SLAM-工作汇报-2021

路线图

- 计划投入： 3 * 5 = 15 （人月）
- 实际投入： 3 * 4 = 12（人月）

项目启动
A，基于导航算法，构建出 ROSA 导航能力

基础搭建
A，具备基本的数学知识
B，通过 ROS 仿真构建起导航整体功能

A, 建图-路径规划软件包去 ROS 化

导航调试
A，参照算法评估体系，输出评估报告
~~B，选定合适的算法，集成到实体机器人~~

算法深度调研
A，深入算法框架源码，输出原理及流程文档
~~B，对地图，定位关键功能进行评估，优化~~

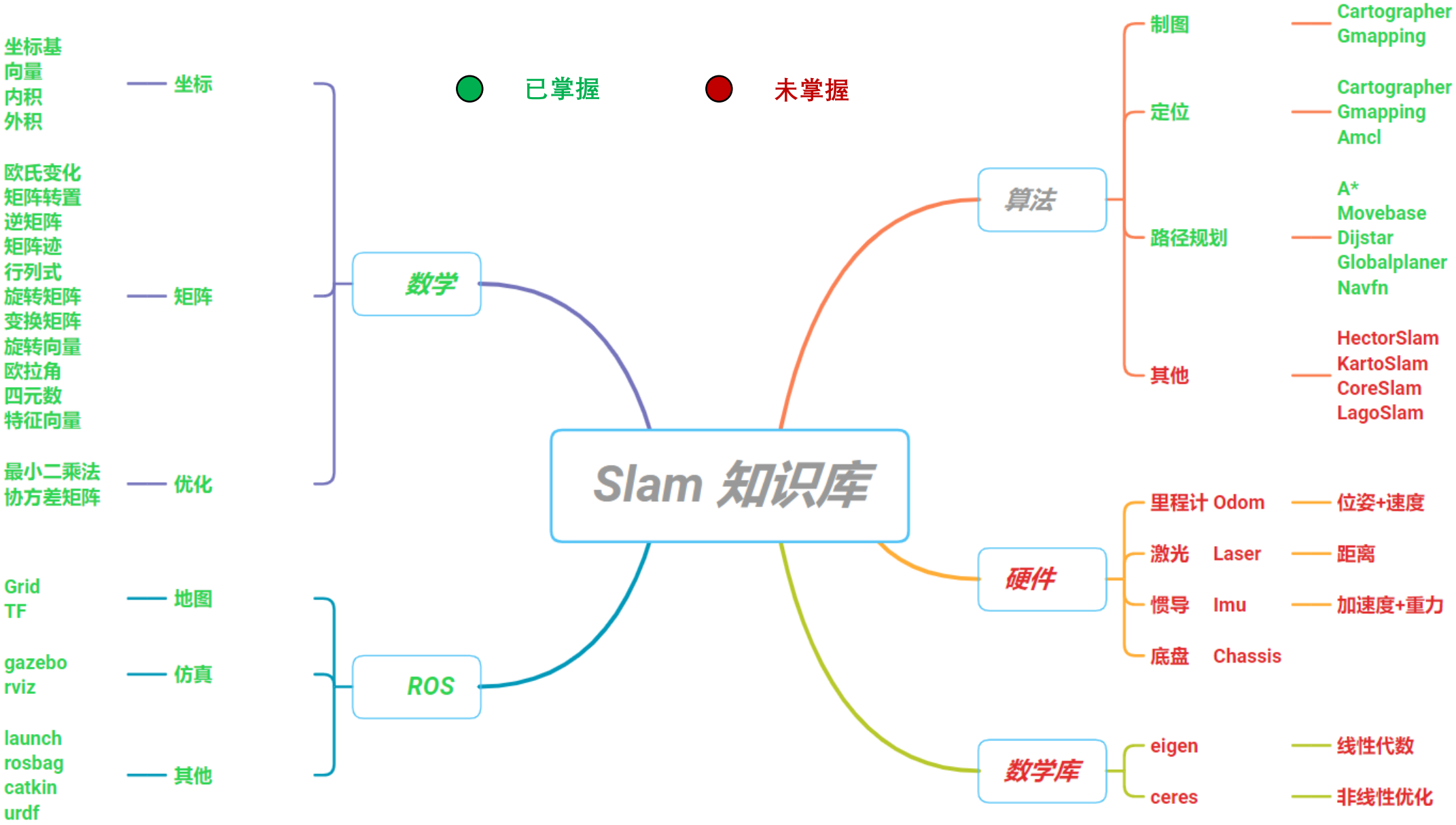
