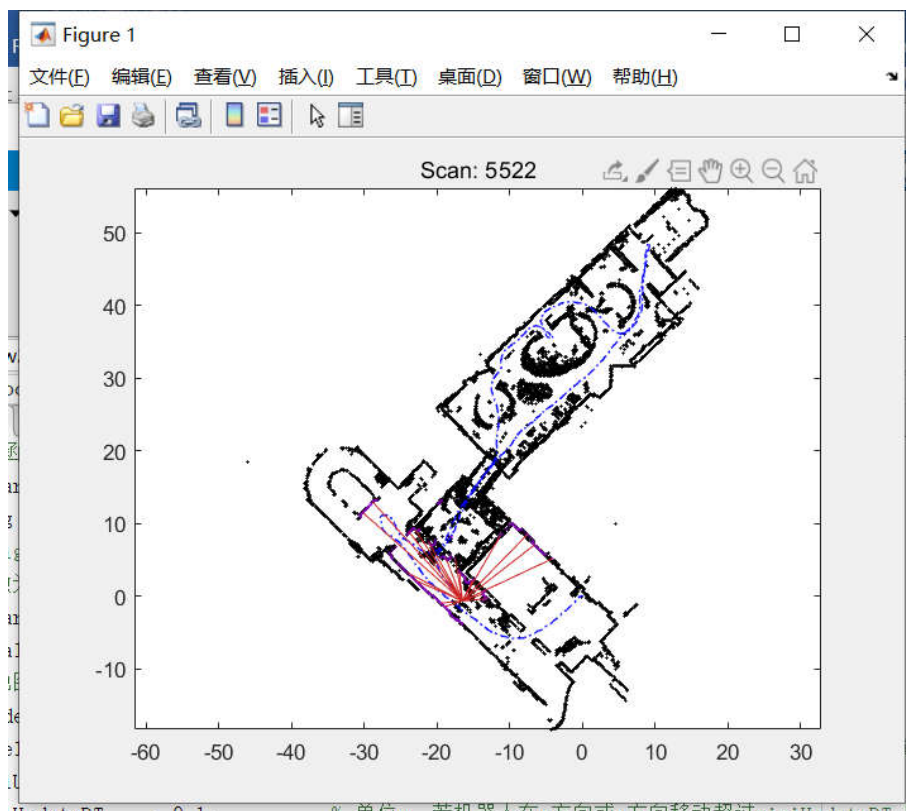


→Matlab 2D SLAM 理清程序逻辑、调试完成


| | | |
|---------------|---|---|
| 1. 参数设置 | <ul style="list-style-type: none"> ➢ 搜索窗添加边界的尺寸 ➢ 像素尺寸 一个像素代表的长度 ➢ 添加的关键帧的参数(运动滤波器) | <ul style="list-style-type: none"> ➢ CSM 参数设置 搜索分辨率 FastResolution ➢ 闭环参数设置 搜索分辨率 BruteResolution |
| 2. 整体初始化 | ➢ 全局 map 初始化 | |
| 3. 扫描帧的收集和预处理 | <ul style="list-style-type: none"> ➢ 提取单个帧 ➢ 只取一定 range 范围内的扫描点 | |
| 4. CSM | <ul style="list-style-type: none"> ➢ 提取 scan 周围可能的局部地图 ➢ 局部地图生成栅格地图 | <ul style="list-style-type: none"> ➢ 恒定速度模型提供预测初始值 ➢ Hill-Climbing 搜索算法 |
| 5. 关键帧处理 | <ul style="list-style-type: none"> ➢ 判断关键帧 ➢ 寻找和上一个 keyscan 的新点, 然后添加 keyscan, 添加 connection 用于之后的图优化 | |
| 6. 回环 | <ul style="list-style-type: none"> ➢ 检查是否进行回环 >10 个关键帧 并且 travel 距离需要大于 10m ➢ 确定回环检测进行的关键帧的范围 轨迹长度小于 50 并且找到一个与当前帧距离较近的关键帧 ➢ 非常暴力 -r-x-y 的 CSM 基本这个回环是费的 没有办法使用 | |

效果图



→在 2D SLAM 加入需要加入的内容:体素滤波、分支定界 CSM、两次三双差值、submaps 的搜索结构、回环

| | | |
|---------------|--|--|
| 1. 参数设置 | <ul style="list-style-type: none"> ➢ 搜索窗添加边界的尺寸 ➢ 像素尺寸 一个像素代表的长度 ➢ 添加的关键帧的参数(运动滤波器) ➢ Submaps 的参数设置 <input checked="" type="checkbox"/> | <ul style="list-style-type: none"> ➢ CSM 参数设置 搜索分辨率 FastResolution ➢ 闭环参数设置 搜索分辨率 BruteResolution |
| 2. 整体初始化 | ➢ 全局 map 初始化 keyframe 换成 cartographer 的 trajectory_node submaps <input checked="" type="checkbox"/> | |
| 3. 扫描帧的收集和预处理 | <ul style="list-style-type: none"> ➢ 提取单个帧 ➢ 只取一定 range 范围内的扫描点 ➢ 体素滤波 留下自适应滤波之后做 <input checked="" type="checkbox"/> | |
| 4. CSM | <ul style="list-style-type: none"> ➢ Cartographer submap 匹配 <input checked="" type="checkbox"/> ➢ 提取 scan 周围可能的局部地图 ➢ 局部地图生成栅格地图(HITS)地图 确定很明显只要扫到了就确定存在的地图, 在很多现实场景中并没有概率分布的概念, 一切都是一个确定的值, 一旦有动态环境出现, 全部完蛋。相当于没有利用好每一个扫描数据, 只利用新数据 ➢ 换成栅格地图(odds) | <ul style="list-style-type: none"> ➢ 恒定速度模型提供预测初始值 ➢ Hill-Climbing 搜索算法在局部情况下要快得多 -r-x-y 换成 r 层 <input checked="" type="checkbox"/>或者多分辨率 ➢ 精细化 CSM 换成优化匹配 |

