深蓝学院
shenlanxueyuan.com

第三章李群与李代数 作业分享

主讲人 刘苗

纲要

一：群的性质

深蓝学院
shenlanxueyuan.com

- 群的性质
  - 封闭性，$\forall a_1, a_2 \in \mathrm{A}, \quad a_1 \cdot a_2 \in A$
  - 结合律，$\forall a_1, a_2, a_3 \in \mathrm{A}, \quad (a_1 \cdot a_2) \cdot a_3 = a_1 \cdot (a_2 \cdot a_3)$
  - 幺元：$\exists a_0 \in A, \quad s.t. \quad \forall a \in A, a_0 \cdot a = a \cdot a_0 = a$
  - 逆：$\forall a \in A, \exists a^{-1} \in A, \quad s.t. \quad a \cdot a^{-1} = a_0$

- {Z,+}

- 如果有整数 a_1, a_2，那么 $a_1 + a_2$ 也是整数
- 如果有整数 $a_1, a2, a_3$，那么 $a_1 + a_2$ 是整数，此整数加 a_3 也是整数，$a_2 + a_3$ 是整数，a_1 加此整数也是整数
- 对于整数集，有 a，使得 $a + 0 = 0 + a = a$
- 对于整数集的 a，必存在 $-a$，使得 $a + (-a) = 0$

- {N,+}

- 如果有自然数 a_1, a_2，那么 $a_1 + a_2$ 也是自然数数
- 如果有自然数 $a_1, a2, a_3$，那么 $a_1 + a_2$ 是自然数，此自然数加 a_3 也是自然数，$a_2 + a_3$ 是自然数，a_1 加此自然数也是自然数
- 对于自然数集，有 a，使得 $a + 0 = 0 + a = a$
- 对于自然数集的 a，不存在负数 $-a$，因此无法满足逆条件

二：验证向量叉乘的李代数性质

● 李代数性质

- 封闭性 $\forall X, Y \in V, [X, Y] \in \mathrm{V}$
- 双线性 $\forall X, Y, Z \in V, a, b \in F$, 有： $[aX + bY, Z] = a[X, Z] + b[Y, Z], [Z, aX + bY] = a[Z, X] + b[Z, Y]$
- 自反性 $\forall X \in \mathrm{V}, [X, X] = 0$
- 雅可比等价 $\forall X, Y, Z \in \mathrm{V}, [X, [Y, Z]] + [Y, [Z, X]] + [Z, [X, Y]] = 0$

● 向量叉乘 $(\mathrm{R}^3, \mathrm{R}, \times)$

- 封闭性：
如果有三维向量 v_1, v_2 ，那么 $v_1 \times v_2$ 也是三维向量

$$v_1 \times v_2 = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \end{vmatrix}$$
$$= (y_1 z_2 - z_1 y_2)\vec{i} + (z_1 x_2 - x_1 z_2)\vec{j} + (x_1 y_2 - y_1 x_2)\vec{k}$$

- 双线性
如果有三维向量 v_1, v_2, v_3 ， 有实数 a, b

$$(av_1 + bv_2) \times v_3 = \begin{bmatrix} 0 & -(az_1 + bz_2) & ay_1 + by_2 \\ az_1 + bz_2 & 0 & -(ax_1 + bx_2) \\ -(ay_1 + by_2) & ax_1 + bx_2 & 0 \end{bmatrix}$$
$$= (av_1^{\wedge} + bv_2^{\wedge}) \cdot v_3 = av_1^{\wedge} \cdot v_3 + bv_2^{\wedge} \cdot v_3$$
$$= av_1 \times v_3 + bv_2 \times v_3$$

以及

$$v_3 \times (av_1 + bv_2) = v_3^{\wedge} \cdot (av_1 + bv_2) = av_3^{\wedge} \cdot v_1 + bv_3^{\wedge} \cdot v_2$$
$$= av_3 \times v_1 + bv_3 \times v_2$$

二：验证向量叉乘的李代数性质

深蓝学院 shenlanxueyuan.com

●李代数性质

- 封闭性 $\forall X, Y \in V, [X, Y] \in \mathrm{V}$
- 双线性 $\forall X, Y, Z \in V, a, b \in F$, 有: $[aX + bY, Z] = a[X, Z] + b[Y, Z], [Z, aX + bY] = a[Z, X] + b[Z, Y]$
- 自反性 $\forall X \in \mathrm{V}, [X, X] = 0$
- 雅可比等价 $\forall X, Y, Z \in \mathrm{V}, [X, [Y, Z]] + [Y, [Z, X]] + [Z, [X, Y]] = 0$

