$, $e = [0, \boldsymbol{e}]$，那么

$$ea_\perp = \left[-\boldsymbol{e} \cdot \boldsymbol{a}_\perp, \boldsymbol{e} \times \boldsymbol{a}_\perp \right]$$

而 $\boldsymbol{e} \perp \boldsymbol{a}_\perp \Rightarrow \boldsymbol{e} \cdot \boldsymbol{a}_\perp = 0$，所以

$$ea_\perp = \left[0, \boldsymbol{e} \times \boldsymbol{a}_\perp\right] = \boldsymbol{e} \times \boldsymbol{a}_\perp$$

所以 $\boldsymbol{a}_\perp$ 旋转公式的四元数表达为

$$b_\perp = \cos \Phi a_\perp + \sin \Phi (ea_\perp) = \left(\cos \Phi + \sin \Phi e\right) a_\perp \tag{2.72}$$

令 $q = \cos \Phi + \sin \Phi e$，则可以得到

$$b_\perp = q a_\perp \tag{2.73}$$

所以，我们只需要构造一个四元数 q，就可以完成垂直分量的旋转。对 q 进一步变形，得

$$\begin{aligned} q &= \cos \Phi + \sin \Phi e \\ &= \left[\cos \Phi, \boldsymbol{0}\right] + \left[0, \sin \Phi \boldsymbol{e}\right] \\ &= \left[\cos \Phi, \sin \Phi \boldsymbol{e}\right] \end{aligned} \tag{2.74}$$

如果已经知道旋转轴的坐标 $\boldsymbol{e} = \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix}$ 和旋转角 Φ，那么四元数 q 就确定为

$$q = \left[\cos \Phi, \sin \Phi \boldsymbol{e}\right] = \cos \Phi + \sin \Phi e_x \boldsymbol{i} + \sin \Phi e_y \boldsymbol{j} + \sin \Phi e_z \boldsymbol{k} \tag{2.75}$$

四元数 q 还有一个性质（注：$\|\boldsymbol{u}\| = 1$）

$$\begin{aligned} \|q\| &= \sqrt{\cos^2 \Phi + (\sin \Phi \boldsymbol{u} \cdot \sin \Phi \boldsymbol{u})} \\ &= \sqrt{\cos^2 \Phi + \sin^2 \Phi (\boldsymbol{u} \cdot \boldsymbol{u})} \\ &= \sqrt{\cos^2 \Phi + \sin^2 \Phi} = 1 \end{aligned} \tag{2.76}$$

说明旋转四元数参数 q 是单位四元数，一定意义上表面它所做的变换并不会对原向量进行缩放，是一个纯旋转。

又因为平行分量旋转前后不改变，即 $\boldsymbol{b}_\parallel = \boldsymbol{a}_\parallel \Rightarrow b_\parallel = a_\parallel$ 所以，向量 $\boldsymbol{a}$ 的最终四元数旋转表示为

$$\begin{aligned} b &= b_\parallel + b_\perp \\ &= a_\parallel + q a_\perp \qquad (\text{其中 } q = \left[\cos \Phi, \sin \Phi \boldsymbol{e}\right]) \end{aligned} \tag{2.77}$$

通过进一步的化简[1]，可以得到

[1] 由于篇幅所限，此部分化简内容请参见第 3 章：3.1 旋转四元数参数的化简，Page 41

定理 2.7 向量旋转的四元数参数

在同一坐标系下，任意向量 $\boldsymbol{a}$ 沿着以单位向量定义的旋转轴 $\boldsymbol{e}$ 旋转 Φ 度之后的向量 $\boldsymbol{b}$ 可以用四元数表达。令 $a=[0,\boldsymbol{a}]$，$q=\left[\cos\left(\frac{1}{2}\Phi\right),\sin\left(\frac{1}{2}\Phi\right)\boldsymbol{e}\right]$，那么

$$b=qaq^*=qaq^{-1} \tag{2.78}$$

为了更好的理解这个公式的意义，我们可以得到[2]

$$b=qaq^*=qq^*a_{\parallel}+qqa_{\perp}=a_{\parallel}+q^2a_{\perp} \tag{2.79}$$

也就是说，qaq^* 这个变换，实际上可以等价为，对 a 平行于坐标轴的分量 $a_{\parallel}$ 实施的变换是 qq^*，这两个变换是互逆的，完全抵消了，也就是没有旋转。而对于正交于旋转轴的分量 $a_{\perp}$，则实施的是两次完全一样的变换 $q^2=qq$，将它旋转 $\frac{\Phi}{2}+\frac{\Phi}{2}=\Phi$ 度。

由这个公式的由来，可以发现其和上一章节的欧拉轴 / 角的证明是类似的。实际上，旋转四元数参数和欧拉轴 / 角时完全等价的[3]，即

$$b=qaq^*=\cos\Phi\boldsymbol{a}+(1-\cos\Phi)(\boldsymbol{e}\cdot\boldsymbol{a})\boldsymbol{e}+\sin\Phi(\boldsymbol{e}\times\boldsymbol{a}) \tag{2.80}$$

若已经知道旋转四元数参数 $q=[q_0,\boldsymbol{q}]$，则可以很容易获得其对应的旋转角度和旋转轴向量

$$\frac{\Phi}{2}=\cos^{-1}q_0 \tag{2.81}$$

$$\boldsymbol{e}=\frac{\boldsymbol{q}}{\sin\left(\cos^{-1}q_0\right)} \tag{2.82}$$

2.5.4 欧拉参数

1. [欧拉参数的定义和表示]

为了和方向余弦矩阵具有一致的形式，我们可以将四元数写成矩阵的形式。在四元数的乘法中，我们计算过左乘矩阵和右乘矩阵，若乘数 $q=q_0+q_1i+q_2j+q_3k$，那么左乘矩阵和右乘矩阵可以分别表示为

$$L(q)=\begin{bmatrix}q_0&-q_1&-q_2&-q_3\\q_1&q_0&-q_3&q_2\\q_2&q_3&q_0&-q_1\\q_3&-q_2&q_1&q_0\end{bmatrix},\qquad R(q)=\begin{bmatrix}q_0&-q_1&-q_2&-q_3\\q_1&q_0&q_3&-q_2\\q_2&-q_3&q_0&q_1\\q_3&q_2&-q_1&q_0\end{bmatrix} \tag{2.83}$$

同时，对于四元数 q 的共轭 q^* 的乘数矩阵，有

$$L(q^*)=\begin{bmatrix}q_0&q_1&q_2&q_3\\-q_1&q_0&q_3&-q_2\\-q_2&-q_3&q_0&q_1\\-q_3&q_2&-q_1&q_0\end{bmatrix}=[L(q)],\qquad R(q^*)=\begin{bmatrix}q_0&q_1&q_2&q_3\\-q_1&q_0&-q_3&q_2\\-q_2&q_3&q_0&-q_1\\-q_3&-q_2&q_1&q_0\end{bmatrix}=[R(q)] \tag{2.84}$$

[2]公式 (2.79) 是原式公式化简第一步的结果，详情请见第 42 页的公式 (3.2) 。

[3]其证明见第 3 章：3.2 旋转的欧拉轴 / 角参数表达和四元数表达的等价性，Page 43

取

$$\begin{cases} q = [q_0, \boldsymbol{q}] = [q_0 \quad q_1 \quad q_2 \quad q_3], \\ q_0 = \cos\dfrac{\Phi}{2}, \qquad q_1 = e_x \sin\dfrac{\Phi}{2} \\ q_2 = e_y \cos\dfrac{\Phi}{2}, \qquad q_3 = e_z \sin\dfrac{\Phi}{2} \end{cases}$$

其中，四元数 q 是旋转四元数参数，也称为**欧拉参数**。那么我们可以得到

$$\begin{aligned} b = qaq^* &= L(q)R(q^*)a \ \left[= R(q^*)L(q)a \ \right] \\ &= \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \\ -q_1 & q_0 & -q_3 & q_2 \\ -q_2 & q_3 & q_0 & -q_1 \\ -q_3 & -q_2 & q_1 & q_0 \end{bmatrix} a \\ &= \begin{bmatrix} q_0^2+q_1^2+q_2^2+q_3^2 & q_0q_1-q_1q_0-q_2q_3+q_3q_2 & q_0q_2+q_1q_3-q_2q_0-q_3q_1 & q_0q_3-q_1q_2+q_2q_1-q_3q_0 \\ q_1q_0-q_0q_1+q_3q_2-q_2q_3 & q_1^2+q_0^2-q_3^2-q_2^2 & q_1q_2-q_0q_3-q_3q_0+q_2q_1 & q_1q_3+q_0q_2+q_3q_1+q_2q_0 \\ q_2q_0-q_1q_3-q_0q_2+q_1q_3 & q_2q_1+q_3q_0+q_0q_3+q_1q_2 & q_2^2-q_3^2+q_0^2-q_1^2 & q_2q_3+q_3q_2-q_0q_1-q_1q_0 \\ q_3q_0+q_2q_1-q_1q_2-q_0q_3 & q_3q_1-q_2q_0+q_1q_3-q_0q_2 & q_3q_2+q_2q_3+q_1q_0+q_0q_1 & q_3^2-q_2^2-q_1^2+q_0^2 \end{bmatrix} a \end{aligned}$$