| | |
|----------|---|
| 5. 关键帧处理 | <ul style="list-style-type: none"> ➤ 判断关键帧 ➤ 寻找和上一个 keyscan 的新点, 然后添加 keyscan, 添加 connection 用于之后的图优化 阅读 Carto 的 submap 内容在全局地图中进行配置  |
| 6. 回环 | <ul style="list-style-type: none"> ➤ 检查是否进行回环 >10 个关键帧 并且 travel 距离需要大于 10m 换成 cartographer 的 submap 插入结束进行回环的方式 ➤ 确定回环检测进行的关键帧的范围 轨迹长度小于 50 并且找到一个与当前帧距离较近的关键帧 ➤ 非常暴力 -r -x -y 的 CSM 基本这个回环是费时的 没有办法使用 分支定界的进行回环检测 |

具体实现(Cartographer)

→Matlab 往 C++ 转换, 进行一次重构只处理激光数据, 加入 Ceres 优化和线程池还有 OpenCV 点云显示

1. Ceres 学习

整体流程

| | |
|------------------------------|---|
| 1. Construct a Cost Function | The important thing to note here is that operator() is a templated method, which assumes that all its inputs and outputs are of some type T . The use of templating here allows Ceres to call CostFunction::operator<T>() , with T=double when just the value of the residual is needed, and with a special type T=Jet when the Jacobians are needed. |
| 2. 创建 Problem | Problem problem; |
| 3. 计算 Derivative 并加入 Problem | Problem problem; problem.AddResidualBlock(new AutoDiffCostFunction<F1, 1, 1, 1>(new F1), NULL, &x1, &x3); <F1,1,1,1> 第一个数字是输出的维度, 剩下两个是优化的维度(x1 和 x3), F1 是优化方程 NULL 是核函数 |
| 4. Options 的配置 | Solver::Options options; options.max_num_iterations = 100; options.linear_solver_type = ceres::DENSE_QR; options.minimizer_progress_to_stdout = true; |
| 5. 进行优化 | Solve(options, &problem, &summary); |

Tips: operator () 函数要有一个 const 后缀, operator 内的常数数字要在 T()内

Options 参数 <https://blog.csdn.net/DumpDoctorWang/article/details/84890792>

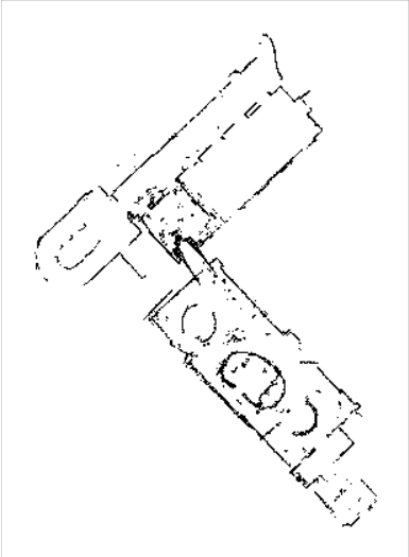
2. C++ 语法细节

| | |
|---|--|
| 1. 智能指针 shared_ptr | 普通指针的转化只能通过显式的方式 make_shared<int> 是一个模板 后面要有一个<> .release(); |
| 2. CMakeLists.txt | Add_executable () 要把文件所有.cpp 文件加进来 不然.cpp 的内容不会进行编译 |
| 3. Set unordered_set 和 Eigen 冲突 | 学 Cartographer 使用 bitset 进行 set 的存储 |
| 4. Mapbuilder | |
| 5. Vector 拼接 | vector a; vector b; a.insert(a.end(),b.begin(),b.end()); |
| 6. C++ 类型转换 | https://www.cnblogs.com/chenyangchun/p/6795923.html |
| 7. C++ 计时 chrono | https://blog.csdn.net/DumpDoctorWang/article/details/81781227 https://blog.csdn.net/bao_bei/article/details/77162321 |
| 8. C++ 多态 | |
| 9. 调试上 VS 好用, Linux 快 | https://stackoverflow.com/cn/q/10202721 VS /O2 和/RCT1 出现冲突 |
| 10. Can not declare the member function to have static linkage 也就是说 static member function 在.cc 中进行定义时不需要再添加 static 关键字 | |
| 11. 过于 LNK 2001 静态成员上的错误: 静态成员需要进行类外定义, 类内的那个只是一个声明 | |
| 12. 之后 undefined reference to WinMain 16 是因为 静态成员的类外定义没有和静态成员的定义放在一个文件内 之前一个放在.h 一个放在.cpp 里了 | |