●向量叉乘 $(\mathrm{R}^3, \mathrm{R}, \times)$

- 自反性

$$v_1 \times v_1 = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

- 雅克比等价
  由于
  - $a \times (b \times c) = (a \cdot c) \cdot b - (b \cdot c) \cdot a$
  - $a \cdot b = b \cdot a$

  所以

$$\begin{aligned} & v_1 \times (v_2 \times v_3) + v_2 \times (v_3 \times v_1) + v_3 \times (v_1 \times v_2) \\ & = (v_3 \cdot v_1) \cdot v_2 - (v_2 \cdot v_1) \cdot v_3 + (v_2 \cdot v_3) \cdot v_1 \\ & \quad - (v_1 \cdot v_3) \cdot v_2 + (v_1 \cdot v_2) \cdot v_3 - (v_3 \cdot v_2) \cdot v_1 \\ & = 0 \end{aligned}$$

三：推导SE(3)的指数映射

● se(3)的推导

欧式变换：

$$g(t) = \begin{bmatrix} R(t) & T(t) \\ 0 & 1 \end{bmatrix}$$

求导及求逆：

$$\dot{g(t)} = \begin{bmatrix} \dot{R(t)} & \dot{T(t)} \\ 0 & 1 \end{bmatrix}$$

$$g(t)^{-1} = \begin{bmatrix} R(t)^{-1} & -R(t)^{-1}T(t) \\ 0 & 1 \end{bmatrix}$$

推导：

$$\dot{g(t)}g(t)^{-1} = \begin{bmatrix} \dot{R(t)}R(t)^{-1} & \dot{R(t)}R(t)^{-1}T(t) + \dot{T(t)} \\ 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} \phi^\wedge & -\phi^\wedge T(t) + \dot{T(t)} \\ 0 & 0 \end{bmatrix}$$

$$\dot{g(t)} = \begin{bmatrix} \phi^\wedge & -\phi^\wedge T(t) + \dot{T(t)} \\ 0 & 0 \end{bmatrix} g(t)$$

$$= \begin{bmatrix} \phi^\wedge & \rho \\ 0 & 0 \end{bmatrix} g(t) = \xi^\wedge g(t)$$

三：推导SE(3)的指数映射

深蓝学院 shenlanxueyuan.com

●指数映射的推导

对指数式泰勒展开：

$$\exp(\xi^\wedge) = \sum_{n=0}^{\infty} \frac{1}{n!}(\xi^\wedge)^n$$

性质：

$$\xi^\wedge = \begin{bmatrix} \phi^\wedge & \rho \\ 0 & 0 \end{bmatrix}$$

$$(\xi^\wedge)^n = \begin{bmatrix} (\phi^\wedge)^n & (\phi^\wedge)^{n-1}\rho \\ 0 & 0 \end{bmatrix}, n \geq 1$$

$$\exp(\xi^\wedge) = \sum_{n=0}^{\infty} \frac{1}{n!}(\xi^\wedge)^n$$

$$= \begin{bmatrix} \sum_{n=0}^{\infty} \frac{1}{n!}(\phi^\wedge)^n & \sum_{n=0}^{\infty} \frac{1}{(n+1)!}(\phi^\wedge)^n \rho \\ 0 & 0 \end{bmatrix}$$

●左雅克比推导

$$\sum_{n=0}^{\infty} \frac{1}{(n+1)!}(\phi^\wedge)^n$$

$$= I + \frac{1}{2!}\theta a^\wedge + \frac{1}{3!}\theta^2 (a^\wedge)^2 - \frac{1}{4!}\theta^3 a^\wedge - \frac{1}{5!}\theta^3 (a^\wedge)^2 + \cdots$$

$$= I + [\frac{\theta}{2!} - \frac{\theta^3}{4!} + \frac{\theta^5}{6!} - \cdots]a^\wedge + [\frac{\theta^2}{3!} - \frac{\theta^4}{5!} + \frac{\theta^6}{7!} - \cdots](a^\wedge)^2$$

$$= I - [1 - \frac{\theta^2}{2!} + \frac{\theta^4}{4!} - \frac{\theta^6}{6!} + \cdots - 1]\frac{a^\wedge}{\theta}$$

$$- [\theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \frac{\theta^7}{7!} + \cdots - \theta]\frac{(a^\wedge)^2}{\theta}$$

$$= I - (\cos\theta - 1)\frac{a^\wedge}{\theta} - (\sin\theta - \theta)\frac{(a^\wedge)^2}{\theta}$$

$$= I - (\cos\theta - 1)\frac{a^\wedge}{\theta} - (\sin\theta - \theta)\frac{a \cdot a^T - I}{\theta}$$

$$= (1 - \frac{\sin\theta}{\theta})a \cdot a^T + \frac{\sin\theta}{\theta}I + \frac{(1-\cos\theta)}{\theta}a^\wedge$$