由于 $q_0^2+q_1^2+q_2^2+q_3^2=1$（旋转四元数参数 q 是一个单位四元数），所以进一步化简为

$$b = qaq^* = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1-2(q_2^2+q_3^2) & 2(q_1q_2-q_3q_0) & 2(q_1q_3+q_2q_0) \\ 0 & 2(q_1q_2+q_3q_0) & 1-2(q_1^2+q_3^2) & 2(q_2q_3-q_1q_0) \\ 0 & 2(q_3q_1-q_2q_0) & 2(q_2q_3+q_1q_0) & 1-2(q_1^2+q_2^2) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} \tag{2.85}$$

其中，$a = [a_0 \quad a_1 \quad a_2 \quad a_3]$。由矩阵乘法可知，矩阵的最外圈不会对 a 进行任何变换，而代表 b 的方向仅取决于 $[a_1 \quad a_2 \quad a_3]$。因此，我们可以删去矩阵最外圈，得到一个 3×3 的矩阵用于三维向量的旋转变换，即

定理 2.8　欧拉参数表示的向量旋转矩阵

在同一坐标系下，任意向量 $\boldsymbol{a}$ 沿着以单位向量定义的旋转轴 $\boldsymbol{e}$ 旋转 Φ 角度之后的向量 $\boldsymbol{b}$ 可以用矩阵乘法来获得

$$\begin{cases} q_0 = \cos\dfrac{\Phi}{2} \\ q_1 = e_x \sin\dfrac{\Phi}{2} \\ q_2 = e_y \sin\dfrac{\Phi}{2} \\ q_3 = e_z \sin\dfrac{\Phi}{2} \end{cases} \Longrightarrow \boldsymbol{b} = \boldsymbol{R}_{ba}(q)\boldsymbol{a}, \ \boldsymbol{R}_{ba}(q) = \begin{bmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2-q_3q_0) & 2(q_1q_3+q_2q_0) \\ 2(q_1q_2+q_3q_0) & 1-2(q_1^2+q_3^2) & 2(q_2q_3-q_1q_0) \\ 2(q_3q_1-q_2q_0) & 2(q_2q_3+q_1q_0) & 1-2(q_1^2+q_2^2) \end{bmatrix} \tag{2.86}$$

同样地，由向量旋转与坐标系旋转的对应关系，可以知道向量旋转矩阵和坐标系旋转矩阵（方向余弦矩阵）互为转置（逆），所以可以得到[1]

[1] 证明详见第 3 章：3.3 向量旋转矩阵和坐标系旋转矩阵的关系，Page 45

定理 2.9　欧拉参数表示的坐标旋转矩阵（方向余弦矩阵）

欧拉参数表示的坐标旋转矩阵（方向余弦矩阵）为

$$\begin{cases} q_0 = \cos\dfrac{\Phi}{2} \\ q_1 = e_x \sin\dfrac{\Phi}{2} \\ q_2 = e_y \sin\dfrac{\Phi}{2} \\ q_3 = e_z \sin\dfrac{\Phi}{2} \end{cases} \Longrightarrow \boldsymbol{e}_b = \boldsymbol{C}_{ba}(q)\boldsymbol{e}_a,\ \boldsymbol{C}_{ba}(q) = \begin{bmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2+q_3q_0) & 2(q_1q_3-q_2q_0) \\ 2(q_1q_2-q_3q_0) & 1-2(q_1^2+q_3^2) & 2(q_2q_3+q_1q_0) \\ 2(q_3q_1+q_2q_0) & 2(q_2q_3-q_1q_0) & 1-2(q_1^2+q_2^2) \end{bmatrix} \tag{2.87}$$

同时对于四元数 $e_a = [0, \boldsymbol{e}_a]$, $e_b = [0, \boldsymbol{e}_b]$，有

$$e_b = q^* e_a q \tag{2.88}$$

虽然三维旋转的矩阵形式（欧拉参数）可能不如四元数的形式简单，而且占用更多的空间，但是对于大批量的转换，使用**预先计算**好的矩阵是比四元数乘法更有效率的。

我们知道方向余弦矩阵的欧拉轴 / 角参数表述为

$$\boldsymbol{C}_{ba} = \cos\Phi\boldsymbol{E}_3 + (1-\cos\Phi)\boldsymbol{e}\boldsymbol{e} - \sin\Phi\boldsymbol{e}^\times \tag{2.89}$$

类似的，我们也可以将欧拉参数写成矩阵运算的形式，注意到

$$\begin{aligned} q_0 &= \cos\frac{\Phi}{2} \\ \boldsymbol{q} &= \boldsymbol{e}\sin\frac{\Phi}{2} \\ \boldsymbol{q} &= \boldsymbol{e}\sin\frac{\Phi}{2} \\ \boldsymbol{q}^\times &= \begin{bmatrix} 0 & -e_z\sin\dfrac{\Phi}{2} & e_y\sin\dfrac{\Phi}{2} \\ e_z\sin\dfrac{\Phi}{2} & 0 & -e_x\sin\dfrac{\Phi}{2} \\ -e_y\sin\dfrac{\Phi}{2} & e_x\sin\dfrac{\Phi}{2} & 0 \end{bmatrix} = \boldsymbol{e}^\times\sin\frac{\Phi}{2} \end{aligned}$$

$$\boldsymbol{q}^T\boldsymbol{q} = \begin{bmatrix} e_x\sin\dfrac{\Phi}{2} & e_y\sin\dfrac{\Phi}{2} & e_z\sin\dfrac{\Phi}{2} \end{bmatrix} \begin{bmatrix} e_x\sin\dfrac{\Phi}{2} \\ e_y\sin\dfrac{\Phi}{2} \\ e_z\sin\dfrac{\Phi}{2} \end{bmatrix} = (e_x^2+e_y^2+e_z^2)\sin^2\frac{\Phi}{2} = \sin^2\frac{\Phi}{2}$$

对欧拉轴 / 角参数型方向余弦矩阵进行变换，得

$$\begin{aligned} \boldsymbol{C}_{ba} &= \cos\Phi\boldsymbol{E}_3 + (1-\cos\Phi)\boldsymbol{e}\boldsymbol{e} - \sin\Phi\boldsymbol{e}^\times \\ &= \left(1-2\sin^2\frac{\Phi}{2}\right)\boldsymbol{E}_3 + 2\sin^2\frac{\Phi}{2}\boldsymbol{e}\boldsymbol{e} - 2\sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}^\times \\ &= \left(1-2\sin^2\frac{\Phi}{2}\right)\boldsymbol{E}_3 + 2\left(\boldsymbol{e}\sin\frac{\Phi}{2}\right)\left(\boldsymbol{e}\sin\frac{\Phi}{2}\right) - 2\cos\frac{\Phi}{2}\left(\boldsymbol{e}^\times\sin\frac{\Phi}{2}\right) \\ &= (1-2\boldsymbol{q}^{\mathrm{T}}\boldsymbol{q})\boldsymbol{E}_3 + 2\boldsymbol{q}\boldsymbol{q}^{\mathrm{T}} - 2q_0\boldsymbol{q}^\times \end{aligned} \tag{2.90}$$

2. [欧拉参数与方向余弦矩阵之间的转换]

通过比对欧拉参数矩阵和方向余弦矩阵系数

$$\boldsymbol{C}_{ba}=\begin{bmatrix}C_{11}&C_{12}&C_{13}\\C_{21}&C_{22}&C_{23}\\C_{31}&C_{32}&C_{33}\end{bmatrix}=\begin{bmatrix}1-2(q_2^2+q_3^2)&2(q_1q_2+q_3q_0)&2(q_1q_3-q_2q_0)\\2(q_1q_2-q_3q_0)&1-2(q_1^2+q_3^2)&2(q_2q_3+q_1q_0)\\2(q_3q_1+q_2q_0)&2(q_2q_3-q_1q_0)&1-2(q_1^2+q_2^2)\end{bmatrix} \tag{2.91}$$

当方向余弦矩阵系数已知时，可以求出四元数 q

$$\begin{cases}q_0=\pm\dfrac{1}{2}\sqrt{1+C_{11}+C_{22}+C_{33}}\\q_1=\dfrac{1}{4q_0}(C_{23}-C_{32})\\q_2=\dfrac{1}{4q_0}(C_{31}-C_{13})\\q_3=\dfrac{1}{4q_0}(C_{12}-C_{21})\end{cases} \tag{2.92}$$

当 $q_0=0$ 时，公式 (2.92) 是奇异的，无法求解 q_1,q_2,q_3，通过比对系数还可以获得其余三组解：

$$\begin{cases}q_1=\pm\dfrac{1}{2}\sqrt{1+C_{11}-C_{22}-C_{33}}\\q_2=\dfrac{1}{4q_1}(C_{12}+C_{21})\\q_3=\dfrac{1}{4q_1}(C_{13}+C_{31})\\q_0=\dfrac{1}{4q_1}(C_{23}-C_{32})\end{cases} \tag{2.93}$$

$$\begin{cases}q_2=\pm\dfrac{1}{2}\sqrt{1-C_{11}+C_{22}-C_{33}}\\q_3=\dfrac{1}{4q_2}(C_{23}+C_{32})\\q_0=\dfrac{1}{4q_2}(C_{31}-C_{13})\\q_1=\dfrac{1}{4q_2}(C_{12}+C_{21})\end{cases} \tag{2.94}$$

$$\begin{cases}q_3=\pm\dfrac{1}{2}\sqrt{1-C_{11}-C_{22}+C_{33}}\\q_0=\dfrac{1}{4q_3}(C_{12}-C_{21})\\q_1=\dfrac{1}{4q_3}(C_{13}+C_{31})\\q_2=\dfrac{1}{4q_3}(C_{23}+C_{32})\end{cases} \tag{2.95}$$

