学习pinocchio（二）spatial 文件夹

zhuanlan.zhihu.com/p/698933718

尽心用心
一切都是命运石之门的选择

6 人赞同了该文章

目录

收起

2.5 fcl-pinocchio-conversions.hpp

2.6 force-base.hpp

(1) ForceBase类

2.7 force-dense.hpp

(1) ForceDense类

2.8 force-ref.hpp

(1) ForceRef类

2.9 force-tpl.hpp

(1) ForceTpl类

2.10 force.hpp

2.11 fwd.hpp

2.12 inertia.hpp

(1) InertiaBase类

(2) InertiaTpl类

2.13 log.hpp/hxx

(1) log3_impl结构体

(2) Jlog3_impl结构体

(3) log6_impl结构体

(4) Jlog6_impl结构体

2.14 Motion-base.hpp

(1) MotionBase类

2.15 Motion-dense.hpp

(1) MotionDense类

2.16 Motion-ref.hpp

(1) MotionRef类

2.17 Motion-tpl.hpp

(1) MotionTpl类

2.18 Motion-zero.hpp

2.19 Motion.hpp

2.20 se3-base.hpp

(1) SE3Base类

2.21 se3-tpl.hpp

(1) SE3Tpl类

2.22 se3.hpp

2.23 skew.hpp

(1) skew函数

(2) addSkew函数

(3) unSkew函数

(4) alphaSkew函数

(5) skewSquare函数

(6) cross函数

2.24 spatial-axis.hpp

(1) SpatialAxis类

2.25 symmetric3.hpp

(1) Symmetric3Tpl类

(2) SkewSquare结构体

(3) AlphaSkewSquare结构体
该文件夹内的源码主要是空间几何运算，包括SE3上的运算、指数映射运算，六维空间✦运动的运算、空间力的运算，空间惯性矩阵的运算。

2.1 act-on-set.hpp/hxx

该源码主要将SE3的伴随（微分）运算对应不同类的实际实现绑定，包括对力force，运动motion、惯量interia，均是封装的静态函数✦和一些块的优化计算。

主要分为两部分：forceSet和motionSet，每个部分是相似的。

(1) forceSet域名

l se3Action：表示六维力从坐标系i变换到坐标系j，坐标系i相对于坐标系j为m。

l se3ActionInverse：表示六维力从坐标系j变换到坐标系i，坐标系i相对于坐标系j为m。

l motionAction：表示六维力相对于v的微分（伴随）运算。

(2) motionSet域名

l se3Action：表示六维运动从坐标系i变换到坐标系j，坐标系i相对于坐标系j为m。

l se3ActionInverse：表示六维运动从坐标系j变换到坐标系i，坐标系i相对于坐标系j为m。

l motionAction：表示六维运动相对于v的微分（伴随）运算。

l inertiaAction：表示六维运动乘以惯性力生成六维力。

l act：表示六维运动对于六维力作用，与forceSet：：motionAction作用相同。

2.2 cartesian-axis.hpp

该源码主要功能是笛卡尔坐标轴对三维矢量的叉乘运算（矢量绕XYZ笛卡尔坐标✦的微分运算）。

(1) CartesianAxis类

成员对象：

无。

成员函数：

l cross：定义了XYZ三个坐标轴叉乘三维矢量运算✦，即XYZ单位矢量叉乘输入矢量得到输出矢量。

l alphaCross：定义了带系数的XYZ三个坐标轴叉乘三维矢量运算，即XYZ单位矢量叉乘输入矢量，再乘以标量系数✦得到输出矢量。

l setTo：提取当前类XYZ坐标轴的三维矢量。

2.3 explog-quaternion.hpp

该源码主要是实现四元数与李群和李代数✦的变换，主要以函数为主。

(1) exp3函数

将三维速度矢量积分（指数映射✦）成四元数。

(2) log3函数

将四元数对应的旋转矩阵对数映射成三维矢量，获取等效旋转角度。

(3) Jexp3CoeffWise函数

四元数导数与指数映射导数的雅克比矩阵✦，v为姿态指数映射三维矢量，Jexp为所求4x3雅克比矩阵，满足 $q = J_{exp}\dot{\xi}$ 。

(4) Jlog3函数

用四元数表示的姿态求其指数映射的雅克比矩阵，Jlog为 $dexp^{-1}_{\xi}$，满足 $\dot{\xi} = dexp^{-1}_{\xi}V$ 。

