# 矩阵Eigen：

**using namespace Eigen;**

## 内核Core：

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

|  |  |  |
| --- | --- | --- |
| Vector<dtype, nr> / Matrix<dtype, nr=-1, nc=-1> | | **向量 / 矩阵** |
| 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 |
| block(ir, ic, nr, nc) | **切片** |
| 运算符 | << x0, x1, ... | 写入元素 |
| \* x | 矩阵乘法 |
| 变换 | transpose() | 派生转置 |
| **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** |
| 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 |

|  |  |  |
| --- | --- | --- |
| 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** |

## 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** |
| 实例方法 | **eigenvalues()** | **特征值** |
| **eigenvectors()** | **特征向量** |

## 几何Geometry：

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

|  |  |  |
| --- | --- | --- |
| Quaterniond(w, x, y, z) | | **四元数 (旋转)** |
| **实例方法** | coeffs() | **虚部、实部** |
| prerotate / rotate(rotation) | **原地旋转 (绝对/相对)** |

|  |  |  |
| --- | --- | --- |
| Isometry3d::Identity() | | **等距变换 (齐次坐标系)** |
| **实例方法** | 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)** |

# 格式化fmt：

**lib：fmt::fmt**

# 图像opencv2/opencv.hpp：

**C++：**

**lib：${OpenCV\_LIBS}**

**namespace cv;**

Python：复制Lib\site-packages\cv2\cv2.pyd到上一级

Gray = 0.299 \* R + 0.587 \* G + 0.114 \* B

|  |  |
| --- | --- |
|  | |
| 色调 H |  |
| 饱和 S |  |
| 亮度 V |  |

|  |  |
| --- | --- |
| 光流法： | |
| 假设 | 连续两帧图像之间，目标的像素亮度不改变；相邻的像素之间有相似的运动 |
| 意义 | 像素的位移值，两通道分别表示x,y |

## 三维calib3d.hpp

|  |  |  |
| --- | --- | --- |
| undistort(src, dst, cameraMatrix, distCoeffs) | | 图像去畸 |
| 形参 | Mat\_<float> cameraMatrix(3, 3); | 相机内参 |
| Mat\_<float> distCoeffs(5, 1); | 畸变参数 |

|  |  |  |
| --- | --- | --- |
| StereoSGBM | | 双目图像 |
| 方法 | ::create(0, 96, 9, 8\*9\*9, 32\*9\*9, 1, 63, 10, 100, 32) | 初始化 |
| compute(left, right, dst) | 计算视差 (CV\_16S) |

## 内核core.hpp：

四通道：Blue, Green, Red, Opacity

|  |  |  |
| --- | --- | --- |
| Rect | | 2D切片 |
| 形参 | x, y, w, h | 边界框 |
| pt1, pt2 | 对角线 |
| 方法 | tl / br() | 边界点 |
| area() | 面积 |
| contains(pt) | 判断包含点 |

|  |  |  |
| --- | --- | --- |
| Size | | 2D尺寸 |
| 形参 | w, h | 初始化 |

|  |  |
| --- | --- |
| Ptr<dtype> | 引用计数指针 |

|  |  |  |
| --- | --- | --- |
| Mat / Mat\_<dtype> | | 矩阵 |
| 形参 | rows, cols, type, s | 2D矩阵 |
| ndims, sizes, type, s | nD矩阵 |
| 运算 | << x0, x1, ... | 写入元素 |
| 属性 | rows / cols / channels() | 形状 |
| dims / size | 维度 |
| total() | 元素数 **(需乘通道)** |
| elemSize() | **元素**占用空间 |
| type / depth() | 类型 (位于**core/hal/interface.h**) |
| empty() | 为空 |
| 拷贝 | clone() | 深拷贝 |
| convertTo(dst, rtype, w, b) | 类型转换**，并线性变换** |
| 访问 | ptr<T>(0) / ptr<T>(r, c) | 数据 **/** 像素头指针 |
| at<T>(idx[]) | 像素 |
| row / col(i) | 单行 |
| rowRange / colRange(s, e) | 多行 |
| (rect) | 任意矩形 |
| 填充 | setTo(value, mask) | 定值 |