所以，一共有四种计算方法求解欧拉参数 $q=\begin{bmatrix}q_0&q_1&q_2&q_3\end{bmatrix}$，为了降低计算结果的误差，我们选取奇异

性最小的一组解，即通过计算

$$\begin{cases} q_0 = \pm\frac{1}{2}\sqrt{1+C_{11}+C_{22}+C_{33}} \\ q_1 = \pm\frac{1}{2}\sqrt{1+C_{11}-C_{22}-C_{33}} \\ q_2 = \pm\frac{1}{2}\sqrt{1-C_{11}+C_{22}-C_{33}} \\ q_3 = \pm\frac{1}{2}\sqrt{1-C_{11}-C_{22}+C_{33}} \end{cases} \xrightarrow{\text{选取最大的一个作为解}} q_j = \max_{i=1,2,3,4} q_i \tag{2.96}$$

这样以后，后面计算用到的 $\frac{1}{4q_j}$ 的奇异性最小，计算的精度最高。

上面计算得到的欧拉参数称为从坐标系 S_a 到坐标系 S_b 的欧拉参数，或者坐标系 S_b 相对于坐标系 S_a 的欧拉参数。为了更明确的显示坐标系的转换关系，将其记为 q_{ba}。

3. [相继转动的欧拉参数表示]

假设有两个表示沿着不同轴，不同角度旋转的非零四元数 q_{ba}, q_{cb}，我们先对 a 进行 q_{ba} 变换得到 b

$$b = q_{ba} a q_{ba}^*$$

再对 b 进行 q_{cb} 变换得到 c

$$\begin{aligned} c &= q_{cb} b q_{cb}^* \\ &= q_{cb} q_{ba} a q_{ba}^* q_{cb}^* \end{aligned}$$

下面给出一个引理

引理 2.1

对任意四元数 $q_1 = [s, \boldsymbol{v}]$, $q_2 = [t, \boldsymbol{u}]$，有

$$q_1^* q_2^* = (q_2 q_1)^* \tag{2.97}$$

证　由 Grassmann 积，

$$\begin{aligned} \text{LHS} &= q_1^* q_2^* \\ &= [s, -\boldsymbol{v}] \cdot [t, -\boldsymbol{u}] \\ &= [st - (-\boldsymbol{v}) \cdot (-\boldsymbol{u}), s(-\boldsymbol{u}) + t(-\boldsymbol{v}) + (-\boldsymbol{v}) \times (-\boldsymbol{u})] \\ &= [st - \boldsymbol{v} \cdot \boldsymbol{u}, -s\boldsymbol{u} - t\boldsymbol{v} + \boldsymbol{v} \times \boldsymbol{u}] \end{aligned}$$

$$\begin{aligned} \text{RHS} = (q_2 q_1)^* &= \left([t, \boldsymbol{u}] \cdot [s, \boldsymbol{v}]\right)^* \\ &= [ts - \boldsymbol{u} \cdot \boldsymbol{v}, t\boldsymbol{v} + s\boldsymbol{u} + \boldsymbol{u} \times \boldsymbol{v}]^* \\ &= [st - \boldsymbol{v} \cdot \boldsymbol{u}, -s\boldsymbol{u} - t\boldsymbol{v} - \boldsymbol{u} \times \boldsymbol{v}] \\ &= [st - \boldsymbol{v} \cdot \boldsymbol{u}, -s\boldsymbol{u} - t\boldsymbol{v} + \boldsymbol{v} \times \boldsymbol{u}] = \text{LHS} \end{aligned}$$

□

所以，根据 引理 **2.1** 可以得到

$$\begin{aligned} c &= q_{cb}q_{ba}aq_{ba}^*q_{cb}^* \\ &= (q_{cb}q_{ba})a(q_{cb}q_{ba})^* \\ &= q_{ca}aq_{ca}^* \end{aligned} \tag{2.98}$$

其中，$q_{ca} = q_{cb} \cdot q_{ba}$。也就是说，坐标系 S_a 通过两次旋转 q_{ba}, q_{cb} 后得到坐标系 S_c 等价于一次旋转 q_{ca}。

注意

旋转顺序与四元数乘法顺序

我们先进行的是 q_{ba} 变换，再进行 q_{cb} 变换，其对应的四元数乘法为 $q_{ca} = q_{cb} \cdot q_{ba}$，即先进行的变换放在后面，对于多旋转来说也符合这个计算顺序。

我们这个只是两个旋转的复合，很容易可以推广到多旋转，即

$$d = q_{dc}(q_{cb}q_{ba})a(q_{cb}q_{ba})^*q_3^* = (q_{dc}q_{cb}q_{ba})a(q_{dc}q_{cb}q_{ba})^* = q_{da}aq_{da}^* \tag{2.99}$$

2.6 欧式变换

2.6.1 欧式变换

上面介绍的都是坐标系旋转的几种描述方式，实际上欧式变换中，除了旋转还有平移。考虑世界坐标系中的向量 $\boldsymbol{a}$，经过一次旋转（这里用旋转矩阵 $\boldsymbol{R}$ 描述）和一次平移 $\boldsymbol{t}$ 后，得到了 $\boldsymbol{a}'$，那么把旋转和平移合到一起，有

$$\boldsymbol{a}' = \boldsymbol{R}\boldsymbol{a} + \boldsymbol{t} \tag{2.100}$$

其中，$\boldsymbol{t}$ 称为**平移向量**。在实际情况下，我们会定义坐标系 1、坐标系 2，那么向量 $\boldsymbol{a}$ 在两个坐标系下的坐标分别为 $\boldsymbol{a}_1, \boldsymbol{a}_2$，它们之间的关系为

$$\boldsymbol{a}_1 = \boldsymbol{R}_{12}\boldsymbol{a}_2 + \boldsymbol{t}_{12} \tag{2.101}$$

注意

旋转矩阵和平移向量的下标

旋转矩阵 $\boldsymbol{R}_{12}$ 表示的是“把坐标系 2 的向量分量变换到坐标系 1 中”。由于向量乘在这个矩阵的右边，它的下标是**从右读到左**的。

平移向量 $\boldsymbol{t}_{12}$ 表示的是“坐标系 1 原点指向坐标系 2 原点的向量”，这个向量应该表示为在**坐标系 1 下取的分量**。即平移向量的分量 $\boldsymbol{t}_{12} \neq -\boldsymbol{t}_{21}$，这是因为这两个分量是不同坐标系下的分量。

2.6.2 变换矩阵与齐次坐标

公式 (2.101) 完整地表达了欧式空间的旋转和平移，但是这里的变换不是一个线性关系。假设我们进行了两次变换 $\boldsymbol{R}_1, \boldsymbol{t}_1$ 和 $\boldsymbol{R}_1, \boldsymbol{t}_2$，即

$$\boldsymbol{b} = \boldsymbol{R}_1\boldsymbol{a} + \boldsymbol{t}_1 \qquad \boldsymbol{c} = \boldsymbol{R}_2\boldsymbol{b} + \boldsymbol{t}_2$$

那么 $\boldsymbol{c}, \boldsymbol{a}$ 之间的转换为

$$\boldsymbol{c} = \boldsymbol{R}_2(\boldsymbol{R}_1\boldsymbol{a} + \boldsymbol{t}_1) + \boldsymbol{t}_2$$

这样的形式在变换多次后会显得很复杂。因此，我们引入其次坐标和坐标变换，重写公式 (2.101)

$$\begin{bmatrix} \boldsymbol{a}' \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}^{\mathrm{T}} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{a} \\ 1 \end{bmatrix} \xlongequal{\text{def}} \boldsymbol{T} \begin{bmatrix} \boldsymbol{a} \\ 1 \end{bmatrix} \tag{2.102}$$

这是一个数学技巧：我们在一个三维向量的末尾添加 1，将其变成了四维向量，称为**齐次坐标**。对于这个四维向量，我们可以把旋转和平移写在一个矩阵里，使得整个关系变成线性关系。矩阵 $\boldsymbol{T}$ 称为**变换矩阵**。

那么记 $\tilde{\boldsymbol{a}}$ 表示 $\boldsymbol{a}$ 的齐次坐标。依靠其次坐标和变换矩阵，两次变换的叠加就有很好的形式

$$\tilde{\boldsymbol{b}} = \boldsymbol{T}_1 \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{c}} = \boldsymbol{T}_2 \tilde{\boldsymbol{b}} \quad \Rightarrow \tilde{\boldsymbol{c}} = \boldsymbol{T}_1 \boldsymbol{T}_1 \tilde{\boldsymbol{a}} \tag{2.103}$$

为了方便表示，在不引起歧义的情况下，不区分齐次和非齐次坐标的符号。

变换矩阵 $\boldsymbol{T}$ 具有比较特别的结构：左上角为旋转矩阵，右侧为平移向量，左下角为 $\boldsymbol{0}$ 向量，右下角为 1。其逆矩阵表示为反向的变换，即

$$\boldsymbol{T}^{-1} = \begin{bmatrix} \boldsymbol{R}^{\mathrm{T}} & -\boldsymbol{R}^{\mathrm{T}}\boldsymbol{t} \\ \boldsymbol{0}^{\mathrm{T}} & 1 \end{bmatrix} \tag{2.104}$$