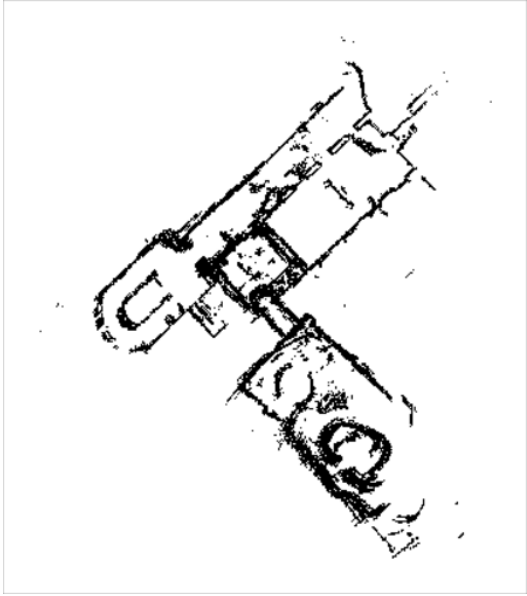
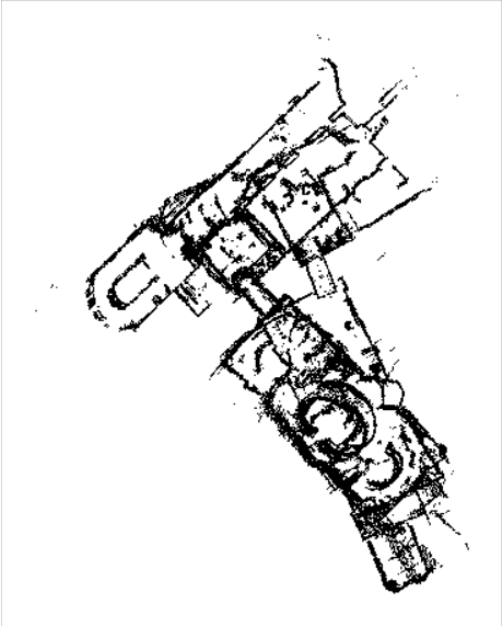
3. 实现细节

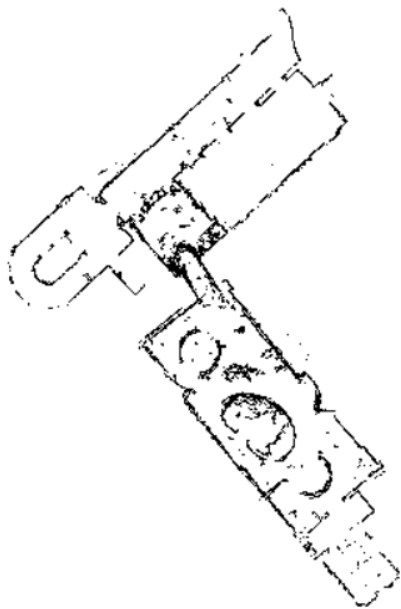
V01

| | |
|--|---|
| 1. VoxelFilter 结合 unordered_set 和 bitset 来完成 | ➤ Eigen::Array2d 和 set 和 unordered_set 犯冲 移位操作<<完成。 2020.3.20 发现 |
| 2. OpenCV 点云显示 | ➤ DrawPoint 函数来写 |
| 3. GetAllData () | ➤ 信息 txt 文件 应该放在可执行文件的目录下 |
| 4. ComputeOccGrid->distanceTransform | distanceTransform+ threshold 实现比较繁琐 |

| | |
|------------------------|---|
| | <p>所以直接采用自己写的 self-distanceTransform</p> <p>Mat::copyTo 有很多坑 第二个参数一定要是一个 CV_8UC1 不然内存泄漏 最后用 CV_16FC1 代替，并且写了一个和函数将 self_distanceTransform 实现 Mat 访问元素时的指针需要多注意，如果类型步匹配也会出问题</p> <p>性能差了 3700 倍还是使用 opencv 的函数了(实现的细节上水平差太多了)</p> <p>Matlab round +1 因为 matlab 矩阵从 1 开始计数</p> <p>C++ 从 0 开始计数</p> |
| 5. Kdtree | <p>报错在.obj 中有定义 LNK2005 因为没有把声明和定义放在了一起导致的</p> <div data-bbox="633 385 1042 940"></div> <p>更改后效果图：回环误差加大，猜测是之后添加 voxel_filter 让地图计算不是那么准导致的，毕竟这里用的 voxel_filter 没有取质心，对于地图的形状还是有所改变。会在向 kdtree 插入 node 的时候偶尔出现卡顿，kdtree 更适合在已有地图上进行定位使用，而不是在构建地图的时候存储地图点。但是在搜索时确实是可以加速点的搜索。</p> |
| 6. Drawer OpenCV 2d 显示 | DrawerPoint TESTPASS |

效果分析

| | |
|--|--|
| 效果 3000 帧 | 效果 5500 帧 |
| <div data-bbox="106 1265 635 1859"></div> | <div data-bbox="665 1270 1168 1895"></div> |
| 修改代码后的效果 | |



原本错误的地方：

1. 做 DownHill 搜索的时候，旋转搜索的时间，直接在 LocalMap 系中进行了，而没有在世界坐标系下进行。 第一版
2. 一种是先做一个 Δ 的微小运动，然后在做 $guess_pose$.
一种是先做一个 $guess_pose$ 然后做一个微小运动。
区别是一个 Δ 的运动矩阵左乘一个是右乘。

但是搜索类算法的想法是直接找一个最合适的 $guess_pose$ 和微小运动结合成的运动矩阵，也就是最合适的 $xy\ theta$ 三个参数。

错误原因：最开始的一半想要寻找一个微小运动然后和 $guess_pose$ 结合算出结果(也就是先把点从 $guess_pose$ 坐标投射到微小运动的坐标内，然后寻找最优微小运动参数)，但是因为这种矩阵乘法表现在运动参数($xy\theta$)上时，是一个非线性的相加。所以最后在返回值的时候我让两个运动的参数直接相加的方式，必然会导致平移方面的误差(xy ；因为是 2d 的原因)。也就是说最后最优的参数是从两个运动矩阵相乘后，得到的参数值，而不是一个粗暴地将两个运动矩阵的参数相加。