导航软件一期
A，总结

8月2日 9月2日 10月2日 11月2日 12月2日

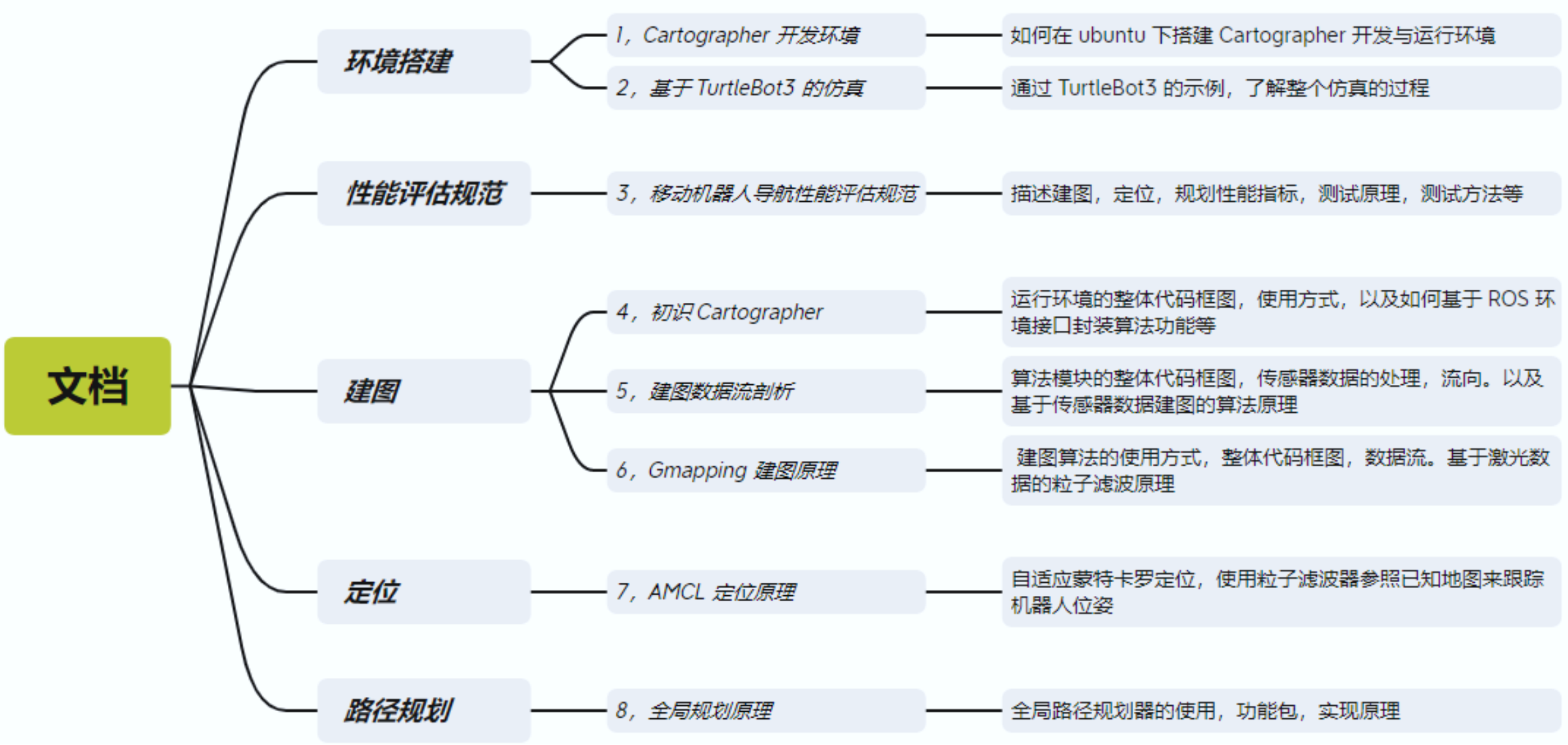
概述

1. 团队掌握了数学基础知识，熟练使用 ROS 开发环境，仿真工具等
2. 熟悉了主流的导航开源算法框架及使用方式，输出了使用及原理文档
3. 构建出开源导航算法评估体系，评估指标，测试方法等
4. ROS 环境使用 Gmapping/Amcl/Movebase 等软件包构建出导航能力
5. 对建图，路径规划软件包进行去 ROS 化，实现建图，全局规划功能