2.7 相似、仿射、射影变换

除了欧式变换，3D 空间还存在其他几种变换方式，只不过欧式变换是最简单的。它们一部分和测量几何有关。欧式变换保持了向量的长度和夹角，相当于我们把一个刚体原封不动地进行了移动或旋转，不改变它自身的样子。其他几种变换则会改变它的外形。

2.7.1 相似变换

相似变换比欧式变换多了一个自由度，它允许物体进行均匀缩放，其矩阵表示为

$$\boldsymbol{T}_S = \begin{bmatrix} s\boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}^{\mathrm{T}} & 1 \end{bmatrix} \tag{2.105}$$

相似变换矩阵中的旋转部分多了一个缩放因子 s，表示我们在对向量旋转之后，可以在 x, y, z 三个坐标上均匀缩放。由于含有缩放，相似变换不再保持图形的面积不变。

2.7.2 仿射变换

仿射变换的矩阵表示为

$$\boldsymbol{T}_A = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{t} \\ \boldsymbol{0}^{\mathrm{T}} & 1 \end{bmatrix} \tag{2.106}$$

与欧式变换不同的是，仿射变换只要求 $\boldsymbol{A}$ 是一个可逆矩阵，而不必是正交矩阵。仿射变换也叫正交投影。经过仿射变换之后，立方体就不再是方的了，但各个面仍然是平行四边形。

2.7.3 射影变换

射影变换是最一般的变换，其变换矩阵表示为

$$\boldsymbol{T}_P = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{t} \\ \boldsymbol{a}^{\mathrm{T}} & v \end{bmatrix} \tag{2.107}$$

它的左上角为可逆矩阵 $\boldsymbol{A}$，右上角为平移矩阵 $\boldsymbol{t}$，左下角为缩放 $\boldsymbol{a}^{\mathrm{T}}$。由于采用了齐次坐标，当 $v \neq 0$ 时，我们可以对整个矩阵除以 v 得到一个右下角为 1 的矩阵；否则得到右下角为 0 的矩阵。因此，2D 的射影变换一共有 8 个自由度，3D 则有 15 个自由度。射影变换是最一般的变换。从真是世界到相机照片的变换可以看成一个摄影变换。一个原本方形的地板砖在照片中的变化：首先不是方形，其次由于金达远小的关系，它甚至不是平行四边形，是一个不规则的四边形。

各个变换的性质如表 2.3 所示。注意在“不变性质”中，从上到下是由包含关系的。

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

表 2.3: 常见变换的性质比较

2.8 C++ 实践：Eigen 库

C++ 实践主要基于教材和对应 Github 上的开源代码进行整理，结合查找的资料进行归纳整理，下面将进行分块讲解，如果不需要分块讲解，可以直接跳往第 4 章 4.1 Eigen，Page 47 查看例子源码.

Eigen 是一个 C++ 开源线性代数库。它提供了快速的有关矩阵的线性代数运算，还包括解方程等功能。在使用 Eigen 之前，需要将 Eigen 库文件目录导入到 CMakeList.txt 文件内。

```
# 添加头文件
include_directories("/usr/include/eigen3")
```

特别地，由于 Eigen 库只有头文件，所以不需要再用 `target_link_libraries` 语句将程序链接到库上。使用 Eigen 只需要在程序中加入头文件即可。

```
1 // Eigen 核心部分
2 #include <Eigen/Core>
3 // 稠密矩阵的代数运算（逆，特征值等）
4 #include <Eigen/Dense>
```

2.8.1 矩阵的定义

Eigen 中所有向量和矩阵都是 Eigen::Matrix，它是一个模板类。它的前三个参数为：数据类型，行，列。

```
1 Matrix<double, 3, 3> A;                 // 定义3x3double 矩阵
2 Matrix<double, 3, Dynamic> A;           //定义3xn double 矩阵,列为动态变化
```

```
Matrix<double, Dynamic, Dynamic>  A;   // 定义 double 矩阵,行、列为动态变化，由需要决定
MatrixXd A;                            // 定义 double 矩阵,行、列为动态变化，由需要决定
Matrix<double, 3, 3, RowMajor>  A;     // 定义3x3 double 矩阵,按行储存，默认按列储存效率较高。
Matrix3f  A;                           // 定义3x3 float 矩阵A.
Vector3f  A;                           // 定义3x1 float 列向量A.
VectorXd  A;                           // 定义动态double列向量A
RowVector3f  A;                        // 定义1x3 float 行向量A.
RowVectorXd  A;                        // 定义动态double行向量A.
```

2.8.2 矩阵的基本操作

1. 访问元素及大小

```
// 下面是对Eigen阵的操作
A.size()        // 元素个数
C.rows()        // 行个数
C.cols()        // 列个数

A(i)            // x(i+1)           // 默认情况下列优先，访问(0,i)的元素
C(i, j)         // C(i+1,j+1)       //访问（i, j）的元素。
A.resize(4, 4); // 如果之前已经定义过形状则会报错。
B.resize(4, 9); // 如果之前已经定义过形状则会报错。

A << 1, 2, 3,     // 初始化矩阵A
     4, 5, 6,     // 初始化的输入元素既可以是数字也可以是矩阵
     7, 8, 9;     // 可以将矩阵合并为一个大矩阵
B << A, A, A;     // 三个A矩阵按行排列合并为B
A.fill(10);       // A所有元素都赋值10
```

2. 特殊的矩阵

```
// Eigen
//单位矩阵定义
MatrixXd::Identity(rows,cols)
C.setIdentity(rows,cols)
//零矩阵定义
MatrixXd::Zero(rows,cols)
C.setZero(rows,cols)
//全1矩阵定义
MatrixXd::Ones(rows,cols)
C.setOnes(rows,cols)
//随即矩阵定义
MatrixXd::Random(rows,cols)
C.setRandom(rows,cols)
//线性阵定义
VectorXd::LinSpaced(size,low,high)
v.setLinSpaced(size,low,high)
```

3. 矩阵分块操作

```
// 下面x为列或行向量，P为矩阵
```

```
/*************************只能对向量操作*************************/
x.head(n)                          // 列向量的前n个元素
x.head<n>()                        // 行向量的前n个元素
x.tail(n)                          // 列向量的倒数n个元素
x.tail<n>()                        // 行向量的倒数n个元素
x.segment(i, n)                    // 行向量从i开始的n个元素
x.segment<n>(i)                    // 列向量从i开始的n个元素
/*************************只能对矩阵操作*************************/
P.block(i, j, rows, cols)          // 从i行j列开始的rows行cols列块。
P.block<rows, cols>(i, j)          // 从i行j列开始的rows行cols列块
P.row(i)                           // 矩阵P的第i行元素
P.col(j)                           // 矩阵P的第j列元素
P.leftCols<cols>()                 // P矩阵左边cols列元素
P.leftCols(cols)                   // P矩阵左边cols列元素
P.middleCols<cols>(j)              // P矩阵第j列开始的cols列元素
P.middleCols(j, cols)              // P矩阵第j列开始的cols列元素
P.rightCols<cols>()                // P矩阵右边cols列元素
P.rightCols(cols)                  // P矩阵右边cols列元素
P.topRows<rows>()                  // P矩阵前rows行元素
P.topRows(rows)                    // P矩阵前rows行元素
P.middleRows<rows>(i)              // P矩阵第i行开始的row行元素
P.middleRows(i, rows)              // P矩阵第i行开始的row行元素
P.bottomRows<rows>()               // P矩阵倒数row行
P.bottomRows(rows)                 // P矩阵倒数row行
P.topLeftCorner(rows, cols)        // P矩阵左上角rows行, cols列元素
P.topRightCorner(rows, cols)       // P矩阵右上角rows行, cols列元素
P.bottomLeftCorner(rows, cols)
P.bottomRightCorner(rows, cols)
P.topLeftCorner<rows,cols>()
P.topRightCorner<rows,cols>()
P.bottomLeftCorner<rows,cols>()
P.bottomRightCorner<rows,cols>()
```

2.8.3 矩阵的基本运算

1. 矩阵乘法

```
// 矩阵*向量       矩阵*矩阵          矩阵数乘
y  = M*x;          R  = P*Q;          R  = P*s;
a  = b*M;          R  = P - Q;        R  = s*P;
a *= M;            R  = P + Q;        R  = P/s;
                   R *= Q;            R  = s*P;
                   R += Q;            R *= s;
                   R -= Q;            R /= s;
```

这里要注意的是，在 Eigen 里你不能混合两种不同类型的矩阵，例如

```
// 错误示范
// Matrix<double, 2, 1> result_wrong_type = matrix_23 * v_3d;
// 原因是 matrix_23 为float型, 而 v_3d 为 double 型
// 应该显式转换
```

```
auto result = matrix_23.cast<double>() * v_3d;
cout << "[1,2,3;4,5,6]*[3;2;1]=" << result.transpose() << endl;

auto result2 = matrix_23 * vd_3d;
cout << "[1,2,3;4,5,6]*[4;5;6]: " << result2.transpose() << endl;
```

同时需要注意矩阵乘法的维度，例如

```
// 错误示范
// Eigen::Matrix<double, 2, 3> result_wrong_dimension = matrix_23.cast<double>() * v_3d;
// 正确写法
Eigen::Matrix<double, 2, 1> result_correct_dimension = matrix_23.cast<double>() * v_3d;
// 可以直接用 auto 类型，但是这样可能会造成后续不知道矩阵维度
auto result_correct_dimension = matrix_23.cast<double>() * v_3d;
```