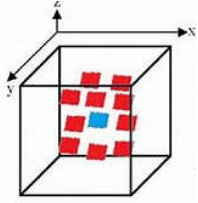
正确的方法：直接在 $guess_pose$ 的三个参数上进行搜索，这会让运动在李代数上做每次不等值的变换，但是在 $R^3(x\ y\ \theta)$ 上做等值的变换。省掉了 $guess_pose$ 往微小运动坐标系的变换，找到的直接就是最优参数。

这里可以说也是一个对于 Cartographer 的小改进 Cartographer 用的是后者，我用的是前者

结果分析&后期改进

1. 在 Matlab 里面实现的时候，可以发现 HillClimb 是一种对变化非常敏感的搜索方式。但是速度快!! HillClimb 可以作为一种 subpixel 的方式来提高搜索的精度，猜测是可以代替 Ceres 来进行搜索的。而且 HillClimb 在对 odd 地图的适应性上比 ICP 要强。
2. 从结果可以看出来这个里程计的精度已经算是不错的，因为每一次点的累计只是累计了新的点。所以误差增加很慢。猜想一旦地图中出现一个绝对路标进行矫正，误差不会太大。
3. 为了保证项目是一个 SLAM 系统，而不是一个里程计，在此基础上添加后端优化的过程
4. HillClimb 最致命的问题是如果这个地图中存在移动的物体，那么因为算法对变化的敏感性，那么很容易出问题。所以还是决定先用 Odd 来代替这个地图，然后想办法把 HillClimb 和 odd 地图结合起来，降低对动态地图的敏感性。
5. 在进行回环检测的时候 Cartographer 的分支定界明显是比 HillClimb 暴力匹配更好的方法。一个是全局最优一个是局部最优，而且在速度上会有很大的差别。但是 Cartographer 的回环检测也是可以再优化一点点的，关键在于 Cartographer 在分支定界的时候要保存一个现有最大结果，这在整数规划的时候是有体现的。但是在 Cartographer v1.0 版本中并没有体现。