四：伴随

●SO(3)伴随

对于李群 $SO(3)$，李代数 $so(3)$，伴随的性质定义为

$$p^\wedge \in so(3), R \in SO(3)$$

$$R \cdot \exp(p^\wedge) = \exp((Adj_R \cdot p)^\wedge) \cdot R$$

证明 $\forall a \in \mathrm{R}^3, \mathrm{R}a^\wedge \mathrm{R}^T = (\mathrm{R}a)^\wedge$:

- 向量 $\vec{v}, \vec{u}$， $u^\wedge v = u \times v$
- 对于任何旋转矩阵 R，有 $(Ra) \times (Rb) = R(a \times b)$
- 如果有旋转矩阵 R

$$\begin{aligned}(R \cdot a)^\wedge v &= (R \cdot a) \times v \\ &= (R \cdot a) \times (RR^{-1} v) \\ &= R[a \times (R^{-1} v)] = Ra^\wedge R^T v\end{aligned}$$

$$(R \cdot a)^\wedge = Ra^\wedge R^T$$

证明：

$$\exp((Adj_R \cdot p)^\wedge) = R \cdot \exp(p^\wedge) \cdot R^{-1}$$

使得 $p = t \cdot v$，对于 $t \in \mathrm{R}$，在 $t = 0$ 处求导

$$\frac{t}{dt}|_{t=0} \exp((Adj_R \cdot t \cdot v)^\wedge) = \frac{t}{dt}|_{t=0} [R \cdot \exp(t \cdot v^\wedge) \cdot R^{-1}]$$

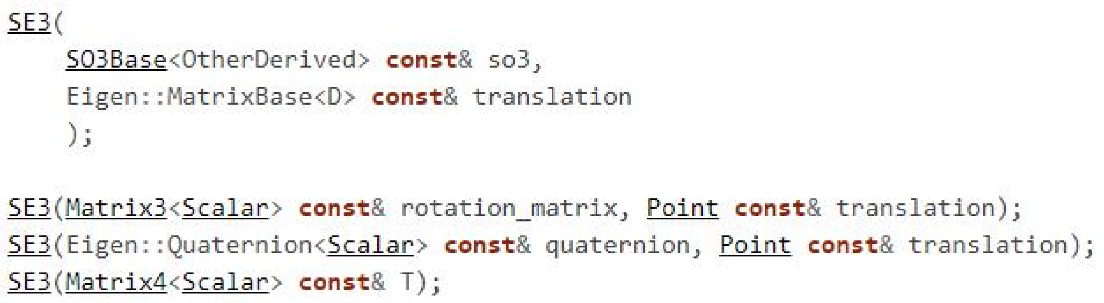
$$\frac{t}{dt}|_{t=0} [I + (Adj_R \cdot t \cdot v)^\wedge + O(t^2)] = R \cdot \frac{t}{dt}|_{t=0} [I + (t \cdot v^\wedge) + O(t^2)] R^{-1}$$

$$(Adj_R \cdot v)^\wedge = R \cdot v^\wedge \cdot R^{-1} = (R \cdot v)^\wedge$$

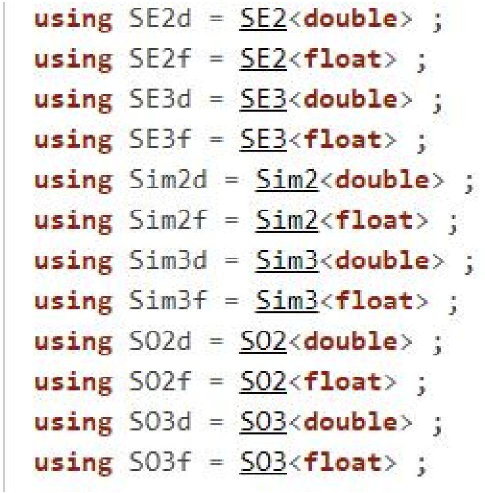
得到 $Adj_R = R$ 因此 $\exp((Rp)^\wedge) = R \exp(p^\wedge) R^T$

五： 轨迹的描绘

- Sophus 下载安装
https://github.com/strasdat/Sophus
- API Document
https://strasdat.github.io/Sophus/namespace_Sophus.html
- 模板类SE3，SE3Base