2. 矩阵的基本运算

```
R.adjoint()                          // R矩阵的伴随矩阵
R.transpose()                        // R矩阵的转置
R.diagonal()                         // R矩阵的迹，用列表示
x.asDiagonal()                       // 对角矩阵
R.reverse()                          // R矩阵逆时针旋转180度(反转)
R.colwise().reverse();               // R矩阵的列反转
R.rowwise().reverse();               // R矩阵的行反转
R.transpose().colwise().reverse(); // R矩阵逆时针旋转90度
R.transpose().rowwise().reverse(); // R矩阵顺时针旋转90度
R.conjugate()                        // conj(R)共轭矩阵
```

2.8.4 矩阵内部元素运算

矩阵内部元素运算（比如两个矩阵对应元素相乘而不是矩阵乘法）主要用到array() 类方法。

```
// 矩阵内部元素运算
R = P.cwiseProduct(Q);      // R = P .* Q对应点乘
R = P.array() * s.array();  // R = P .* s对应点乘
R = P.cwiseQuotient(Q);     // R = P ./ Q对应点除
R = P.array() / Q.array();  // R = P ./ Q对应点除
R = P.array() + s.array();  // R = P + s  对应点加，和矩阵加法结果一致
R = P.array() - s.array();  // R = P - s  对应点减，和矩阵减法结果一致
R.array() += s;             // R = R + s
R.array() -= s;             // R = R - s
R.array() < Q.array();      // R < Q  ，Q矩阵元素比较，会在相应位置置0或1
R.array() <= Q.array();     // R <= Q ，Q矩阵元素比较，会在相应位置置0或1
R.cwiseInverse();           // 1 ./ R  1点除以R
R.array().inverse();        // 1 ./ R  1点除以R
R.array().sin()             // sin(R)
R.array().cos()             // cos(R)
R.array().pow(s)            // R .^ s
R.array().square()          // R .^ 2
R.array().cube()            // R .^ 3
R.cwiseSqrt()               // sqrt(R)
R.array().sqrt()            // sqrt(R)
```

```
R.array().exp()             // exp(R)
R.array().log()             // log(R)
R.cwiseMax(P)               // max(R, P) 对应位置取大元素
R.array().max(P.array())    // max(R, P)
R.cwiseMin(P)               // min(R, P) 对应位置取小元素
R.array().min(P.array())    // min(R, P)
R.cwiseAbs()                // abs(R)
R.array().abs()             // abs(R)
R.cwiseAbs2()               // abs(R.^2)
R.array().abs2()            // abs(R.^2)
```

2.8.5 利用矩阵求解线性方程

```
// 我们求解 matrix_NN * x = v_Nd 这个方程，直接求逆自然是最直接的，但是求逆运算量大

#define MATRIX_SIZE 50
Eigen::Matrix<double, MATRIX_SIZE, MATRIX_SIZE> matrix_NN = Eigen::MatrixXd::Random(MATRIX_SIZE,
     MATRIX_SIZE);
matrix_NN = matrix_NN * matrix_NN.transpose(); // 保证半正定
Eigen::Matrix<double, MATRIX_SIZE, 1> v_Nd = MatrixXd::Random(MATRIX_SIZE, 1);

// 直接求逆
clock_t time_stt = clock(); // 计时
Eigen::MatrixXd x = matrix_NN.inverse() * v_Nd;
cout << "time of normal inverse is "
     << 1000 * (clock() - time_stt) / (double)CLOCKS_PER_SEC << "ms" << endl;
cout << "x = " << x.transpose() << endl;

// 通常用矩阵分解来求，例如QR分解，速度会快很多
time_stt = clock();
x = matrix_NN.colPivHouseholderQr().solve(v_Nd);
cout << "time of Qr decomposition is "
     << 1000 * (clock() - time_stt) / (double)CLOCKS_PER_SEC << "ms" << endl;
cout << "x = " << x.transpose() << endl;

// 对于正定矩阵，还可以用cholesky分解来解方程
time_stt = clock();
x = matrix_NN.ldlt().solve(v_Nd);
cout << "time of ldlt decomposition is "
     << 1000 * (clock() - time_stt) / (double)CLOCKS_PER_SEC << "ms" << endl;
cout << "x = " << x.transpose() << endl;
```

2.8.6 Eigen 几何模块

Eigen 中各个转换矩阵和几何参数的数据定义如下，每种类型都有单精度和双精度两种数据类型（末尾 d：双精度，末尾 f：单精度），而且不能由编译器自动转换。下面以双精度为例。

- 旋转矩阵 (3×3)： Eigen::Matrix3d
- 旋转向量 (3×1)： Eigen::AngleAxisd

- 欧拉角 (3×1): Eigen::Vector3d
- 四元数 (4×1): Eigen:::Quaterniond
- 欧式变换矩阵 (4×4): Eigen::Isometry3d
- 仿射变换矩阵 (4×4): Eigen::Affine3d
- 射影变换矩阵 (4×4): Eigen::Projective3d

第 3 章　参考内容

3.1　旋转四元数参数的化简

旋转四元数原始表达式为

$$\begin{aligned} b &= b_{\parallel} + b_{\perp} \\ &= a_{\parallel} + q a_{\perp} \qquad (\text{其中 } q = \left[\cos\Phi, \sin\Phi \boldsymbol{e}\right]) \end{aligned} \tag{3.1}$$

在进一步化简前，我们需要证明几个引理：

引理 3.1

如果 $q = \left[\cos\Phi, \sin\Phi \boldsymbol{e}\right]$，而且 $\boldsymbol{e}$ 为单位向量，那么 $q^2 = qq = \left[\cos(2\Phi), \sin(2\Phi)\boldsymbol{a}\right]$.

证　这个引理证明只需要用到 Grassmann 积的定义和三角函数的公式。

$$\begin{aligned} q^2 &= \left[\cos\Phi, \sin\Phi \boldsymbol{e}\right] \cdot \left[\cos\Phi, \sin\Phi \boldsymbol{e}\right] \\ &= \left[\cos^2\Phi - (\sin\Phi \boldsymbol{e} \cdot \sin\Phi \boldsymbol{e}), (\cos\Phi \sin\Phi + \sin\Phi \cos\Phi)\boldsymbol{e} + (\sin\Phi \boldsymbol{e} \times \sin\Phi \boldsymbol{e})\right] \\ &= \left[\cos^2\Phi - \sin^2\Phi \|a\|^2, 2\sin\Phi\cos\Phi \boldsymbol{e} + 0\right] \\ &= \left[\cos^2\Phi - \sin^2\Phi, 2\sin\Phi\cos\Phi \boldsymbol{e}\right] \\ &= \left[\cos(2\Phi), \sin(2\Phi)\boldsymbol{e}\right] \end{aligned}$$

□

这个引理的几何意义就是，如果绕着同一个轴 $\boldsymbol{e}$ 连续旋转 Φ 度 2 次，那么所做出的的变换等同于直接绕着 $\boldsymbol{a}$ 旋转 2Φ 度。

引理 3.2

假设 $a_{\parallel} = \left[0, \boldsymbol{a}_{\parallel}\right]$ 是一个纯四元数，而 $q = \left[\alpha, \beta \boldsymbol{e}\right]$，其中 $\boldsymbol{e}$ 是一个单位向量且 $\alpha, \beta \in \mathbb{R}$. 在这种条件下，如果 $\boldsymbol{a}_{\parallel}$ 平行于 $\boldsymbol{e}$，那么 $qa_{\parallel} = a_{\parallel}q$.

证　这个引理的证明同样用到了 Grassmann 积，分别计算等式的左右两边。等式左边：

$$\begin{aligned} \text{LHS} &= q\boldsymbol{a}_{\parallel} \\ &= \left[\alpha, \beta \boldsymbol{e}\right] \cdot \left[0, \boldsymbol{a}_{\parallel}\right] \\ &= \left[0 - \beta \boldsymbol{e} \cdot \boldsymbol{a}_{\parallel}, \alpha \boldsymbol{a}_{\parallel} + \boldsymbol{0} + \beta \boldsymbol{e} \times \boldsymbol{a}_{\parallel}\right] \\ &= \left[-\beta \boldsymbol{e} \cdot \boldsymbol{a}_{\parallel}, \alpha \boldsymbol{a}_{\parallel}\right] \qquad (\boldsymbol{a}_{\parallel}\text{平行于}\boldsymbol{e}\text{，所以}\beta\boldsymbol{e} \times \boldsymbol{a}_{\parallel} = \boldsymbol{0}) \end{aligned}$$

等式右边：

$$
\begin{aligned}
\text{RHS} &= a_{\parallel} q \\
&= \left[0, \boldsymbol{a}_{\parallel}\right] \cdot \left[\alpha, \beta \boldsymbol{e}\right] \\
&= \left[0 - \boldsymbol{a}_{\parallel} \cdot \beta \boldsymbol{e}, \mathbf{0} + \alpha + \alpha \boldsymbol{a}_{\parallel} + \boldsymbol{a}_{\parallel} \times \beta \boldsymbol{e}\right] \\
&= \left[-\boldsymbol{a}_{\parallel} \cdot \beta \boldsymbol{e}, \alpha \boldsymbol{a}_{\parallel}\right] && (\boldsymbol{a}_{\parallel}\text{平行于}\boldsymbol{e}\text{，所以}\beta \boldsymbol{a}_{\parallel} \times \beta \boldsymbol{e} l = \mathbf{0}) \\
&= \left[-\beta \boldsymbol{e} \cdot \boldsymbol{a}_{\parallel}, \alpha \boldsymbol{a}_{\parallel}\right] && (\text{点乘遵守交换律}) \\
&= \text{LHS}
\end{aligned}
$$

□

引理 3.3

假设 $a_{\perp} = \left[0, \boldsymbol{a}_{\perp}\right]$ 是一个纯四元数，而 $q = \left[\alpha, \beta \boldsymbol{e}\right]$，其中 $\boldsymbol{e}$ 是一个单位向量且 $\alpha, \beta \in \mathbb{R}$. 在这种条件下，如果 $\boldsymbol{a}_{\parallel}$ 正交于 $\boldsymbol{e}$，那么 $q a_{\parallel} = a_{\parallel} q^{*}$.

