# 矩阵Eigen：

**package：Ceres**

**lib：${CERES\_LIBRARIES}**

**namespace：Eigen**

## 分解Cholesky：

**todo: .ldlt**

|  |  |  |
| --- | --- | --- |
| LDLT | | Cholesky分解器 **()** |
| **方法** | **transpositionsP()** | **排列矩阵** |
| **matrixL()** | **下三角矩阵** |
| **vectorD()** | **对角线向量** |
| **solve(b)** | **求解** |

## 内核Core：

|  |  |
| --- | --- |
| 宏： | |
| EIGEN\_MAKE\_ALIGNED\_OPERATOR\_NEW | **重写内存对齐**的**new**、delete运算 |

|  |  |
| --- | --- |
| l1CacheSize / l2CacheSize / l3CacheSize() | 返回**CPU缓存**大小 |
| aligned\_allocator<Type> | **内存分配**器 |

|  |  |  |
| --- | --- | --- |
| Vector<dtype, nr> / VectorXd | | **向量** |
| Matrix<dtype, nr=-1, nc=-1> / MatrixXd | | **矩阵** |
| Map<EigenT> | | **映射 (浅拷贝)** |
| 形参 | \*dtype | **深拷贝数组 / 矩阵** |
| 属性 | size() | 元素数 |
| rows / cols() | 行列数 |
| cast<dtype>() | 类型转换 |
| array() | **返回数组** (值传递) |
| 复数 | real / imag() | 实/虚部 |
| conjugate() | **对应共轭复数** |
| 元素引用 | (r, c) / row(i) / col(i) | **指定**元素 |
| segment<n>(i) / block<nr, nc>(ir, ic) | **切片** |
| 运算 | << x0, x1, ... | 写入元素 |
| \* x | 矩阵乘法 |
| dot(x) | 向量点积 |
| eval() | 派生解除 |
| 变换 | transpose() | 派生转置 |
| asDiagnal() | 派生对角矩阵 |
| **reverse / reverseInPlace()** | **所有轴翻转** |
| 矩阵创建 | ::Identity(nr, nc) | 单位阵 |
| ::Zero / ::Ones / ::Random(nr, nc) | 默认值**填充** |
| ::Constant(nr, nc, v) / fill(v) | 指定值**填充** |
| normalize / normalized() | 单位化 **(原地/副本)** |
| 向量创建 | ::LinSpaced(size, low, high) | 等差**数列** |
| ::Unit(i) / ::UnitX / ::UnitY / ::UnitZ | 旋转向量 |
| 线性代数 | trace() | 迹 **(对角线之和)** |
| inverse() | 逆 **(include <.../Dense>)** |
| determinant() | 行列式 |
| eulerAngles(2, 1, 0) | **返回**欧拉角 |
| ldlt() | **返回**Cholesky分解 |
| jacobiSvd / bdcSvd(**DecompositionOptions** opt) | **返回**奇异值分解 |

|  |  |  |
| --- | --- | --- |
| Array<dtype, nr=-1, nc=-1> | | **数组** |
| 属性 | matrix() | **矩阵** (值传递) |
| 函数 | all / any / count / hasNaN() | 非零**判断** |
| mean / sum / prod / norm() | 均值 / 和 / 积 / 二范数 |
| sign / abs / sqrt / pow(x) | 基本 |
| maxCoeff / minCoeff() | **最值** |
| cwiseMax / cwiseMin(x) | 逐元素最值 |
| sin / cos / tan / asin / acos / atan() | **三角函数** |
| exp / log / log2 / log10 / logistic() | **指数函数** |
| round / ceil / floor() | 取整 |
| isInf / isFinite / isNaN() | 值**判断** |

|  |  |  |
| --- | --- | --- |
| DiagonalMatrix<dtype, n> | | **对角矩阵** |
| **属性** | diagonal() | **对角向量** |

## 稠密Dense：

|  |  |  |
| --- | --- | --- |
| **特征值分解：** | | |
| EigenSolver<mat>(mat) | | **普通分解器** |
| SelfAdjointEigenSolver<mat>(mat) | | **对称阵分解器** |
| 方法 | **eigenvalues()** | **特征值** |
| **eigenvectors()** | **特征向量** |