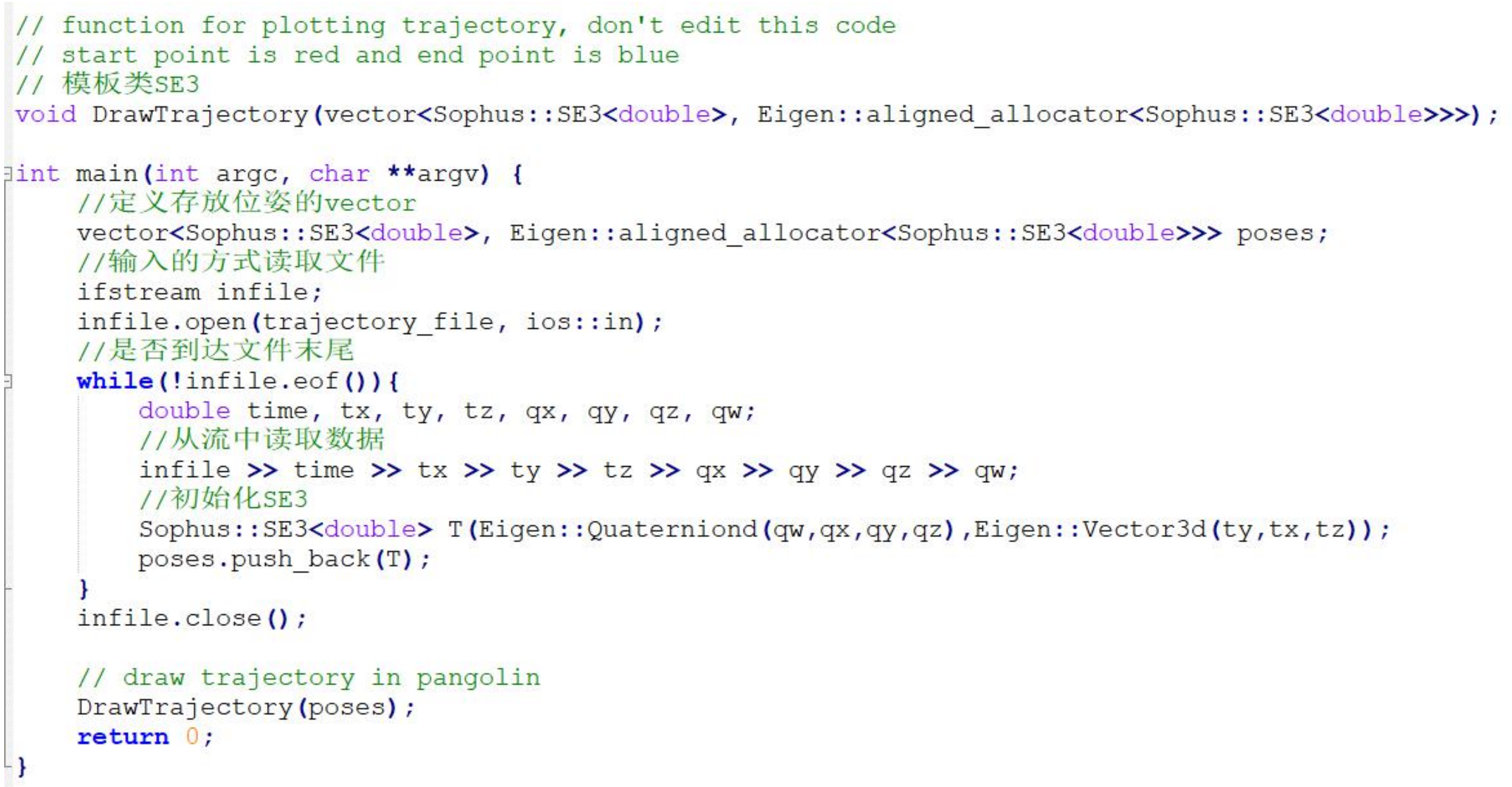


```
SE3(
    SO3Base<OtherDerived> const& so3,
    Eigen::MatrixBase<D> const& translation
    );

SE3(Matrix3<Scalar> const& rotation_matrix, Point const& translation);
SE3(Eigen::Quaternion<Scalar> const& quaternion, Point const& translation);
SE3(Matrix4<Scalar> const& T);
```



```
using SE2d = SE2<double> ;
using SE2f = SE2<float> ;
using SE3d = SE3<double> ;
using SE3f = SE3<float> ;
using Sim2d = Sim2<double> ;
using Sim2f = Sim2<float> ;
using Sim3d = Sim3<double> ;
using Sim3f = Sim3<float> ;
using SO2d = SO2<double> ;
using SO2f = SO2<float> ;
using SO3d = SO3<double> ;
using SO3f = SO3<float> ;
```