证 这个引理的证明和 **引理 3.2** 类似。

$$
\begin{aligned}
\text{LHS} &= e a_{\perp} \\
&= \left[\alpha, \beta \boldsymbol{e}\right] \cdot \left[0, \boldsymbol{a}_{\perp}\right] \\
&= \left[0 - \beta \boldsymbol{e} \cdot \boldsymbol{a}_{\perp}, \alpha \boldsymbol{a}_{\perp} + \mathbf{0} + \beta \boldsymbol{e} \times \boldsymbol{a}_{\perp}\right] \\
&= \left[0, \alpha \boldsymbol{a}_{\perp} + \beta \boldsymbol{e} \times \boldsymbol{a}_{\perp}\right] && (\boldsymbol{a}_{\perp}\text{正交于}\boldsymbol{e}\text{，所以}\beta \boldsymbol{e} \times \boldsymbol{a}_{\perp} = \mathbf{0})
\end{aligned}
$$

$$
\begin{aligned}
\text{RHS} &= a_{\perp} q^{*} \\
&= \left[0, \boldsymbol{a}_{\perp}\right] \cdot \left[\alpha, -\beta \boldsymbol{e}\right] \\
&= \left[0 + \boldsymbol{a}_{\perp} \cdot \beta \boldsymbol{e}, \mathbf{0} + \alpha \boldsymbol{a}_{\perp} + \boldsymbol{a}_{\perp} \times (-\beta \boldsymbol{e})\right] \\
&= \left[0, \alpha \boldsymbol{a}_{\perp} + \boldsymbol{a}_{\perp} \times (-\beta \boldsymbol{e})\right] && (\boldsymbol{a}_{\perp}\text{正交于}\boldsymbol{e}\text{，所以}\beta \boldsymbol{e} \times \boldsymbol{a}_{\perp} = \mathbf{0}) \\
&= \left[0, \alpha \boldsymbol{a}_{\perp} - (-\beta \boldsymbol{e}) \times \boldsymbol{a}_{\perp}\right] && (\boldsymbol{a} \times \boldsymbol{b} = -\boldsymbol{b} \times \boldsymbol{a}) \\
&= \left[0, \alpha \boldsymbol{a}_{\perp} + \beta \boldsymbol{e} \times \boldsymbol{a}_{\perp}\right] \\
&= \text{LHS}
\end{aligned}
$$

□

现在，我们利用引理对公式进行变形。首先，利用四元数逆的定义 $q q^{-1} = 1$，可得

$$
\begin{aligned}
b &= a_{\parallel} + q a_{\perp} \\
&= 1 \cdot a_{\parallel} + q a_{\perp} \\
&= p p^{-1} a_{\parallel} + p p a_{\perp}
\end{aligned} \tag{3.2}
$$

其中，引入了新的四元数 p 且

$$
\begin{aligned}
q = p^2 &= \left[\cos \Phi, \sin \Phi \boldsymbol{e}\right] \\
p &= \left[\cos \frac{\Phi}{2}, \sin \frac{\Phi}{2} \boldsymbol{e}\right]
\end{aligned}
$$

根据 **引理 3.1** 可以验证其正确性：

$$\begin{aligned} pp &= p^2 \\ &= \left[\cos\left(2\cdot\frac{\Phi}{2}\right), \sin\left(2\cdot\frac{\Phi}{2}\right)\boldsymbol{e}\right] \\ &= \left[\cos\Phi, \sin\Phi\boldsymbol{e}\right] = q \end{aligned}$$

同样的，容易得到 p 也是一个单位四元数，即

$$p^{-1} = p^*$$

那么

$$\begin{aligned} b &= pp^{-1}a_{\parallel} + ppa_{\perp} \\ &= pp^*a_{\parallel} + ppa_{\perp} \end{aligned}$$

结合 **引理 3.2** 和 **引理 3.3** ，对公式再次变形

$$\begin{aligned} b &= pp^*a_{\parallel} + ppa_{\perp} \\ &= pa_{\parallel}p^* + pa_{\perp}p^* \qquad (3.3) \\ &= p(a_{\parallel} + a_{\perp})p^* \qquad (3.4) \end{aligned}$$

由因为 $a_{\parallel}, a_{\perp}$ 是 a 的分量，所以 $a_{\parallel} + a_{\perp} = a$，即

$$b = pap^* \qquad (3.5)$$

其中，

$$p = \left[\cos\frac{\Phi}{2}, \sin\frac{\Phi}{2}\boldsymbol{e}\right] \qquad (3.6)$$

3.2 旋转的欧拉轴 / 角参数表达和四元数表达的等价性

若定义这些旋转参数和向量为四元数

$$a = [0, \boldsymbol{a}] \qquad b = [0, \boldsymbol{b}]$$

$$e = [0, \boldsymbol{e}] \qquad q = \left[\cos\frac{\Phi}{2}, \sin\frac{\Phi}{2}\boldsymbol{e}\right]$$

其中，$\boldsymbol{e}$ 是旋转轴单位向量，Φ 是旋转的角度。则

$$b = qaq^* = \boldsymbol{b} = \cos\Phi\boldsymbol{a} + (1-\cos\Phi)(\boldsymbol{e}\cdot\boldsymbol{a})\boldsymbol{e} + \sin\Phi(\boldsymbol{e}\times\boldsymbol{a}) \qquad (3.7)$$

证　首先，引入一个引理，即**向量二重叉乘公式**.

引理 3.4　向量二重叉乘公式

向量 $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$ 满足

$$(\boldsymbol{a}\times\boldsymbol{b})\times\boldsymbol{c} = (\boldsymbol{c}\cdot\boldsymbol{a})\boldsymbol{b} - (\boldsymbol{c}\cdot\boldsymbol{b})\boldsymbol{a} \qquad (3.8)$$

$$\boldsymbol{a}\times(\boldsymbol{b}\times\boldsymbol{c}) = (\boldsymbol{a}\cdot\boldsymbol{c})\boldsymbol{b} - (\boldsymbol{a}\cdot\boldsymbol{b})\boldsymbol{c} \qquad (3.9)$$

这个引理的证明略。

下面我们开始推导旋转的欧拉轴 / 角参数表达和四元数表达的等价性，由 Grassmann 积，

$$
\begin{aligned}
b &= qaq^* \\
&= \left[\cos\frac{\Phi}{2}, \sin\frac{\Phi}{2}\boldsymbol{e}\right] \cdot [0, \boldsymbol{a}] \cdot \left[\cos\frac{\Phi}{2}, -\sin\frac{\Phi}{2}\boldsymbol{e}\right] \\
&= \left[0 - \sin\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a}, \cos\frac{\Phi}{2}\boldsymbol{a} + \boldsymbol{0} + \sin\frac{\Phi}{2}\boldsymbol{e}\times\boldsymbol{a}\right] \cdot \left[\cos\frac{\Phi}{2}, -\sin\frac{\Phi}{2}\boldsymbol{e}\right] \\
&= \left[-\sin\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a}, \cos\frac{\Phi}{2}\boldsymbol{a} + \sin\frac{\Phi}{2}\boldsymbol{e}\times\boldsymbol{a}\right] \cdot \left[\cos\frac{\Phi}{2}, -\sin\frac{\Phi}{2}\boldsymbol{e}\right] \\
&= \left[-\sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a} - \left(-\sin\frac{\Phi}{2}\boldsymbol{e}\cdot\left(\cos\frac{\Phi}{2}\boldsymbol{a} + \sin\frac{\Phi}{2}\boldsymbol{e}\times\boldsymbol{a}\right)\right),\right. \\
&\quad \left. -\sin\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a}\cdot\left(-\sin\frac{\Phi}{2}\boldsymbol{e}\right) + \cos\frac{\Phi}{2}\left(\cos\frac{\Phi}{2}\boldsymbol{a} + \sin\frac{\Phi}{2}\boldsymbol{e}\times\boldsymbol{a}\right) + \left(\cos\frac{\Phi}{2}\boldsymbol{a} + \sin\frac{\Phi}{2}\boldsymbol{e}\times\boldsymbol{a}\right)\times\left(-\sin\frac{\Phi}{2}\boldsymbol{e}\right)\right] \\
&= \left[-\sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a} + \sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a} + \sin^2\frac{\Phi}{2}\boldsymbol{e}\cdot(\boldsymbol{e}\times\boldsymbol{a}),\right. \\
&\quad \left. \sin^2\frac{\Phi}{2}(\boldsymbol{e}\cdot\boldsymbol{a})\cdot\boldsymbol{e} + \cos^2\frac{\Phi}{2}\boldsymbol{a} + \sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\times\boldsymbol{a} - \sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{a}\times\boldsymbol{e} - \sin^2\frac{\Phi}{2}(\boldsymbol{e}\times\boldsymbol{a})\times\boldsymbol{e}\right] \\
&= \left[-\sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a} + \sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a} + \sin^2\frac{\Phi}{2}\boldsymbol{e}\cdot(\boldsymbol{e}\times\boldsymbol{a}),\right. \\
&\quad \left. \sin^2\frac{\Phi}{2}(\boldsymbol{e}\cdot\boldsymbol{a})\cdot\boldsymbol{e} + \cos^2\frac{\Phi}{2}\boldsymbol{a} + 2\sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\times\boldsymbol{a} - \sin^2\frac{\Phi}{2}(\boldsymbol{e}\times\boldsymbol{a})\times\boldsymbol{e}\right]
\end{aligned}
$$