|  |  |  |
| --- | --- | --- |
| 2D操作： | | |
| 常规操作 | print(src) | 输出 |
| flip(src, dst, flipCode) | 翻转 |
| 数值计算 | repeat(src, ny, nx) | 重复 |
| LUT(src, lut, dst) | 查表取值 |
| 形状修改 | copyMakeBorder(src, dst, t, b, l, r, borderType=0, value) | 边界填充 |
| hconcat / vconcat(vector<cv::Mat> src, dst) | 图像拼接 |
| transpose(src, dst) | 转置 |

|  |  |
| --- | --- |
| 内存管理： | |
| copyTo(src, dst, mask) | 拷贝 |
| convertFp16(src, dst) | CV\_32F ↔ CV\_16S |

|  |  |
| --- | --- |
| 数值计算： | |
| add / subtract / multiply / divide /  bitwise\_and / bitwise\_or / bitwise\_xor(src1, src2, dst) | 逐元素运算 (可用运算符) |
| addWeighted(src1, alpha, src2, beta, gamma) | 逐元素加权 |
| max / min(src1, src2, dst) | 逐元素最值 |
| sum / mean(src) | 逐通道聚合 |
| sqrt(src, dst) / pow(src, power, dst) | 幂运算 |
| exp / log(src, dst) | 对数运算 |
| polarToCart / cartToPolar(x, y, magnitude, angle) | 极坐标转换 |
| norm(src) | 聚合 |
| hasNonZero / countNonZero(src) | 非零聚合 |
| flipND(src, dst, axis) | 翻转 |
| normalize(src, dst, alpha, beta, norm\_type) | 规范化 (默认单位化) |

|  |  |
| --- | --- |
| 形状修改： | |
| merge(vector<cv::Mat> src, dst) | 通道拼接 |
| split(src, vector<cv::Mat> dst) | 通道拆分 |
| rotate(src, dst, rotateCode) | 旋转 |
| transposeND(src, vector<int> order, dst) |  |

## 交互highgui.hpp：

|  |  |  |
| --- | --- | --- |
| **图像编码：** | | |
| **.png** | **CV\_8U / CV\_16U** | **无损压缩，支持BGRA** |
| **.jpg / .jpeg** | **CV\_8U / CV\_16U** | **有损压缩** |
| **.tiff** | **CV\_8U / CV\_16U / CV\_32F / CV\_64F** | **支持多图像** |
| **.exr** | **CV\_32F** | **支持压缩** |
| **.hdr** | **CV\_32F** | **无压缩** |

|  |  |  |
| --- | --- | --- |
| imread(filename, flags) | 打开图像返回BGR图像 | |
| imwrite(filename, img) | 导出图像到文件 | |
| imshow(winname, mat) | 显示图像 | |
| waitKey(delay) | 0 | 无限期等待键盘输入 |
| int | 等待对应ms后关闭 |
| destroyAllWindows() | 关闭所有展示窗口 | |