五： 轨迹的描绘



```
// function for plotting trajectory, don't edit this code
// start point is red and end point is blue
// 模板类SE3
void DrawTrajectory(vector<Sophus::SE3<double>, Eigen::aligned_allocator<Sophus::SE3<double>>>);

int main(int argc, char **argv) {
    //定义存放位姿的vector
    vector<Sophus::SE3<double>, Eigen::aligned_allocator<Sophus::SE3<double>>> poses;
    //输入的方式读取文件
    ifstream infile;
    infile.open(trajectory_file, ios::in);
    //是否到达文件末尾
    while(!infile.eof()){
        double time, tx, ty, tz, qx, qy, qz, qw;
        //从流中读取数据
        infile >> time >> tx >> ty >> tz >> qx >> qy >> qz >> qw;
        //初始化SE3
        Sophus::SE3<double> T(Eigen::Quaterniond(qw,qx,qy,qz),Eigen::Vector3d(ty,tx,tz));
        poses.push_back(T);
    }
    infile.close();

    // draw trajectory in pangolin
    DrawTrajectory(poses);
    return 0;
}
```

六：轨迹的误差

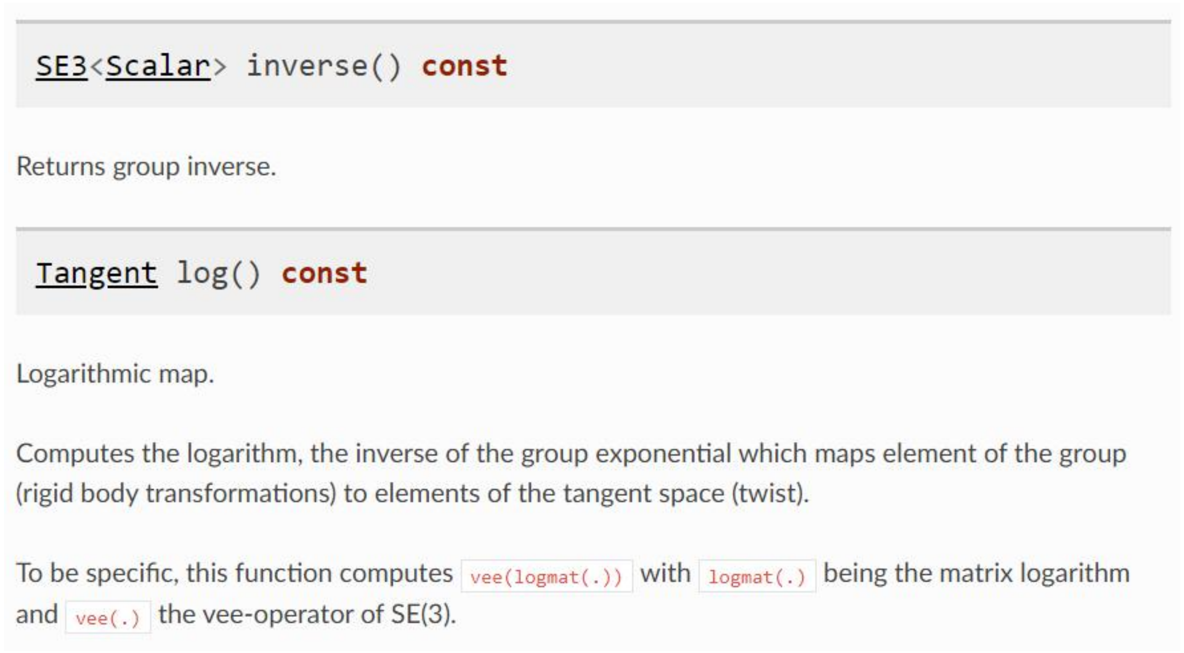
- SE3 与se3

$$T = \exp(\xi^{\wedge})$$
$$\xi = \ln T^{\vee}$$

- 两条轨迹的均方根误差

$$e_i = ||log(T_{gi}^{-1} T_{ei})^{\vee}||_2$$
$$RMSE(g,e) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2}$$

- SE3Base中函数



```
SE3<Scalar> inverse() const
```

Returns group inverse.