由于 $\boldsymbol{e}\times\boldsymbol{a}$ 正交于 $\boldsymbol{e}$，即

$$\boldsymbol{e}\cdot(\boldsymbol{e}\times\boldsymbol{a}) = 0$$

进一步化简得

$$
\begin{aligned}
b &= \left[-\sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a} + \sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a} + \sin^2\frac{\Phi}{2}\boldsymbol{e}\cdot(\boldsymbol{e}\times\boldsymbol{a}),\right. \\
&\quad \left. \sin^2\frac{\Phi}{2}(\boldsymbol{e}\cdot\boldsymbol{a})\cdot\boldsymbol{e} + \cos^2\frac{\Phi}{2}\boldsymbol{a} + 2\sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\times\boldsymbol{a} - \sin^2\frac{\Phi}{2}(\boldsymbol{e}\times\boldsymbol{a})\times\boldsymbol{e}\right] \\
&= \left[-\sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a} + \frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a}, \sin^2\frac{\Phi}{2}(\boldsymbol{e}\cdot\boldsymbol{a})\cdot\boldsymbol{e} + \cos^2\frac{\Phi}{2}\boldsymbol{a} + 2\sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\times\boldsymbol{a} - \sin^2\frac{\Phi}{2}(\boldsymbol{e}\times\boldsymbol{a})\times\boldsymbol{e}\right] \\
&= \left[0, \sin^2\frac{\Phi}{2}(\boldsymbol{e}\cdot\boldsymbol{a})\cdot\boldsymbol{e} + \cos^2\frac{\Phi}{2}\boldsymbol{a} + 2\sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\times\boldsymbol{a} - \sin^2\frac{\Phi}{2}(\boldsymbol{e}\times\boldsymbol{a})\times\boldsymbol{e}\right]
\end{aligned}
$$

利用倍角公式化简和 引理 **3.4** 进一步化简得

$$
\begin{aligned}
b &= \left[0, \sin^2\frac{\Phi}{2}(\boldsymbol{e}\cdot\boldsymbol{a})\cdot\boldsymbol{e} + \cos^2\frac{\Phi}{2}\boldsymbol{a} + 2\sin\frac{\Phi}{2}\cos\frac{\Phi}{2}\boldsymbol{e}\times\boldsymbol{a} - \sin^2\frac{\Phi}{2}(\boldsymbol{e}\times\boldsymbol{a})\times\boldsymbol{e}\right] \\
&= \left[0, \frac{1-\cos\Phi}{2}\boldsymbol{e}\cdot\boldsymbol{a}\cdot\boldsymbol{e} + \frac{1+\cos\Phi}{2}\boldsymbol{a} + \sin\Phi(\boldsymbol{e}\times\boldsymbol{a}) - \frac{1-\cos\Phi}{2}(\boldsymbol{e}\times\boldsymbol{a})\times\boldsymbol{e}\right] \\
&= \left[0, \frac{1-\cos\Phi}{2}(\boldsymbol{e}\cdot\boldsymbol{a})\cdot\boldsymbol{e} + \frac{1+\cos\Phi}{2}\boldsymbol{a} + \sin\Phi(\boldsymbol{e}\times\boldsymbol{a}) - \frac{1-\cos\Phi}{2}\big((\boldsymbol{e}\cdot\boldsymbol{e})\boldsymbol{a} - (\boldsymbol{e}\cdot\boldsymbol{a})\boldsymbol{e}\big)\right] \\
&= \left[0, \frac{1-\cos\Phi}{2}(\boldsymbol{e}\cdot\boldsymbol{a})\cdot\boldsymbol{e} + \frac{1+\cos\Phi}{2}\boldsymbol{a} + \sin\Phi(\boldsymbol{e}\times\boldsymbol{a}) - \frac{1-\cos\Phi}{2}\big(\boldsymbol{a} - (\boldsymbol{e}\cdot\boldsymbol{a})\boldsymbol{e}\big)\right] \\
&= \left[0, (1-\cos\Phi)(\boldsymbol{e}\cdot\boldsymbol{a})\cdot\boldsymbol{e} + \cos\Phi\boldsymbol{a} + \sin\Phi(\boldsymbol{e}\times\boldsymbol{a})\right] = \boldsymbol{b}
\end{aligned}
$$

□

3.3 向量旋转矩阵和坐标系旋转矩阵的关系

假设向量 $\boldsymbol{a}$ 在坐标系 S_a, S_b 下的分量为

$$\boldsymbol{a} = \boldsymbol{u}\boldsymbol{e}_a = \boldsymbol{v}\boldsymbol{e}_b$$

为了方便计算，这里向量均表示为分量阵列，即 $\boldsymbol{u}$ 和 $\boldsymbol{v}$ 都是 3×3 的对角矩阵

$$\boldsymbol{a} = \begin{bmatrix} u_x & 0 & 0 \\ 0 & u_y & 0 \\ 0 & 0 & u_z \end{bmatrix} \begin{bmatrix} \boldsymbol{i}_a \\ \boldsymbol{j}_a \\ \boldsymbol{k}_a \end{bmatrix} = \begin{bmatrix} v_x & 0 & 0 \\ 0 & v_y & 0 \\ 0 & 0 & v_z \end{bmatrix} \begin{bmatrix} \boldsymbol{i}_b \\ \boldsymbol{j}_b \\ \boldsymbol{k}_b \end{bmatrix}$$

考虑向量 $\boldsymbol{a}$ 在绕轴 $\boldsymbol{e}$ 旋转 Φ 后得到的向量 $\boldsymbol{b}$ 在坐标系 S_b 的分量为 $\boldsymbol{u}$，即

$$\boldsymbol{b} = \begin{bmatrix} u_x & 0 & 0 \\ 0 & u_y & 0 \\ 0 & 0 & u_z \end{bmatrix} \begin{bmatrix} \boldsymbol{i}_a \\ \boldsymbol{j}_a \\ \boldsymbol{k}_a \end{bmatrix} = \boldsymbol{u}\boldsymbol{e}_b$$

且其中的向量旋转矩阵记为 $\boldsymbol{R}_{ba}$，坐标系旋转矩阵记为 $\boldsymbol{C}_{ba}$，那么

$$\begin{cases} \boldsymbol{b} = \boldsymbol{u}\boldsymbol{e}_b \\ \boldsymbol{b} = \boldsymbol{R}_{ba}\boldsymbol{a} = \boldsymbol{R}_{ba}\boldsymbol{u}\boldsymbol{e}_a \end{cases} \xrightarrow{\text{消去 } \boldsymbol{b}} \boldsymbol{u}\boldsymbol{e}_b = \boldsymbol{R}_{ba}\boldsymbol{u}\boldsymbol{e}_a \xrightarrow{\boldsymbol{e}_b = \boldsymbol{C}_{ba}\boldsymbol{e}_a} \boldsymbol{u}\boldsymbol{C}_{ba}\boldsymbol{e}_a = \boldsymbol{R}_{ba}\boldsymbol{u}\boldsymbol{e}_a \tag{3.10}$$

等式两边右乘 $(\boldsymbol{e}_a)^{-1}$，消去 $\boldsymbol{e}_a$，可得

$$\boldsymbol{u}\boldsymbol{C}_{ba} = \boldsymbol{R}_{ba}\boldsymbol{u} \tag{3.11}$$

3.4 欧拉轴角向量恒等式的证明

1. $[\ \boldsymbol{e}^{\mathrm{T}}\boldsymbol{e} = 1]$

证　由于欧拉轴向量 $\boldsymbol{e}$ 是单位向量，所以

$$\boldsymbol{e}\cdot\boldsymbol{e} = \boldsymbol{e}^{\mathrm{T}}\boldsymbol{e} = 1 \tag{3.12}$$

□

2. $[\ \boldsymbol{e}^{\times}\boldsymbol{e} = 0]$

证　由于平行向量叉乘为 0，所以

$$\boldsymbol{e}\times\boldsymbol{e} = \boldsymbol{e}^{\times}\boldsymbol{e} = 0 \tag{3.13}$$

□

3. $[\ \dot{\boldsymbol{e}}^{\times}\boldsymbol{e} = 0]$

证　对公式 (3.13) 两边同时求导，有

$$\left(\boldsymbol{e}^{\times}\boldsymbol{e}\right)' = 0 \quad \Rightarrow \quad \dot{\boldsymbol{e}}^{\times}\boldsymbol{e} + \boldsymbol{e}^{\times}\dot{\boldsymbol{e}} = 0 \quad \Rightarrow \quad \dot{\boldsymbol{e}}^{\times}\boldsymbol{e} = -\boldsymbol{e}^{\times}\dot{\boldsymbol{e}} \tag{3.14}$$