## 几何Geometry：

|  |  |  |
| --- | --- | --- |
| AngleAxisd(degree, axis) | | **旋转向量** |
| **运算** | \* vector | **对向量进行旋转** |
| **方法** | matrix() | **旋转矩阵** |

**点变换：**

|  |  |  |
| --- | --- | --- |
| Quaterniond(w, x, y, z) | | **单位四元数 (实部为0表示点)** |
| **属性** | matrix() | **矩阵** |
| **创建** | ::Identity() | **单位旋转阵** |
| **方法** | coeffs() | **虚部、实部** |

|  |  |  |
| --- | --- | --- |
| Isometry3d | | **等距变换 (齐次坐标系)** |
| **创建** | ::Identity() | **单位旋转阵** |
| **方法** | matrix() | **矩阵** |
| prerotate / rotate(rotation) | **原地旋转 (绝对/相对)** |
| pretranslate / translate(vec) | **原地平移 (绝对/相对)** |

## 奇异值SVD：

|  |  |  |
| --- | --- | --- |
| JacobiSVD<mat> | | **迭代分解** |
| BDCSVD<mat> | | **大矩阵分解** |
| **形参** | mat, **DecompositionOptions** opt | **分解模式** |
| **属性** | matrixU / singularValues / matrixV() |  |
| **方法** | solve(b) |  |

# 八叉树octomap/octomap.h：

**package：octomap**

**lib：${OCTOMAP\_LIBRARIES}**

**namespace：octomap**

**bin：octovis**

|  |  |
| --- | --- |
| point3d(x, y, z) | 3D点 |

|  |  |  |
| --- | --- | --- |
| Pointcloud | | 点云 |
| 方法 | push\_back(x, y, z) | **添加**点 |

|  |  |  |
| --- | --- | --- |
| OcTree | | **八叉树**地图 |
| 形参 | resolution | 分辨率 |
| file | 二进制文件 |
| 方法 | insertPointCloud(cloud, point3d origin) | **给定原点**插入点云 |
| updateInnerOccupancy() | 更新节点**占用** |
| writeBinary(file) | 导出**二进制文件** |

# 绘图pangolin/pangolin.h：

**package：Pangolin**

**l**ib：${Pangolin\_LIBRARIES}

**namespace：pangolin**

|  |  |  |
| --- | --- | --- |
| OpenGlMatrix (4×4)： | | |
| ProjectionMatrix(w, h, fu, fv, u0, v0, zNear, zFar) | | 投影矩阵 |
| ModelViewLookAt(cx, cy, cz, lx, ly, lz, AxisDirection up = AxisY) | | 视图矩阵 |
| 创建 | SetIdentity() | 恒等映射 |
| 元素引用 | (r, c) | **指定**元素 |
| 线性代数 | Transpose() | 转置 |
| Inverse() | 逆 |

|  |  |  |
| --- | --- | --- |
| 函数 | CreateWindowAndBind(title, width, height) | 创建窗口 |
| bool ShouldQuit() | 状态 |
| FinishFrame() | 帧提交 |
| 类 | Handler3D(OpenGlRenderState) | 3D事件处理器 |

|  |  |  |
| --- | --- | --- |
| 渲染： | | |
| OpenGlRenderState(proj\_mat, view\_mat) | | 渲染状态 |
| 属性 | GetProjectionMatrix() / SetProjectionMatrix(m) | 投影矩阵 |
| GetModelViewMatrix() / SetModelViewMatrix(m) | 视图矩阵 |
| 方法 | Follow(OpenGlMatrix T\_wc) | 跟随 |

|  |  |  |
| --- | --- | --- |
| 面板变量： | | |
| Var<T> | | 面板元素 |
| 形参 | name, value | 变量 |
| name, value, toggle | 复选框 |
| name, value, min, max, logscale | 滑块 |

|  |  |  |
| --- | --- | --- |
| 视图： | | |
| View CreateDisplay() | | 视图 |
| View CreatePanel(name) | | 面板 |
| 设置 | SetBounds(bottom, top, left, right, aspect) | 位置 (Attach::Pix) |
| SetHandler(Handler \*) | 处理器 |
| 方法 | Activate(OpenGlRenderState) | 读取渲染状态 |