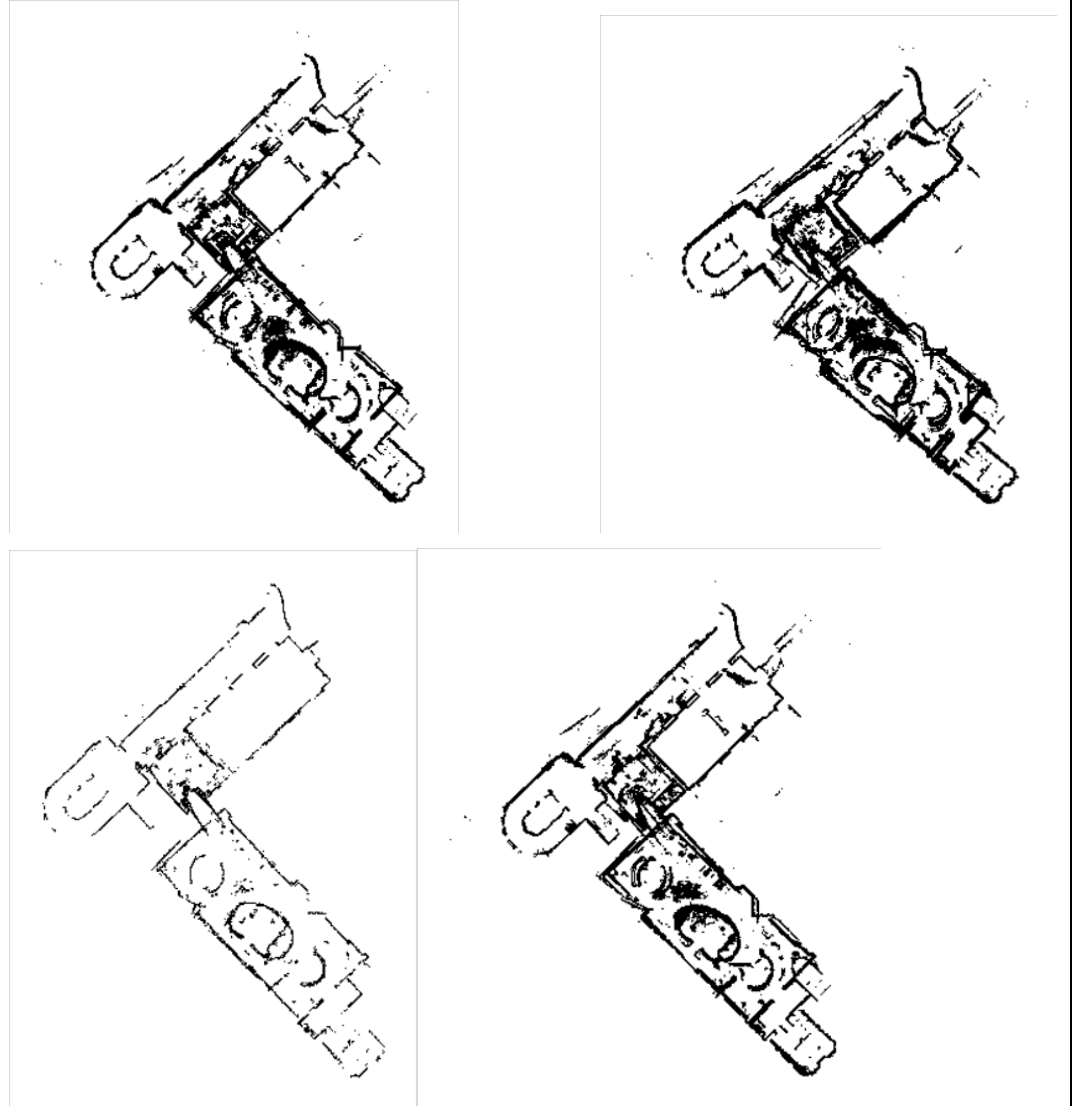
V02

| | |
|-------------|--|
| 1. 重构代码 | 各个类支持各个功能 |
| 2. 添加占据栅格地图 | <ul style="list-style-type: none"> ➤ PGMBuild 在构造函数时，进行图的初始化。 ➤ GrowAsNeeded() 使用 Eigen::AlignedBox 来进行拓展 |
| |  <p>AlignedBox3d 的例子 .extend 用于扩展 .min .max 提取最大最小值</p> |
| | <ul style="list-style-type: none"> ➤ Submap() 拼接 每一个子图有一个 origin 当做子图的坐标，用于之后检索每一帧和子图的位置。然后每一个子图都有一个 AlignedBox2i 来确定这个子图在整体子图的位置 AlignedBox .min 和 .max 都是直接可以以离散的世界坐标进行保存 RayCast 划线 ApplyLookUpPoint() 一个概率查询表 表的预计算 Odd 和 概率数值转化 Debug: 1. 在测试 Insert 的时候出现问题，问题就是估计出来的数值和实际使用的值是一个相反的关系，经过检查之后， |

| | |
|---------------|--|
| | <p>发现原因如下：在处理新一帧时，在匹配的时候，做的操作可以理解把新的一帧的坐标进行旋转平移，和原本的地图进行匹配。所以这样得出的微小运动的估计值和实际的微小运动运动是一个相反的值。</p> <p>测试通过☑</p> <p>2. 集成到 main.cpp 中进行估计</p> <p>A . pixelSize =0.1 效果没有问题 pixelSize = 0.05 结果螺旋爆炸。猜测:离散化的时候有的参数是直接赋的数值。 原因：SearchWindow 的 pixelSize 参数在 GenerateRotDiscre 的时候用来离散化了，就影响了离散点的坐标。</p> <p>B . 第二个问题是会在 62 到 64 帧的时候出问题，运动会超过预设值。因为，一直没有往内部进行插值，也就是说 MotionFilter 设置的太大了。原因：出现了 1/0 的情况</p> <p>C . 虚拟机出问题无法进行 Ceres 测试进行精匹配</p> <p>用了大量的时间，在 debug 这里，现在是明白里面的坑了</p> <p>后期改进：需要把 CSM::Match 函数里面的 TODO 完成！</p> <div data-bbox="489 555 925 1055"></div> <p>CSM 对于滤波的容忍性更好，不会因为滤波漏掉某些重要的地图点，但是目前看起来这个地图不够精准。</p> <p>Ceres 的加入可能有问题，导致整个系统并没有起色。</p> |
| 3. 多分辨率搜索地图 | <p>➤ 高分辨率往低分辨率的转换☑</p> <p>二维地图转一维的缘故，先处理 x 方向，然后多个 x 方向一起处理 y 方向，为了提高复用率往向前移动一个格子的时候里面有很多值都没有改变，这就是 Cartographeer 的一个实现的细节。实现的方式和我用 Matlab 实现的方式不一样实现的非常好。</p> <p>具体实现使用 SlideWindow 运动 deque 来实现。</p> <p>具体细节：因为之后做分支定界的时候，大小确定的方式是[To,To + delta_T] 所以在做概率占据地图的时候需要考虑，整个模糊的方向的问题。</p> <p>小项目中采取 从左往右 从上往下取最大的方向。</p> <p>➤ 确定低分辨率时的搜索范围 ☑</p> <p>➤ 还需要优化一下地图存储的模式</p> <p>➤ Debug: 10 点了 发现问题是因为在构建地图的时候没有对地图进行扩大导致信息没有补充完全导致的</p> <p>1. 修改 BBM 的打分机制 全部都在 ScoreCandidate 函数中，把所有不满足条件的 candidate 的去掉</p> <p>2. 加上不合格的分数的下限 和一起实现</p> <p>3. 用 LoopResult 来传递最终结果的合理性</p> <p>Debug ：到这里发现 BBM 存在问题只能找到一定范围内的值 猜测还是 ShrinktoFit 的问题</p> <p>不是 ShrinktoFit 的问题 是在 Candidate 的问题。</p> <p>4. 修改 BBM MatchFullMAP 和小范围 map 时 GenerateWindow</p> <p>5. 修改 CSM::Generate 的问题</p> |
| 4. 分支定界 C++转换 | <p>和 CSM 的搜索不同。CSM 是一个在已有平移坐标系的情况下进行微调，但是在 BranchAndBound 的中为了更大地进行扫描，就要换成后者，再进行计算。还是有一些细节上要看一下修改的内容</p> <p>🚦 M2MPGM: 各种 get 函数 ☑</p> <p>🚦 CSM 添加 BBM (分支定界的部分) 还差递归的部分☑</p> |