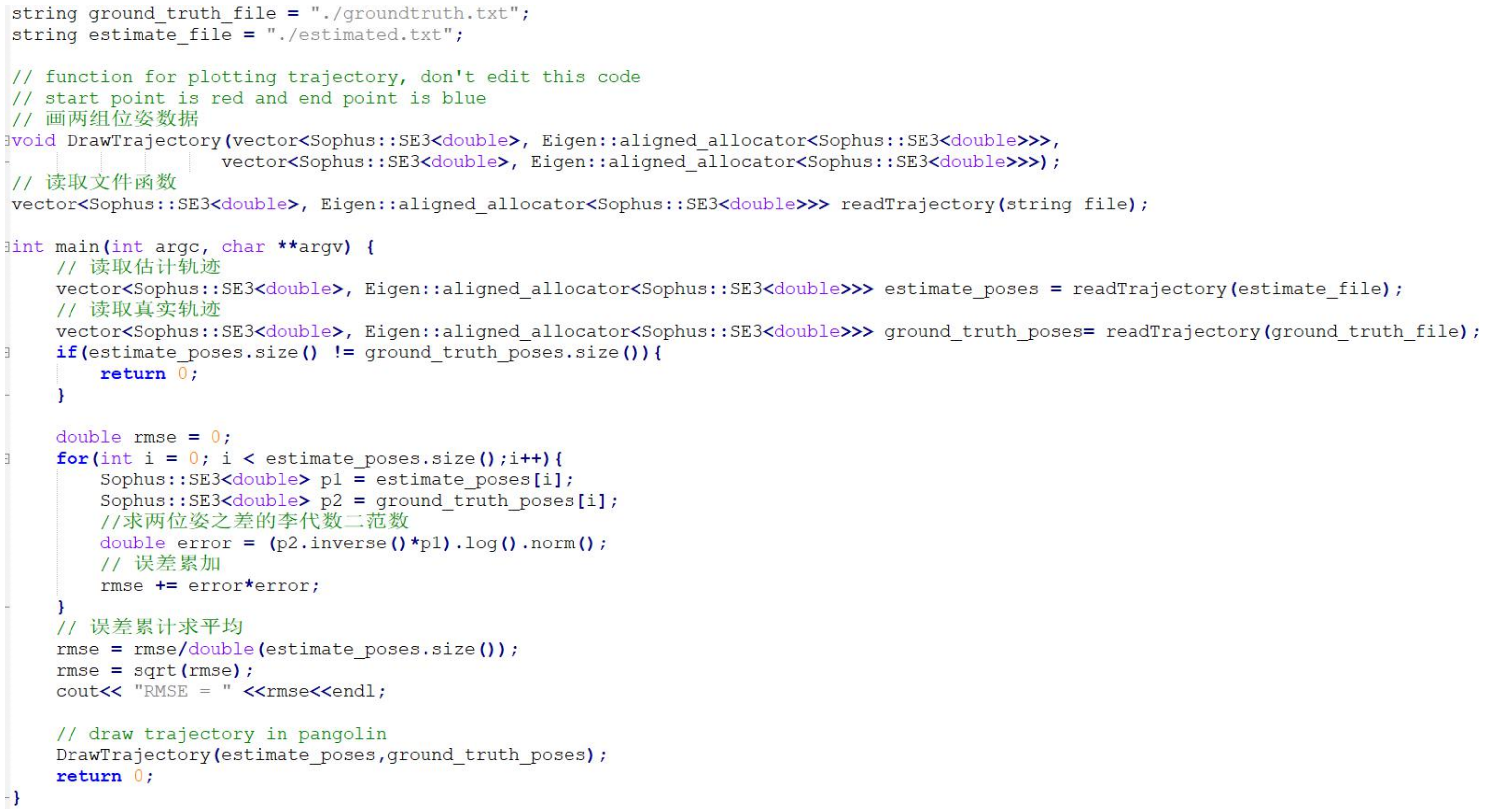
```
Tangent log() const
```

Logarithmic map.

Computes the logarithm, the inverse of the group exponential which maps element of the group (rigid body transformations) to elements of the tangent space (twist).