## 渲染OpenGL：

|  |  |
| --- | --- |
| glEnable / glDisable(GL\_DEPTH\_TEST) | 功能开关 (深度测试，以处理遮挡) |
| glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT) | 清空画布 |
| glVertex3f(x, y, z) | 图形顶点 |
| glBegin(mode) / glEnd() | 绘图模式 |
| glLineWidth(int) / glPointSize(int) | 设置 |
| glClearColor(r, g, b, a) / glColor3f(r, g, b) | 绘图参数 |
| glPushMatrix() / glPopMatrix() / glMultMatrixf(OpenGlMatrix.m) | 变换矩阵堆顶操作 |

|  |  |  |  |
| --- | --- | --- | --- |
| 绘图模式： | | | |
| GL\_POINTS | 点 | GL\_LINES | 线段 |

# 点云pcl：

**package：PCL**

**lib：${PCL\_LIBRARIES}**

**namespace：pcl**

**bin：pcl\_viewer \*.pcd**

## 滤波filters：

|  |  |  |
| --- | --- | --- |
| Filter<PointT> | | **滤波器**基类 **(filter.h)** |
| **设置** | setInputCloud(pcl) | 输入点云 |
| **方法** | filter(pcl) | 输出**滤波结果** |

|  |  |  |
| --- | --- | --- |
| StatisticalOutlierRemoval<PointT> | | **基于点邻域统计的**离群点滤除  **(statistical\_outlier\_removal.h)** |
| **设置** | setMeanK(k) | **平均距离估计的**点数 |
| setStddevMulThresh(stddev\_mult) | 标准差倍数 |

|  |  |  |
| --- | --- | --- |
| VoxelGrid<PointT> | | 3D体素网格**过滤器，保留每个体素的质心**  **(voxel\_grid.h)** |
| **设置** | setLeafSize(lx, ly, lz) | 体素尺寸 |

## 流io：

namespace：io

|  |  |
| --- | --- |
| **pcd\_io.h：** | |
| **loadPCDFile("\*.pcd", pcl)** | 加载点云**文件** |
| savePCDFileBinary("\*.pcd", pcl) | 导出点云**文件** |

## 点云point\_cloud.h：

|  |  |  |
| --- | --- | --- |
| PointCloud<PointT> | | 点云 |
| **形参** | pc, indices | **深拷贝** |
| **运算** | + pcl | **叠加点云** |
| **属性** | ::Ptr | **引用计数指针** |
| is\_dense | **点无缺失** |
| points | **点容器** |
| size() | **大小** |

## 点类point\_types.h：

|  |  |  |
| --- | --- | --- |
| PointXYZRGB(r, g, b) | | RGB**空间点** |
| PointXYZRGBNormal | | RGB**空间点 (带**法线**)** |
| **属性** | r / g / b / x / y / z | 颜色 **/** 位置 |
| normal\_x / normal\_y / normal\_z | 法线 |

## 网格PolygonMesh.h：

|  |  |
| --- | --- |
| PolygonMesh | 三角网格 |

## 注册registration：

|  |  |  |
| --- | --- | --- |
| IterativeClosestPoint<PointSrc, PointTarget> | | 迭代最近点 **(icp.h)** |
| **方法** | setInputSource / setInputTarget(pcl) | 输入点云 |
| align(pcl) | **对齐并**输出 |
| **属性** | hasConverged() | 收敛状态 |
| getFinalTransformation() | 欧氏变换 |
| getFitnessScore() | 平均距离 |

## 搜索search：

namespace：search

|  |  |  |
| --- | --- | --- |
| KdTree<PointT> | | k-d树 **(kdtree.h)** |
| **设置** | setInputCloud(pcl) | 输入点云 |