- ✚ 打分函数对于外点 如何确保函数运作正常 ☑
这个问题在前端 submap 的时候是不会出现的，到了后端因为大位移的出现所以就需要调整一下
- ✚ CSM 的 BBM 里面的 ShrinktoFit 看一下 Cartographer 的转换。☑ **进一步缩小搜索范围**
ShrinktoFit 每一个 Node 对于 map 大小来确定能左右上下移动多少，来生成相对少量的 Candidates。通过看代码 ShrinkToFit 在大地图成功率更高。也就是说因为 ShrinkToFit 无法每次都精准的进行匹配，所以回环是有成功率的，而不是百分百成功。
- ✚ 还没有添加一个硬性的 threshold 已经添加好☑ 但是还没有具体设置值
- ✚ BBM submap 并没有使用拼接，来做总体的匹配，而是用每一个 Node 和 每一个 submap 来进行匹配。☑
- ✚ Debug
 1. 优化可能的 candidates 从 5s 到 1s 。**TODO: 还是有大量无用的值，下次优化的时候，计算点云的均值和方差，然后根据这个特点来进一步消除冗余 candidates。**
 2. 主要问题是在多层地图生成上，要在地图的左边和上边两边各生成 stride -1 个格子，第一次单 scan 已经通过。但是角度上差了 4 度。弧度差了 0.07 还是有问题。但是基本 8m 内的回环都可以检测到。但是 20m 的单帧回环就没有办法了。按理来说一个通了应该所有都可以，这里还需要再注意一下。全部可以跑通，之前的问题是因为 BBM_Loop 中每次切分成四个 candidates 的时候，划分 stride 的符号写错了， 应该和制作多层地图时取最大值的方向相对应。
 3. BBM 加速改成/o2 之后速度加速 300 倍。基本可用
 4. 已经在 surface 上逻辑推导画图证明基本可以用。

5. HillClimb 精匹配匹配



分析：

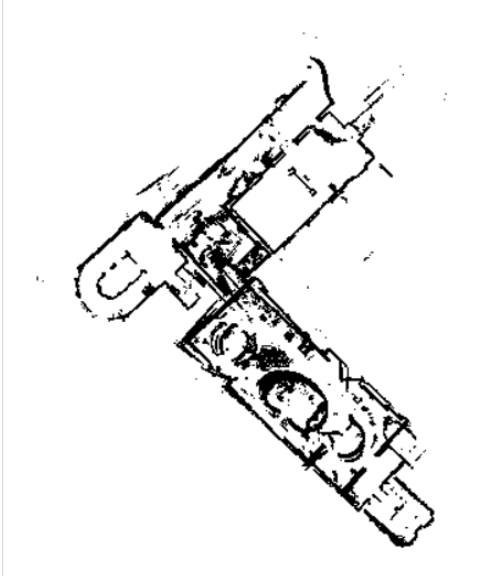
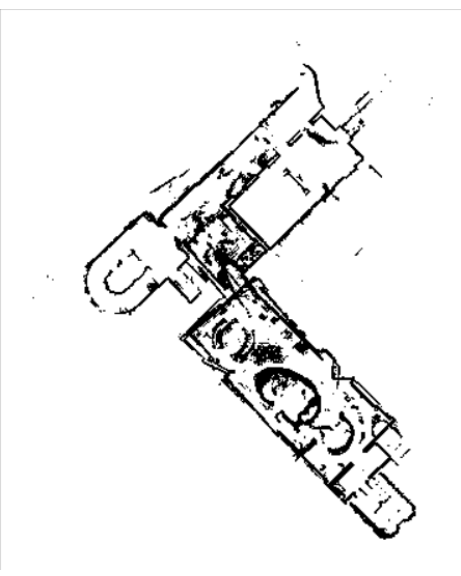
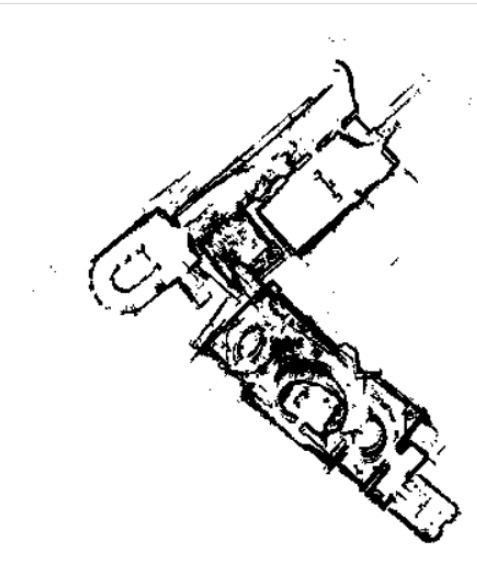
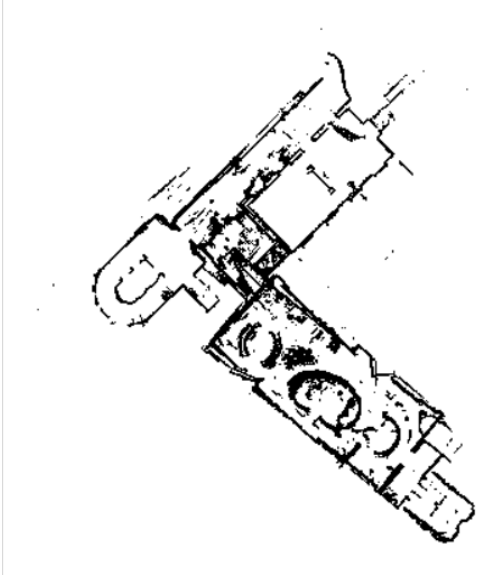
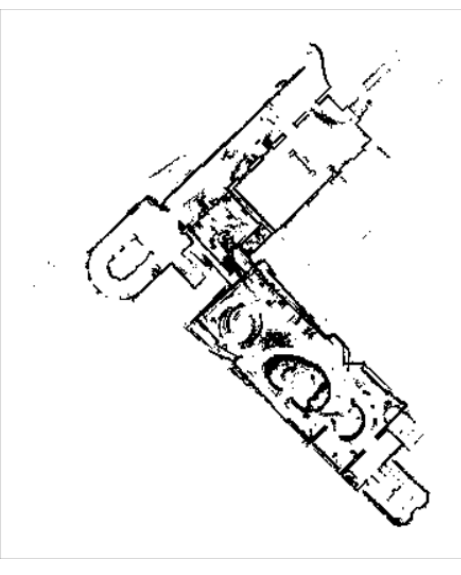
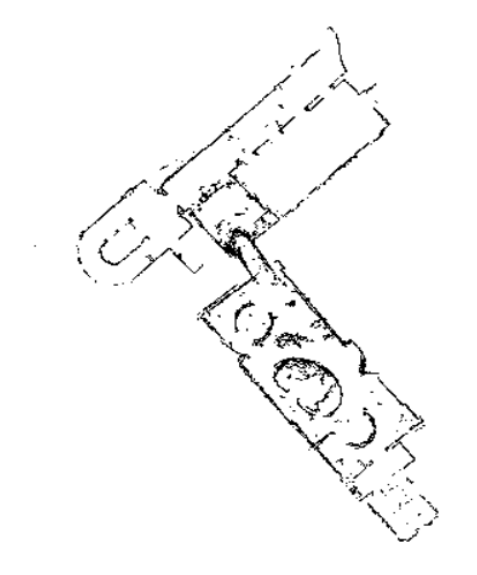
从结果来看精匹配的确是加大了计算量，V01 的 SLAM 在设计上更像是 LOAM 的 lidarMapping，这种从扫描点周围提取点的手法给 odometry 增加了很多的精度。