□

4. [$\dot{\boldsymbol{e}}^{\times}\boldsymbol{e}^{\times} = \boldsymbol{e}\dot{\boldsymbol{e}}^{\mathrm{T}}$]

证 设 $\boldsymbol{e} = \begin{bmatrix} e_x & e_y & e_z \end{bmatrix}^{\mathrm{T}}$，则

$$\dot{\boldsymbol{e}} = \begin{bmatrix} \dot{e}_x \\ \dot{e}_y \\ \dot{e}_z \end{bmatrix}, \qquad \dot{\boldsymbol{e}}^{\times} = \begin{bmatrix} 0 & -\dot{e}_x & \dot{e}_y \\ \dot{e}_z & 0 & -\dot{e}_x \\ -\dot{e}_y & \dot{e}_x & 0 \end{bmatrix}$$

注意：$\dot{\boldsymbol{e}}^{\times}$ 同样是反对称矩阵，即 $\left(\dot{\boldsymbol{e}}^{\times}\right)^{\mathrm{T}} = -\dot{\boldsymbol{e}}^{\times}$，所以有

$$\dot{\boldsymbol{e}}^{\times}\boldsymbol{e}^{\times} = \begin{bmatrix} 0 & -\dot{e}_z & \dot{e}_y \\ \dot{e}_z & 0 & -\dot{e}_x \\ -\dot{e}_y & \dot{e}_x & 0 \end{bmatrix} \begin{bmatrix} 0 & e_z & e_y \\ e_z & 0 & -e_x \\ e_y & e_x & 0 \end{bmatrix} = \begin{bmatrix} -\dot{e}_y e_y - \dot{e}_z e_z & \dot{e}_y e_x & \dot{e}_z e_x \\ \dot{e}_x e_y & -\dot{e}_x x - \dot{e}_z e_z & \dot{e}_z e_y \\ \dot{e}_x e_z & \dot{e}_y e_z & -\dot{e}_x x - \dot{e}_y y \end{bmatrix} \tag{3.15}$$

由于向量的导数与向量正交（$\boldsymbol{u} = \boldsymbol{\omega} \times \boldsymbol{u}$），所以

$$\dot{\boldsymbol{e}} \cdot \boldsymbol{e} = \dot{\boldsymbol{e}}^{\mathrm{T}}\boldsymbol{e} = \begin{bmatrix} \dot{e}_x & \dot{e}_y & \dot{e}_z \end{bmatrix} \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} = \dot{e}_x x + \dot{e}_y e_y + \dot{e}_z e_z = 0 \tag{3.16}$$

替换掉公式 (3.15) 中的对角线元素，可以得到

$$\dot{\boldsymbol{e}}^{\times}\boldsymbol{e}^{\times} = \begin{bmatrix} -\dot{e}_y e_y - \dot{e}_z e_z & \dot{e}_y e_x & \dot{e}_z e_x \\ \dot{e}_x e_y & -\dot{e}_x x - \dot{e}_z e_z & \dot{e}_z e_y \\ \dot{e}_x e_z & \dot{e}_y e_z & -\dot{e}_x x - \dot{e}_y y \end{bmatrix} = \begin{bmatrix} \dot{e}_x e_x & \dot{e}_y e_x & \dot{e}_z e_x \\ \dot{e}_x e_y & \dot{e}_y e_y & \dot{e}_z e_y \\ \dot{e}_x e_z & \dot{e}_y e_z & \dot{e}_z z \end{bmatrix} = \dot{\boldsymbol{e}}\boldsymbol{e}^{\mathrm{T}} \tag{3.17}$$

□

第 4 章　部分示例代码

4.1 Eigen

4.1.1 Eigen 的基本使用

```cpp
#include <iostream>

using namespace std;

// Eigen 核心部分
#include <Eigen/Core>
// 稠密矩阵的代数运算（逆，特征值等）
#include <Eigen/Dense>

using namespace Eigen;

#define MATRIX_SIZE 50

void solve_eqn(MatrixXd mtr, MatrixXd vt);

int main(int argc, char **argv)
{
    // Eigen 中所有向量和矩阵都是Eigen::Matrix，它是一个模板类。前三个参数为：数据类型，行，列
    // 声明一个2*3的float矩阵
    Matrix<float, 2, 3> matrix_23;

    // 同时，Eigen 通过 typedef 提供了许多内置类型，不过底层仍是Eigen::Matrix
    // 例如 Vector3d 实质上是 Eigen::Matrix<double, 3, 1>，即三维向量
    Vector3d v_3d;
    // 这是一样的
    Matrix<float, 3, 1> vd_3d;

    // Matrix3d 实质上是 Eigen::Matrix<double, 3, 3>
    Matrix3d matrix_33 = Matrix3d::Zero(); //初始化为零
    // 如果不确定矩阵大小，可以使用动态大小的矩阵
    Matrix<double, Dynamic, Dynamic> matrix_dynamic;
    // 更简单的
    MatrixXd matrix_x;
    // 这种类型还有很多，我们不一一列举

```

```cpp
    // 下面是对Eigen阵的操作
    // 输入数据
    matrix_23 << 1, 2, 3, 4, 5, 6;
    // 输出
    cout << "matrix 2x3 from 1 to 6: \n"
         << matrix_23 << endl;

    // 用()访问矩阵中的元素，循环遍历输出各个元素
    cout << "print matrix 2x3: " << endl;
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 3; j++)
            cout << matrix_23(i, j) << "\t";
        cout << endl;
    }

    // 矩阵和向量相乘（实际上仍是矩阵和矩阵）
    v_3d << 3, 2, 1;
    vd_3d << 4, 5, 6;

    // 但是在Eigen里你不能混合两种不同类型的矩阵，像这样是错的
    // Matrix<double, 2, 1> result_wrong_type = matrix_23 * v_3d;
    // 原因是 matrix_23 为float型，而 v_3d 为 double 型
    // 应该显式转换
    auto result = matrix_23.cast<double>() * v_3d;
    cout << "[1,2,3;4,5,6]*[3;2;1]=" << result.transpose() << endl;

    auto result2 = matrix_23 * vd_3d;
    cout << "[1,2,3;4,5,6]*[4;5;6]: " << result2.transpose() << endl;

    // 同样你不能搞错矩阵的维度
    // 如果不想知道矩阵的大小，矩阵乘法的结果可以直接用 auto 类型

    // 一些矩阵运算
    // 四则运算就不演示了，直接用+-*/即可。
    matrix_33 = Matrix3d::Random(); // 随机数矩阵
    cout << "random matrix: \n"
         << matrix_33 << endl;
    cout << "transpose: \n"
         << matrix_33.transpose() << endl;          // 转置
    cout << "sum: " << matrix_33.sum() << endl;     // 各元素和
    cout << "trace: " << matrix_33.trace() << endl; // 迹
    cout << "times 10: \n"
         << 10 * matrix_33 << endl; // 数乘
    cout << "inverse: \n"
         << matrix_33.inverse() << endl;                // 逆
    cout << "det: " << matrix_33.determinant() << endl; // 行列式

    // 求解特征值
    // 实对称矩阵可以保证对角化成功
    SelfAdjointEigenSolver<Matrix3d> eigen_solver(matrix_33.transpose() * matrix_33);
```

```
87      cout << "Eigen values = \n"
88           << eigen_solver.eigenvalues() << endl; // 特征值
89      cout << "Eigen vectors = \n"
90           << eigen_solver.eigenvectors() << endl; // 特征向量
91
92      // 解方程
93      // 我们求解 matrix_NN * x = v_Nd 这个方程
94      // N的大小在前边的宏里定义，它由随机数生成
95      // 直接求逆自然是最直接的，但是求逆运算量大
96
97      Matrix<double, MATRIX_SIZE, MATRIX_SIZE> matrix_NN = MatrixXd::Random(MATRIX_SIZE,
            MATRIX_SIZE);
98      matrix_NN = matrix_NN * matrix_NN.transpose(); // 保证半正定
99      Matrix<double, MATRIX_SIZE, 1> v_Nd = MatrixXd::Random(MATRIX_SIZE, 1);
100
101     solve_eqn(matrix_NN, v_Nd);
102
103     return 0;
104 }
105
106 void solve_eqn(MatrixXd mtr, MatrixXd vt)
107 {
108     // 直接求逆
109     MatrixXd x = mtr.inverse() * vt;
110     cout << "time of normal inverse is "
111          << 1000 * (clock() - time_stt) / (double)CLOCKS_PER_SEC << "ms" << endl;
112     cout << "x = " << x.transpose() << endl;
113
114     // 通常用矩阵分解来求，例如QR分解，速度会快很多
115     time_stt = clock();
116     x = mtr.colPivHouseholderQr().solve(vt);
117     cout << "time of Qr decomposition is "
118          << 1000 * (clock() - time_stt) / (double)CLOCKS_PER_SEC << "ms" << endl;
119     cout << "x = " << x.transpose() << endl;
120
121     // 对于正定矩阵，还可以用cholesky分解来解方程
122     time_stt = clock();
123     x = mtr.ldlt().solve(vt);
124     cout << "time of ldlt decomposition is "
125          << 1000 * (clock() - time_stt) / (double)CLOCKS_PER_SEC << "ms" << endl;
126     cout << "x = " << x.transpose() << endl;
127 }
```

插图目录

表格目录

索引

Q

R

S

T

V

X

Y

Z