2.4 explog.hpp

该源码主要是指数映射和李代数之间的变换，主要以函数为主。每个函数均有一个参数Op：SETTO为直接计算，ADDTO为在原有数据增加计算后的结果，RMTO为在原有数据减少计算后的结果，其他报错。

(1) exp3函数

三维指数映射矢量求姿态矩阵✦。

(2) log3函数

姿态矩阵的对数运算求三维指数映射。

(3) Jexp3函数

姿态矩阵的指数映射一阶导雅克比矩阵，Jlog为 $dexp_{\_\xi}$，满足 $V = dexp_{\_\xi}\xi$ 与Jlog3互为逆矩阵。

(4) Jlog3函数

姿态矩阵的指数映射一阶导雅克比矩阵，Jlog为 $dexp^{-1}_{\_\xi}$，满足 $\xi = dexp^{-1}_{\_\xi}V$ 与Jexp3互为逆矩阵。

(5) exp6函数

六维指数映射矢量求位姿矩阵。

(6) log6函数

位姿矩阵的对数运算求六维指数映射。

(7) Jexp6函数

位姿矩阵的指数映射一阶导雅克比矩阵，Jlog为 $dexp_{\_\xi}$，满足 $V = dexp_{\_\xi}\xi$ 与Jlog6互为逆矩阵。

(8) Jlog6函数

位姿矩阵的指数映射一阶导雅克比矩阵，Jlog为 $dexp^{-1}_{\_\xi}$，满足 $\xi = dexp^{-1}_{\_\xi}V$ 与Jexp6互为逆矩阵。

(9) Interpolate函数

从位姿A到位姿B，映射在指数空间线性插值✦。

2.5 fcl-pinocchio-conversions.hpp

该源码主要是将pinocchio和hpp::fcl的位姿矩阵相互转化。

2.6 force-base.hpp

该源码主要是六维空间力的的虚基类✦，定义类空间运动的基本操作。

(1) ForceBase类

成员对象：

无。

成员函数：

l se3Action：六维空间力从物体坐标系转换成绝对坐标系，具体实现是由派生类的se3Action_impl实现，m为位姿矩阵，f为六维力，

l se3ActionInverse：六维运动矢量从绝对坐标系转换成物体坐标系，具体实现是由派生类的se3ActionInverse_impl实现，m为位姿矩阵，f为六维力，

l motionAction：六维空间力对运动的伴随运算，具体实现是由派生类的motionAction实现，v为输入 ，输出

l dot：六维力与六维运动点乘（计算功率），点乘=角速度点乘力矩+线速度点乘力，具体实现是由派生类。

2.7 force-dense.hpp

该源码为六维空间力的force-base派生类，实现了大部分空间力功能，主要将六维力分成力和力矩进行实现。

(1) ForceDense类

成员对象：

无。

成员函数：

l se3Action_imp：六维空间力从物体坐标系转换成绝对坐标系，m为位姿矩阵，f为六维力，

l se3ActionInverse_impl：六维运动矢量从绝对坐标系转换成物体坐标系，m为位姿矩阵，f为六维力，

l motionAction：六维空间力对运动的伴随运算，v为输入 ，输出

l dot：六维力与六维运动点乘（计算功率），点乘=角速度点乘力矩+线速度点乘力。

2.8 force-ref.hpp

该源码为六维空间力的force-dense派生类，补充了部分空间力功能，主要将六维力分成合成一个六维矢量进行实现。

(1) ForceRef类

成员对象：

l m_ref：六维力。

成员函数：

2.9 force-tpl.hpp

该源码为六维空间力的force-dense派生类，实现了大部分空间力功能，主要将六维力分成一个六维矢量进行实现。

(1) ForceTpl类

成员对象：

l m_data：六维力矢量。

成员函数：

2.10 force.hpp

该源码主要为了将force的进行统一命名。

2.11 fwd.hpp

该源码主要是对整个文件夹包含文件的命名简化。

2.12 inertia.hpp

该源码主要用于实现空间空间惯性矩阵的相关定义和操作，主要是一个基类和一个派生类，在构造6x6空间矩阵时候，写法为

$$M = \begin{bmatrix} m & -m\hat{p} \\ \hat{mp} & I - \mathrm{m}\hat{p}^2 \end{bmatrix}.$$

存储形式是物体质量m，提取mass();，质心位置 提取lever();，质心惯量I提取inertia();