Submap 的设计在前端匹配上是局限性的 因为他看见的之后一段时间内的几帧，很容易出现这几帧都很像的问题，在这一点上 V01 的地图生成手段就会好很多。

| | |
|---------------------|---|
| | <p>这点的分析在 V01 改成 submap 结构的时候就有体现</p> <p>FinerEstimation 的精匹配可以进行的比例 0.4 因为精度更高的情况下合适的点会更少，对应的 BBM 的比例目前设置是 0.42</p> <p>后期把比例集成到函数的参数项。现在修改起来太麻烦</p> <p>目前的实验值从 10.345, -2.3789, 0.4441 给予初始值 10.2 -2.3 0.2</p> <p>CSM 第一次匹配 匹配到 10.4 -2.4 0.437654</p> <p>HillClimb 精匹配到 10.345 -2.3775 0.44398</p> <p>基本可以确定 HillClimb 的加速是可行的</p> <p>而且 HillClimb 的精匹配和 Ceres 在精匹配上有想同的特性 HillClimb 都是初值敏感的匹配算法，而且 HillClimb 可以直接使用离散性，不用因为算法的连续性来对地图进行插值。</p> <p>而且因为直接作用在概率栅格地图上，让这种算法和 CSM 一样是有方差和均值的</p> |
| 6. Ceres 优化 | <p>Bicubic interpolation</p> <p>https://blog.csdn.net/qq_24451605/article/details/49474113</p> <p>双三次差值 差值的范围是从 0 到 1 但是如果不知道导数值的话，就行只能从[0,1]范围左右分别多取一个点来进行一阶导数的近似。</p> <p>跑到 1000 帧，目前加 Ceres 和不加 Ceres 结果相同。</p> <p>通过查 summary.fullreport(),发现第一次迭代，优化的梯度就小于继续进行的下限，不再进行优化。</p> <p>目前不知道是不是代码写的有问题。还是说我设置的 0.05 的地图过于小，倒是 Ceres 没有办法发挥作用。</p> <p>效果不好栅格的离散化用 Ceres 差值做精匹配作用不大而且，而且似乎是选值的问题,结果并没有很好。主要原因还是 Carto 的里面用的是 floor 我用的是 round 会有一定的误差存在，需要进行测试。</p> |
| 7. PoseExtraPolator | <p>互补滤波：PoseExtrapolator</p> <p>使用的方法来自 https://www.zhihu.com/people/hu-xiao-dao-51/answers 无人机专栏--无人机姿态解算---适应互补滤波</p> <p>Roll Pitch Yaw 先得到前两个值，所以 yaw 会无法得到</p> |
| 8. 回环检测 | <p>Map 容器的使用 https://www.cnblogs.com/fnlingnzb-learner/p/5833051.html</p> <ol style="list-style-type: none"> 1. 修改 MapBuilder 的存储结构 添加 voxel_filter_scan 和 节点存储的内容 2. 修改 PGMBuild 的存储结构添加 index 来直接指向 MapBuilder 存储的 voxel_filter_scan 和节点的存储内容 3. 详细了解 Cartographer 的优化构建手法，权重设置 4. 复现 <p>回环策略：</p> <ol style="list-style-type: none"> 1. 距离上一次的回环检测到的位姿>6 米 2. 和子图平均位姿相差不到 3 米 3. 如果>6m 没有找到一帧，那么之后 20 个关键帧不再进行搜索 4. 检测第一个子图之后剩下的都不看 |
| 9. 线程池 | <p>https://wangpengcheng.github.io/2019/05/17/cplusplus_threadpool/</p> <p>https://www.cnblogs.com/ailumiyana/p/10016965.html</p> <p>https://www.cnblogs.com/yangang92/p/5485868.html</p> |
| 10. OpenCV 画图适应 odd | |
| 11. 后期优化 | <p>TODO LIST:</p> <ol style="list-style-type: none"> 1. 还是有大量无用的值，下次优化的时候，计算点云的均值和方差，然后根据这个特点来进一步消除冗余 candidates。 2. 熟悉一下 shared_ptr 保证没有内存泄漏； 3. 所有的函数只要是传大量的值返回必须是传引用或者指针 <p>-----前三项是最急切的要求-----</p> <ol style="list-style-type: none"> 4. 找到大耗时的 function 优化函数实现 现在已经知道的有 CastRays 和 BBM 5. 后期优化 Cartographer 的位姿融合改成 ESKF imu 和 odometry 的紧耦合 6. 发挥 CSM 的方差让前端变成一个 imu 和 odometry 和 CSM 的 EKF 耦合估计 |

