1 代码及功能文件对应说明

|  |  |  |
| --- | --- | --- |
| 编号 | 文件名 | 对应功能 |
| 1 | CloudRegister.cpp | 入口文件，封装对外提供的接口 |
| 2 | CADModel.cpp | 图纸解析及3D建模，包含输出各项排序功能 |
| 3 | CloudSegment.cpp | 点云粗分割及粗配准 |
| 4 | TransformOptimize.cpp | 点云精配准,平面方程计算,图纸关联 |
| 5 | CloudBorder.cpp | 点云门洞边框检测 |
| 6 | CloudRefine.cpp | 根据墙面及门洞边框进行多余部分点云裁剪精修 |
| 7 | CalcBayAndDepthMeasure.cpp | 开间以及进深测量 |
| 8 | CalcCorner.cpp | 阴阳角测量 |
| 9 | CalcHoleMeasure.cpp | 门洞测量 |
| 10 | CalcNetHeight.cpp | 净高及极差计算 |
| 11 | CalcRootFlatness.cpp | 地面平整度计算 |
| 12 | CalcSquareness.cpp | 方正度计算 |
| 13 | CalcWallFlatness.cpp | 墙面平整度 |
| 14 | CalcWallVerticality.cpp | 墙面垂直度 |
| 15 | CalcHoleMeasure.cpp  funHelper.cpp  GeometryUtils.cpp | 底层基础计算函数 |

2 接口说明

2.1 输入接口说明

图纸解析细节参见前期提供的示例说明，针对各个子项输入顺序，除地面输入顺序为逆时针外，

墙面，门洞，横梁输入顺序均为顺时针，以正对方向的左下角为起点，参见如下示意图

1|————————|2

| |

| |

0|————————| 3

2.2 输出接口说明

输出各子项边框均调整为以正对方向的左下角为起点，顺时针输出，与上图相同

墙面顺序以最大采光面为起点，顺时针输出

各测量接口设置参数说明可参见CloudRegister.h说明，可调整部分参数均已设置默认值

输出默认由三部分组成

1 每项测量的具体测量数字calcMeassurment\_t.value

2 每项测量计算使用的点云范围框 calcMeassurment\_t.rangeSeg，以多个线段方式构成

3 std::vector<seg\_pair\_t> 每项测量指标可视化辅助输出线段信息

3 各模块设计说明

3.1 CADModel

该模块与最终输出排序相关，点云各项输出顺序在完成与图纸的映射后，与图纸顺序完全保持，例如墙面门洞边框顺序，最大采光面顺时针排序，如果后期需要调整各项输出顺序请在该模块完成相关排序功能，点云输出自动调整。reSortWall函数对应现在的墙面输出排序功能，地面及顶部对应边框排序以墙面排序为准。

3.2 各测量指标计算

各测量指标计算方式以接口为准，本文档只描述通用计算部分逻辑和底层通用函数

为了计算提速以及逻辑清晰，所有计算指标均使用以点云检测出的墙/门洞/地面/顶部来计算获取需要计算的点云范围边框，然后再只获得原始的局部点云根据不同的计算方式进行细节指标计算。

以开间计算为例，通过点云地面边框，通过相互投影的方式计算二二平行的且有overlap线段完成开间多个区域划分的计算(参见下面的示意图，投影计算的A1A2,B1B2即为一个计算区间)，随后将门洞门框投影到地面边框，计算出墙面在X或者Y方向有点云的区域范围，以此为基础再计算时间点云范围框，最终使用过滤出的局部点云块完成计算。

A1 A2

------------------------ WallA

| |

| |

--------------------------- WallB

B1 B2

常用基础函数说明：

groupDirectionIndex(horizenSeg, rootBorder, vecVerticalIndex, vecHorizenIndex);

将边框线段rootBorder分为与horizenSeg平行或者垂直的二组数据

std::tie(hasOverlap, s1Pt, e1Pt, s2Pt,e2Pt) = calcOverlap(toSeg,calcSeg);

计算二个平行的线段是否有ovelap,如果hasOverlap为true，则后面4个点有效，分别对应上图中的A1A2B1B2四个点

auto vecPt = createRulerBox(seg, index, thickness, calcHalfPara \* 2);

计算获取局部点云3D尺子边框, seg为尺子边框的中心线， index代表退化的方向索引，例如一面墙的XY二个轴，如果X轴是主要变化方向，那个Y轴就是退化轴，index为1反之为0，thickness退化轴的范围及墙的厚度，calcHalfPara \* 2，尺子的宽度，长度已经由前面的中心线指定

item.rangeSeg = calcBoxSegPair(vecPt);

根据计算出的顶点创建可视化使用的线段

auto vecTmp = getRulerCorners(vecPt);

auto pCloud = filerCloudByConvexHull(pWall, filerPt);

根据尺子过滤出计算需要的点云

3.3 点云粗分割及粗配准

3.4 点云精配准

getModelPlaneCoeff(cadModel, center);

计算每个墙面方程及根据要求修改平面方程法向，满足计算点到平面的凹凸符号定义

matchCloudToMode();

根据粗对齐的结果完成和图纸的一一关联

downSampling();

optimize(transform);

使用降采样后的点云计算全局变换最优解，地面点云不参与计算，策略为顶部优先

transformCloud(transform);

根据求解出的变换矩阵，将原始所有点云进行变换，统计变换后地面点云到平面的距离中值，重新计算垂直方向的移动T，完成底部对齐

//get plane coeff with input Cloud

Eigen::Vector3d newCenter = transform.block<3, 3>(0, 0) \* center + transform.block<3, 1>(0, 3);

getCloudPlaneCoeff(newCenter);

根据变换后的点云位置重新计算每个平面方程