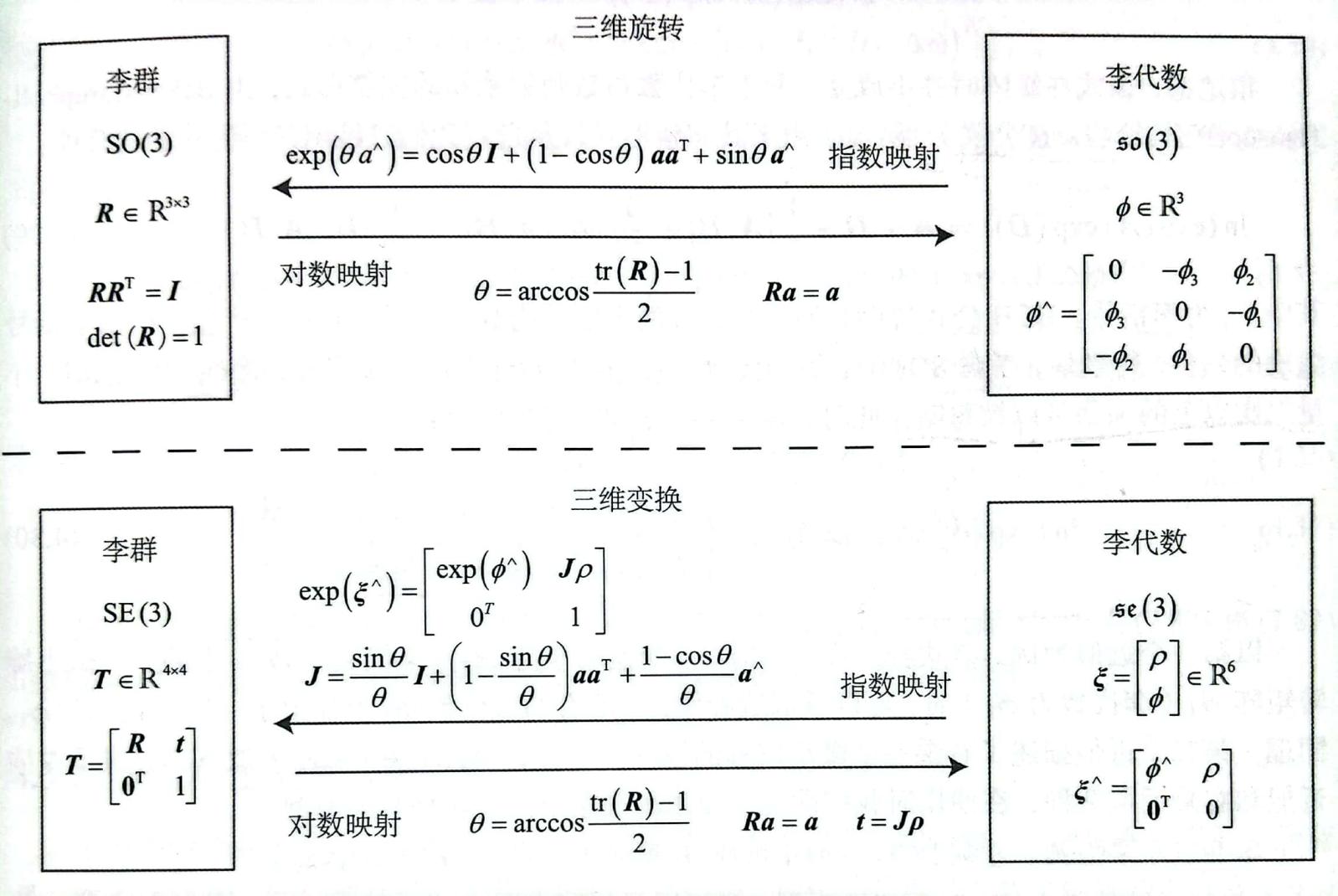
## 处理imgproc.hpp：

|  |  |
| --- | --- |
| resize(src, dst, cv::Size size) | 图像重置尺寸 |

# 可视化pangolin：

**lib：${Pangolin\_LIBRARIES}**

# 李群sophus：

****

**using namespace Sophus;**

**lib：Sophus::Sophus**

扰动模型 (求导)：全0李代数，添加扰动量1e-4后，转换为李群，左乘于李群

## 正交so3.hpp：

|  |  |  |
| --- | --- | --- |
| SO3d(rota) | | **实例化特殊正交群 (旋转矩阵 / 四元数)** |
| **属性** | matrix() | **旋转矩阵** |
| **运算符** | \* 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** |

## 欧氏se3.hpp：

|  |  |  |
| --- | --- | --- |
| SE3d(rota, pos) | | **实例化特殊欧氏群 (旋转矩阵 / 四元数)** |
| **属性** | matrix() | **旋转矩阵** |
| **运算符** | \* 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** |