| | |
|---|---|
| | 7. 为了避免多次传值的 voxel_filter_scan 使用 new 和 delete 来管理点云内存，防止在 AddKeyScan 的时候传值 8. 第 7 的情况在很多函数中，传值的情况还是存在，需要后期全部换成，传引用或者传指针 9. 在有安全的内存保存时，class 内成员采用 const & 的形式来保存变量 value 10. 把所有类成员函数的 const 加上 11. M2MPGM 小头探出下边界的问题!!! 12. 回环的检测可以改成一个小 submap 和 submap 的 match |
| 12. Vector resize 和 reserve 的排查 | PMG CSM 没有添加 BBM 的版本已经 TEST PASS |
| 13. MapLimits 的 Contain 函数的 scaleXY 部分要检查一下 | +1 -1 的问题 也是 0 和 1 谁是最小的问题 直接包含在 8 项内一起检查过并且 TEST PASS |
| 14. LaserFan 的数据是全局坐标还是局部坐标要查一下 | 内部的 pose 是全局坐标，laserdata 是 pose 的局部坐标系内 |

Odd 地图逻辑
数值优化与位姿图
位姿图约束添加逻辑
Cartographer 融合逻辑 PoseExtraPolator
效果分析
V02(无回环版)激光里程计

| | | | | | |
|--|--|--|--|---|--|
| 结果对比 | | | | | |
| Submap motionFilter 0.08 0.02 | | Submap motionfilter 0.12 0.05 | | 单独 CSM 版 | |
|  | |  | |  | |
| Submap 0.15 0.05 | | Submap 0.22 0.1 | | V01 | |
|  | |  | |  | |
| 分析 | | | | | |

| |
|--|
| 1. 从结果来看精匹配的确是加大了计算量，V01 的 SLAM 在设计上更像是 LOAM 的 lidarMapping，这种从扫描点周围提取点的手法给 odometry 增加了很多的精度。 |
| 2. Submap 的设计在前端匹配上是局限性的(有很大的局部性) 因为他看见的之后一段时间内的几帧，很容易出现这几帧都很像的问题，在这一点上 V01 的地图生成手段就会好很多。这点的分析在 V01 改成 submap 结构的时候就有体现。 |
| 3. 有长廊问题还需要注意。 |
| 改进思想(目标是改善 submap 地图的局部性) |
| 1. 方法一 加大 motionfilter 的容许范围 结果发现在修改了 CSM 得分机制之后 MotionFilter 对结果影响不大 |
| 2. 在计算 submap 的时候计算上 submap 的质心和方差，在进行 CSM 粗匹配的时候把通过判别扫描帧的质心和方差选取附近 submap 进行 submap 拼接，然后在一个相对的大地图上进行匹配。 |
| 3. 加入回环 这个一定会需要加 2d 信息太少了 |

→后期优化

OpenCV 编程入门 C++

| | |
|-----------------------------|---|
| 1. Opencv 安装细枝末节 Windows | https://blog.csdn.net/weixin_43503473/article/details/103644313 + OpenCV 编程入门 P27 注意电脑和编译器是 x86 还是 x64 |
| 2. OpenCV imread 各种类型的图像 | https://www.learnopencv.com/read-an-image-in-opencv-python-cpp/ |
| 3. OpenCV Mat 元素访问 | https://www.cnblogs.com/kuotian/p/6389260.html |
| 4. OpenCV Mat 成员变量解读 | https://www.cnblogs.com/wangguchangqing/p/4016179.html |
| 5. OpenCV ROI 感兴趣区域操作 | 1. OpenCV3 编程入门(无目录) P132-133 有目录 115-116 两种方法来读取一个矩阵块 CopyTo() 来实现 2. AddWeight() 线性混合的部分没有再看了 https://www.cnblogs.com/skyfsm/p/6892746.html Rect()有大坑 它的参数 排列不是 row cols 而是 xy 也就是 cols row 的排列 2020.3.7 补充：记得是和 mat 实例化的顺序不同所以 debug 了很久才发现 |
| 6. OpenCV distanceTransform | 0 代表物体 非零值代表背景 |
| 7. OpenCV clone、copyTo | https://blog.csdn.net/DreamLike_zzg/article/details/79169336 https://blog.csdn.net/u013270326/article/details/72730812 clone 和 copyTo 的区别 copyTo 第二个参数 mask 中的非 0 会保留 0 值会变透明忽略 copyTo 的第二个参数只能是 CV_8UC1 不然内存泄漏 要自己写一个函数来进行矩阵和运算 |
| 8. OpenCV 画图 | https://blog.csdn.net/chaipp0607/article/details/56277479 几个画图的函数 https://tool.oschina.net/commons?type=3 RGB 色号 |
| 9. OPencv Mat to vector 转换 | https://blog.csdn.net/qg_27278957/article/details/88652661 |