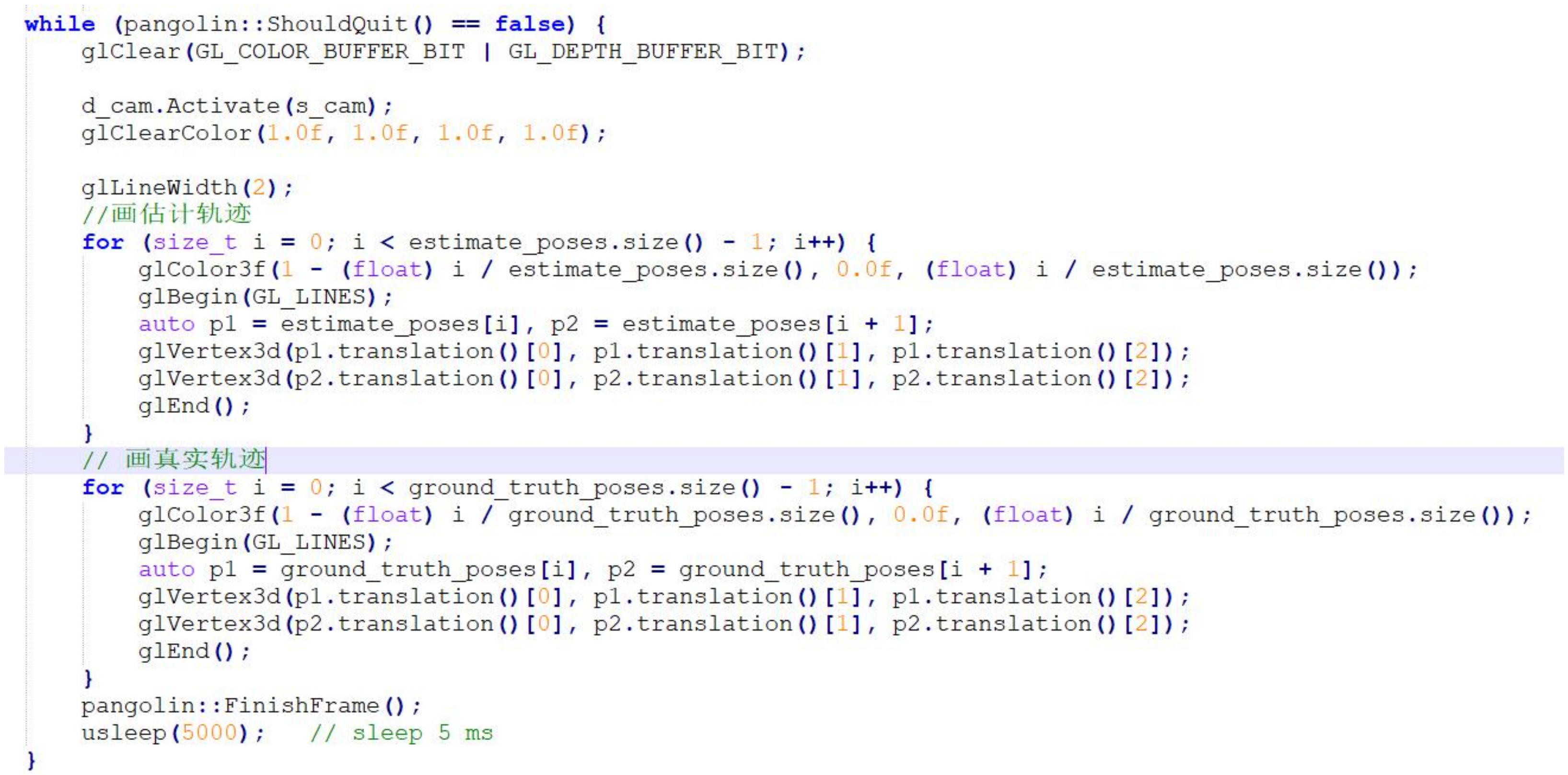
To be specific, this function computes `vee(logmat(.))` with `logmat(.)` being the matrix logarithm and `vee(.)` the vee-operator of SE(3).

六：轨迹的误差



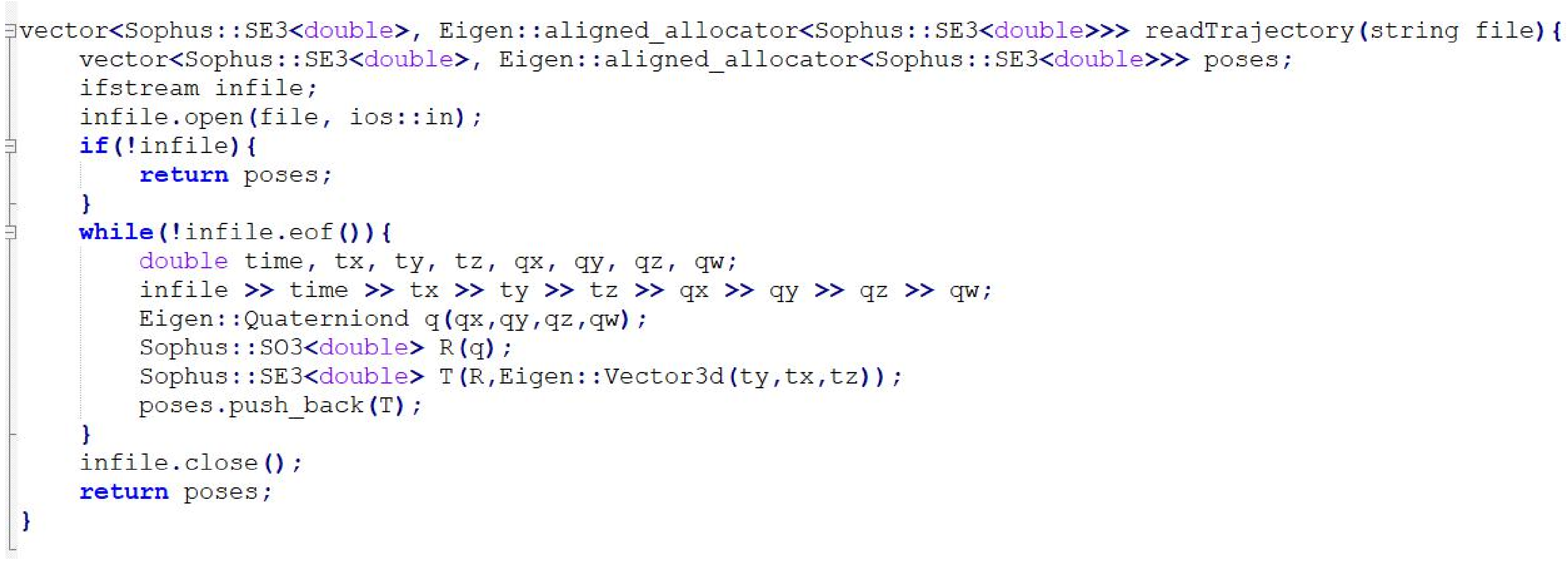
```
string ground_truth_file = "./groundtruth.txt";
string estimate_file = "./estimated.txt";

// function for plotting trajectory, don't edit this code
// start point is red and end point is blue
// 画两组位姿数据
void DrawTrajectory(vector<Sophus::SE3<double>, Eigen::aligned_allocator<Sophus::SE3<double>>>,
                    vector<Sophus::SE3<double>, Eigen::aligned_allocator<Sophus::SE3<double>>>);
// 读取文件函数
vector<Sophus::SE3<double>, Eigen::aligned_allocator<Sophus::SE3<double>>> readTrajectory(string file);

int main(int argc, char **argv) {
    // 读取估计轨迹
    vector<Sophus::SE3<double>, Eigen::aligned_allocator<Sophus::SE3<double>>> estimate_poses = readTrajectory(estimate_file);
    // 读取真实轨迹
    vector<Sophus::SE3<double>, Eigen::aligned_allocator<Sophus::SE3<double>>> ground_truth_poses= readTrajectory(ground_truth_file);
    if(estimate_poses.size() != ground_truth_poses.size()){
        return 0;
    }

    double rmse = 0;
    for(int i = 0; i < estimate_poses.size();i++){
        Sophus::SE3<double> p1 = estimate_poses[i];
        Sophus::SE3<double> p2 = ground_truth_poses[i];
        //求两位姿之差的李代数二范数
        double error = (p2.inverse()*p1).log().norm();
        // 误差累加
        rmse += error*error;
    }
    // 误差累计求平均
    rmse = rmse/double(estimate_poses.size());
    rmse = sqrt(rmse);
    cout<< "RMSE = " <<rmse<<endl;

    // draw trajectory in pangolin
    DrawTrajectory(estimate_poses,ground_truth_poses);
    return 0;
}
```