## 表面surface：

|  |  |  |
| --- | --- | --- |
| MovingLeastSquares<PointInT, PointOutT> | | 移动最小二乘 **(mls.h)** |
| **设置** | setInputCloud(pcl) | 输入点云 |
| setSqrGaussParam(radius\_square) | 高斯参数**的平方** |
| setComputeNormals(bool) | **启用**法线**计算** |
| setSearchMethod(tree) | 搜索算法 |
| setSearchRadius(radius) | 搜索半径 |
| setPolynomialFit(bool) | **启用**多项式近似 |
| setPolynomialOrder(order) | **多项式拟合**阶数 |
| **方法** | process(pcl) | 输出**表面平滑结果** |

|  |  |  |
| --- | --- | --- |
| GreedyProjectionTriangulation<PointT> | | 贪婪三角测量 **(gp3.h)** |
| **设置** | setInputCloud(pcl) | 输入点云 |
| setMu(2.5) | 平滑度 |
| setNormalConsistency(bool) | **保持**法线一致 |
| setSearchMethod(tree) | 搜索算法 |
| setSearchRadius(radius) | 三角边长 |
| setMaximumNearestNeighbors(nnn) | 近邻数量 |
| setMaximumSurfaceAngle(x) | 法线夹角**限制** |
| setMaximumAngle / setMinimumAngle(x) | 三角内角**限制** |
| **方法** | reconstruct(mesh) | 输出**表面重建结果** |

## 可视化visualization：

namespace：visualization

|  |  |  |
| --- | --- | --- |
| PCLVisualizer | | 点云可视化 **(pcl\_visualizer.h)** |
| **方法** | addPointCloud(pcl, id) | **添加**点云 |
| addPolylineFromPolygonMesh(mesh, id) | **添加**多边形 |
| addPolygonMesh(mesh, id) | **添加**多边形网格 |
| resetCamera() | 重置**相机** |
| spin() | **调用**交互**器，进入事件循环** |

# 李群sophus：

**package：Sophus**

**lib：Sophus::Sophus**

**namespace：Sophus**

|  |  |  |
| --- | --- | --- |
| 群 (集合 + 运算) | **封闭性** |  |
| **结合律** |  |
| **幺元** |  |
| **逆** |  |

|  |  |  |
| --- | --- | --- |
| **李群** | **定义** | **具有连续性质的群** |
| **李代数** | **定义** | **单位元附近的正切空间** |
| **封闭性** |  |
| 双线性 |  |
| **自反性** |  |
| **雅可比等价** |  |

****

扰动模型：全0李代数，添加扰动量1e-4后，转换为李群，左乘于李群

欧拉角 (rpy角)：旋转顺序ZYX，万向锁问题 (奇异性问题 - 绕y旋转后，x轴、原z轴共线)

|  |  |
| --- | --- |
| **核函数** |  |
| **绝对误差**  **(位姿李代数的均方根误差)** |  |
| **相对位姿误差** |  |

## 缩放rxso3.hpp：

|  |  |  |
| --- | --- | --- |
| Sim3f / Sim3d(scale, rota, trans) | | **实例化缩放旋转群** |
| **属性** | matrix() | **变换矩阵** |
| scale() | **尺度信息** |
| so3() | **旋转信息** |
| quaternion() | **非单位四元数** |
| **运算** | \* x | 矩阵乘法 |

## 相似sim3.hpp：

|  |  |  |
| --- | --- | --- |
| Sim3f / Sim3d(scale, rota, trans) | | **实例化相似变换群** |
| **属性** | matrix / matrix3x4() | **变换矩阵** |
| RxSO3 rxso3() | **缩放旋转信息** |
| **运算** | \* x | 矩阵乘法 |

## 欧氏se3.hpp：

|  |  |  |
| --- | --- | --- |
| SE3f / SE3d(rota, trans) | | **实例化特殊欧氏群** |
| **属性** | matrix / matrix3x4() | **变换矩阵** |
| SO3 so3() | **旋转信息** |
| translation() | **平移向量** |
| **运算** | \* x | 矩阵乘法 |
| **函数** | log() / ::exp(vec) | **李代数 ↔ 李群** |
| ::hat(vec) / ::vee(mat) | **反对称矩阵 ↔ 向量** |
| ::leftJacobian(vec) | **雅可比矩阵** |

## 正交so3.hpp：

|  |  |  |
| --- | --- | --- |
| SO3f / SO3d(rota) | | **实例化特殊正交群** |
| **属性** | matrix() | **旋转矩阵** |
| unit\_quaternion() | **单位四元数** |
| **运算** | \* x | 矩阵乘法 |
| **函数** | log() / ::exp(vec) | **李代数 ↔ 李群** |
| ::hat(vec) / ::vee(mat) | **反对称矩阵 ↔ 向量** |
| ::leftJacobian(vec) | **雅可比矩阵** |