矩阵的逆为 $M^{-1} = \begin{bmatrix} 1/m - \hat{p}I^{-1}\hat{p} & \hat{p}I^{-1} \\ -I^{-1}\hat{p} & I^{-1} \end{bmatrix}$。

(1) InertiaBase类

成员对象：

无。

成员函数：

l vtiv：计算运动与惯性力的二次型，动能的两倍，具体由派生类实现， $out = V^T MV$

l variation：空间惯性矩阵的一阶微分，具体有派生类实现， ，看作vxi函数和ivx函数求差。

l vxi：空间惯性矩阵的左乘六维运动的伴随矩阵✦（左切空间），用于计算空间惯性矩阵的微分，具体由派生类实现，

l ivx：空间矩阵矩阵的右乘六维运动的伴随矩阵（右切空间），用于计算空间惯性矩阵的微分，具体由派生类实现，

l se3Action：空间惯性矩阵从物体坐标系变换到绝对坐标系，具体由派生类实现，

$$m_2 = m_1 , p_2 = Rp_1 + P , I_2 = RI_1R^T$$

l se3ActionInverse：空间惯性矩阵从绝对坐标系变换到物体坐标系，具体由派生类实现，

$$m_2 = m_1 , p_2 = R^T (p_1 - P) , I_2 = R^T I_1 R$$

l __mult__：单刚体惯性矩阵和运动（六维加速度）乘法，输出六维力，$F = MA$。

(2) InertiaTpl类

成员对象：

l m_mass：物体质量（标量）

l m_com：质心位置（三维矢量）

l m_inertia：质心惯量（三维对称矩阵）

成员函数：

l vtiv：计算运动与惯性力的二次型，动能的两倍，$out = V^T MV$

l variation：空间惯性矩阵的一阶微分，，看作vxi函数和ivx函数求差。

l vxi：空间惯性矩阵的左乘六维运动的伴随矩阵（左切空间），用于计算空间惯性矩阵的微分，

l ivx：空间矩阵矩阵的右乘六维运动的伴随矩阵（右切空间），用于计算空间惯性矩阵的微分，

l se3Action：空间惯性矩阵从物体坐标系变换到绝对坐标系，

$m_2 = m_1 , p_2 = Rp_1 + P , I_2 = RI_1R^T$

l se3ActionInverse：空间惯性矩阵从绝对坐标系变换到物体坐标系，

$$m_2 = m_1 , p_2 = R^T (p_1 - P) , I_2 = R^T I_1 R$$

l toDynamicParameters：动态提取数据，将三个数据串联成一个矢量。

l FromDynamicParameters：动态获取数据，将一串矢量转化成存储形式。

2.13 log.hpp/hxx

该源码主要是实现姿态矩阵或齐次位姿矩阵的对数运算（指数映射）。

(1) log3_impl结构体

将旋转矩阵对数运算成三维矢量，只有一个run函数。

输入：R为旋转矩阵。

输出：theta为等效旋转角度（标量），res为指数映射（三维矢量，模长大小为theta）。

(2) Jlog3_impl结构体

根据指数状态求解指数映射一阶导数的雅可比矩阵✦。

输入：theta为等效旋转角度（标量），log为指数映射（三维矢量，模长大小为theta）。

输出：Jlog为 dexp_{-\xi}^{-1}，满足\dot{\xi}=dexp_{-\xi}^{-1}V。

(3) log6_impl结构体

将姿态矩阵和位置矢量对数运算成六维矢量，包括姿态和位置映射，只有一个run函数。

输入：M为SE(3)的位置和姿态信息。

输出：mout为对数映射六维空间，包括姿态和位置映射。

(4) Jlog6_impl结构体

根据姿态矩阵和位置矢量求指数映射一阶导的雅可比矩阵。

输入：M为SE(3)的位置和姿态信息。

输出：Jlog为 dexp_{-\xi}^{-1}，满足\dot{\xi}=dexp_{-\xi}^{-1}V。

2.14 Motion-base.hpp

该源码为六维空间运动的虚基类，定义类空间运动的基本操作。

(1) MotionBase类

为空间运动的虚基类，具体实现在Motion-dense.hpp、Motion-ref、Motion-tpl等文件。

成员对象：

无。

成员函数：

l toActionMatrix：六维运动矢量的6x6伴随矩阵，具体实现是由派生类的toActionMatrix_impl实现，\text{ad}_V= \left[{\begin{array}{*{20}{c}} {\hat{w}}&{\hat{v}}\\ {0}&{\hat{w}} \end{array}} \right]。