六： 轨迹的误差



```
while (pangolin::ShouldQuit() == false) {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    d_cam.Activate(s_cam);
    glClearColor(1.0f, 1.0f, 1.0f, 1.0f);

    glLineWidth(2);
    //画估计轨迹
    for (size_t i = 0; i < estimate_poses.size() - 1; i++) {
        glColor3f(1 - (float) i / estimate_poses.size(), 0.0f, (float) i / estimate_poses.size());
        glBegin(GL_LINES);
        auto p1 = estimate_poses[i], p2 = estimate_poses[i + 1];
        glVertex3d(p1.translation()[0], p1.translation()[1], p1.translation()[2]);
        glVertex3d(p2.translation()[0], p2.translation()[1], p2.translation()[2]);
        glEnd();
    }
    // 画真实轨迹
    for (size_t i = 0; i < ground_truth_poses.size() - 1; i++) {
        glColor3f(1 - (float) i / ground_truth_poses.size(), 0.0f, (float) i / ground_truth_poses.size());
        glBegin(GL_LINES);
        auto p1 = ground_truth_poses[i], p2 = ground_truth_poses[i + 1];
        glVertex3d(p1.translation()[0], p1.translation()[1], p1.translation()[2]);
        glVertex3d(p2.translation()[0], p2.translation()[1], p2.translation()[2]);
        glEnd();
    }
    pangolin::FinishFrame();
    usleep(5000);   // sleep 5 ms
}
```

六： 轨迹的误差



```
vector<Sophus::SE3<double>, Eigen::aligned_allocator<Sophus::SE3<double>>> readTrajectory(string file){
    vector<Sophus::SE3<double>, Eigen::aligned_allocator<Sophus::SE3<double>>> poses;
    ifstream infile;
    infile.open(file, ios::in);
    if(!infile){
        return poses;
    }
    while(!infile.eof()){
        double time, tx, ty, tz, qx, qy, qz, qw;
        infile >> time >> tx >> ty >> tz >> qx >> qy >> qz >> qw;
        Eigen::Quaterniond q(qx,qy,qz,qw);
        Sophus::SO3<double> R(q);
        Sophus::SE3<double> T(R,Eigen::Vector3d(ty,tx,tz));
        poses.push_back(T);
    }
    infile.close();
    return poses;
}
```

深蓝学院
shenlanxueyuan.com

感谢各位聆听！

Thanks for Listening