l toDualActionMatrix：六维运动矢量的6x6伴随矩阵的对偶形式，具体实现是由派生类的toDualActionMatrix_impl实现，\text{ad}_V'= \left[{\begin{array}{*{20}{c}} {\hat{w}}&{0}\\ {\hat{v}}&{\hat{w}} \end{array}} \right]

l se3Action：六维运动矢量从物体坐标系转换成绝对坐标系，具体实现是由派生类的se3Action_impl实现，m为位姿矩阵，v为六维运动，out=\text{ad}_GV

l se3ActionInverse：六维运动矢量从绝对坐标系转换成物体坐标系，具体实现是由派生类的se3ActionInverse_impl实现，m为位姿矩阵，v为六维运动，out=\text{ad}_G^{-1}V

l dot：六维运动与六维力点乘（计算功率），点乘=角速度点乘力矩+线速度点乘力，具体实现是由派生类。

2.15 Motion-dense.hpp

该源码为六维空间运动的motion-base派生类，实现了大部分空间运动功能，主要将六维运动分成线速度和角速度进行实现。

(1) MotionDense类

成员对象：

成员函数：

l toActionMatrix_impl：六维运动矢量的6x6伴随矩阵\text{ad}_V= \left[{\begin{array}{*{20}{c}} {\hat{w}}&{\hat{v}}\\ {0}&{\hat{w}} \end{array}} \right]。

l toDualActionMatrix_impl：六维运动矢量的6x6伴随矩阵的对偶形式，\text{ad}_V'= \left[{\begin{array}{*{20}{c}} {\hat{w}}&{0}\\ {\hat{v}}&{\hat{w}} \end{array}} \right]

l motionAction：六维运动矢量的6x6伴随运算，v为输入 V_B ，输出 out=\text{ad}_VV_B

l se3Action_impl：六维运动矢量从物体坐标系转换成绝对坐标系，m为位姿矩阵，v为六维运动，out=\text{Ad}_GV

l se3ActionInverse_impl：六维运动矢量从绝对坐标系转换成物体坐标系，m为位姿矩阵，v为六维运动， out=\text{Ad}_G^{-1}V

l dot：六维运动与六维力点乘（计算功率），点乘=角速度点乘力矩+线速度点乘力。

2.16 Motion-ref.hpp

该源码为六维空间运动的motion-dense派生类，补充了部分空间运动功能，主要将六维运动分成合成一个六维矢量进行实现。

(1) MotionRef类

成员对象：

l m_ref：六维运动矢量。

成员函数：

2.17 Motion-tpl.hpp

该源码为六维空间运动的motion-dense派生类，实现了大部分空间运动功能，主要将六维运动分成一个六维矢量进行实现。

(1) MotionTpl类

成员对象：

l m_data：六维运动矢量。

成员函数：

2.18 Motion-zero.hpp

该源码为六维空间运动的motion-base派生类，零运动的相关功能。

2.19 Motion.hpp

该源码主要为了将motion的进行统一命名。

2.20 se3-base.hpp

该源码主要构件位置和姿态SE3上基类，用于派生se3-tpl.hpp文件。

(1) SE3Base类

可以认为是虚基类，实现在派生类SE3Tpl中

2.21 se3-tpl.hpp

该源码主要是在SE3Base基类上实现SE3位姿矩阵的伴随（微分）运算。

(1) SE3Tpl类

成员对象：

l rot：姿态三维矩阵。

l trans：位置三维矢量

成员函数：

l inverse：位置矩阵求逆运算。

l toHomogeneousMatrix_impl：生成4x4齐次矩阵 G= \left[{\begin{array}{*{20}{c}} {R}&{P}\\ {0}&{1} \end{array}} \right]。

l toActionMatrix_impl：生成6x6伴随矩阵 \text{Ad}_G= \left[{\begin{array}{*{20}{c}} {R}&{\hat{P}R}\\ {0}&{R} \end{array}} \right] 。

l toActionMatrixInverse_impl：生成6x6伴随矩阵的逆\text{Ad}_G^{-1}= \left[{\begin{array}{*{20}{c}} {R^T}&{-R^T\hat{P}}\\ {0}&{R^T} \end{array}} \right]。

l toDualActionMatrix_impl：生成6x6伴随矩阵的对偶形式 \text{Ad}_G'= \left[{\begin{array}{*{20}{c}} {R}&{0}\\ {\hat{P}R}&{R} \end{array}} \right] 。

l act_impl：从物体坐标系转化到绝对坐标系的伴随变换，根据结构不同计算不同，包括位姿矩阵，六维空间矢量，三维矢量。G_S=GG_B或\xi_S=\text{Ad}_G\xi_S或 p_S=Rp_B+P

l actInv_impl：从绝对坐标系转化到物体坐标系的伴随变换，根据结构不同计算不同，包括位姿矩阵，六维空间矢量，三维矢量。G_B=G^{-1}G_S 或 \xi_B=\text{Ad}_G^{-1}\xi_S 或 p_B=R^T(p_S-P)

l actOnEigenObject：三维矢量作用在绝对坐标中，即在位姿坐标系下的三维矢量坐标变化 p_S=Rp_B+P

l actInvOnEigenObject：三维矢量作用在物体坐标中，即在位姿坐标系下的三维矢量坐标变化 p_B=R^T(p_S-P)

l Interpolate：SE3的线性插值虚函数。在explog.hpp实现或调用Eigen的quaternion操作。

2.22 se3.hpp

该源码主要功能是集成SE3和其他模块的命名。

2.23 skew.hpp

该结构体主要是三维矢量的三维反对称矩阵✦运算等，以内联模板函数为主。

(1) skew函数

将三维矢量生成三维反对称矩阵。

(2) addSkew函数

将一个三维矢量反对称运算后加入一个已经有的三维矩阵中。

(3) unSkew函数

将三维反对称矩阵转换成三维矢量。

(4) alphaSkew函数

将三维矩阵乘以一个标量并生成三维反对称矩阵。

(5) skewSquare函数

计算平方叉乘线性算子C。

u\times(v\times a)=\hat{u}\hat{v}a\equiv Ca

(6) cross函数

将叉乘应用于矩阵，即三维矢量的反对称矩阵乘以输入矩阵得到输出矩阵。 M_{out}=\hat{v}M_{in}

2.24 spatial-axis.hpp

该源码主要功能是实现XYZ单轴对六维空间矢量（如运动，力旋量）的叉乘运算。旋量绕XYZ笛卡尔坐标的微分运算。

(1) SpatialAxis类

成员对象：

成员函数：

l cross：定义了实现XYZ单轴对六维空间矢量（如运动微分，力微分）的叉乘运算，注意运动和力之间的旋量写法是反向的，所以两个运算也是相反的。

l motionAction：定义了六维空间运动对XYZ单轴的叉乘运算，与cross函数对六维运动的叉乘运算相差负号。

2.25 symmetric3.hpp

该源码主要功能是实现三维对称矩阵相关运算，为惯性矩阵运算做基础。

(1) Symmetric3Tpl类

成员对象：

l m_data (Eigen 6x1矩阵)：3x3对称矩阵的下三角数据。

m= \left[{\begin{array}{*{20}{c}} {m_data[0]}&{m_data[3]}\\ {m_data[1]}&{m_data[2]}&{m_data[4]}\\ {m_data[3]}&{m_data[4]}&{m_data[5]} \end{array}} \right]

成员函数：

l vtiv：将对称矩阵当作输入参数的二次型求值， out=v^Tlv 。

l vxs：将三维对称矩阵每一列和三维矢量叉乘生成输出矩阵， out= \left[{\begin{array}{*{20}{c}} {S_3(1)\times v}&{S_3(2)\times v}&{S_3(3)\times v} \end{array}} \right], {S_3(i)} 为矩阵的第i列。

l svx：将三维矢量和三维对称矩阵每一列叉乘生成输出矩阵。

out= \left[{\begin{array}{*{20}{c}} {v\times S_3(1)}&{v\times S_3(2)}&{v\times S_3(3)} \end{array}} \right], {S_3(i)}为矩阵的第i列。

l rhsMult：将三维对称矩阵和三维输入矢量正常矩阵乘法✦得出三维输出矢量。

l decomposeIt：成员矩阵（对称矩阵）的Cholesky分解的L。

l rotate：将成员矩阵（对称矩阵）进行酉矩阵（旋转矩阵）的伴随变换， out=RIR^T 。

(2) SkewSquare结构体

该结构体是求三维矢量的反对称矩阵的平方（为对称矩阵）。

(3) AlphaSkewSquare结构体

该结构体是求带系数的三维矢量的反对称矩阵的平方（为对称矩阵）。
编辑于 2024-05-24 17:39・IP 属地北京

内容所属专栏

多足轮腿机器人

开开心心研究多足轮腿机器人

pinocchio

动力学

机器人