知识库

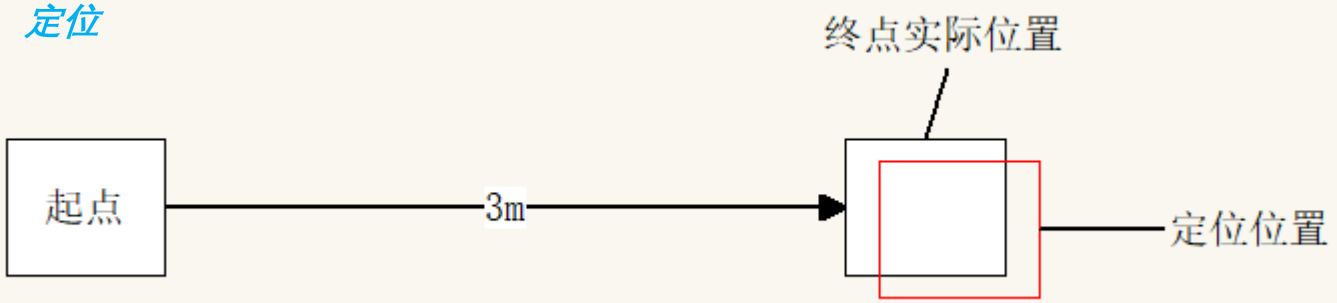

已掌握
未掌握
Slam 知识库
数学
坐标
坐标基
向量
内积
外积
矩阵
欧氏变化
矩阵转置
逆矩阵
矩阵迹
行列式
旋转矩阵
变换矩阵
旋转向量
欧拉角
四元数
特征向量
优化
最小二乘法
协方差矩阵
ROS
地图
Grid
TF
仿真
gazebo
rviz
其他
launch
rosbag
catkin
urdf
算法
制图
Cartographer
Gmapping
定位
Cartographer
Gmapping
Amcl
路径规划
A*
Movebase
Dijstar
Globalplaner
Navfn
其他
HectorSlam
KartoSlam
CoreSlam
LagoSlam
硬件
里程计 Odom
位姿+速度
激光 Laser
距离
惯导 Imu
加速度+重力
底盘 Chassis
数学库
eigen
线性代数
ceres
非线性优化

技术文档

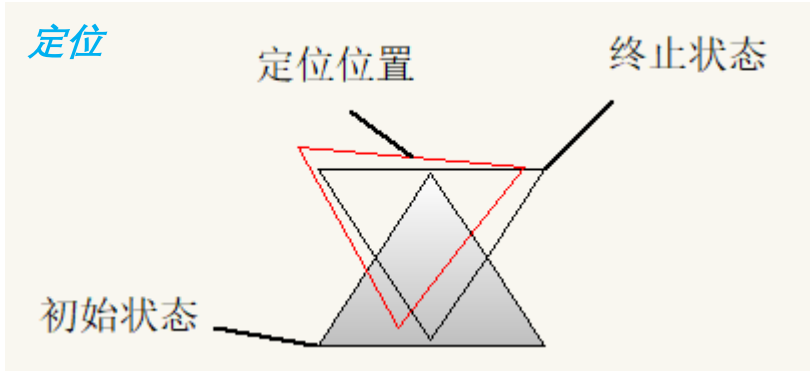

文档
环境搭建
1，Cartographer 开发环境
如何在 ubuntu 下搭建 Cartographer 开发与运行环境
2，基于 TurtleBot3 的仿真
通过 TurtleBot3 的示例，了解整个仿真的过程
性能评估规范
3，移动机器人导航性能评估规范
描述建图，定位，规划性能指标，测试原理，测试方法等
建图
4，初识 Cartographer
运行环境的整体代码框图，使用方式，以及如何基于 ROS 环境接口封装算法功能等
5，建图数据流剖析
算法模块的整体代码框图，传感器数据的处理，流向。以及基于传感器数据建图的算法原理
6，Gmapping 建图原理
建图算法的使用方式，整体代码框图，数据流。基于激光数据的粒子滤波原理
定位
7，AMCL 定位原理
自适应蒙特卡罗定位，使用粒子滤波器参照已知地图来跟踪机器人位姿
路径规划
8，全局规划原理
全局路径规划器的使用，功能包，实现原理

评估规范文档

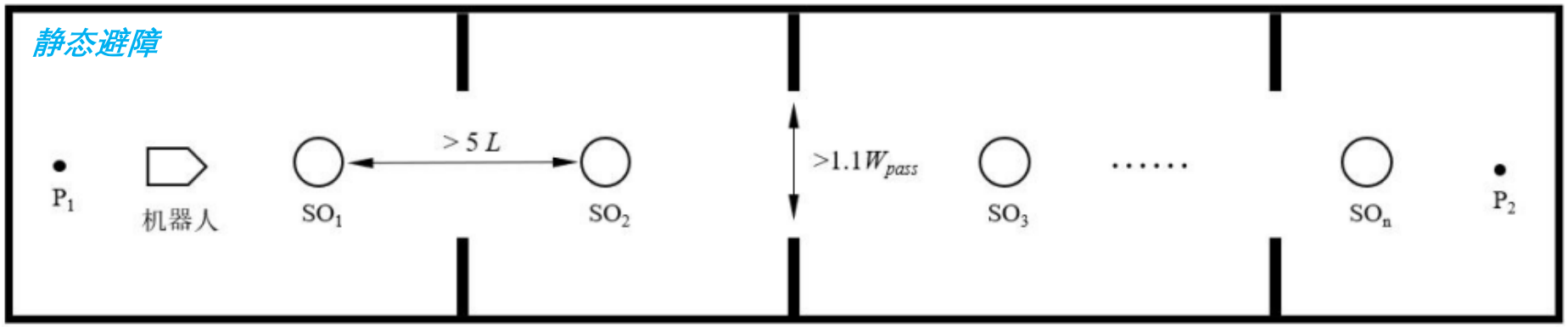
定位



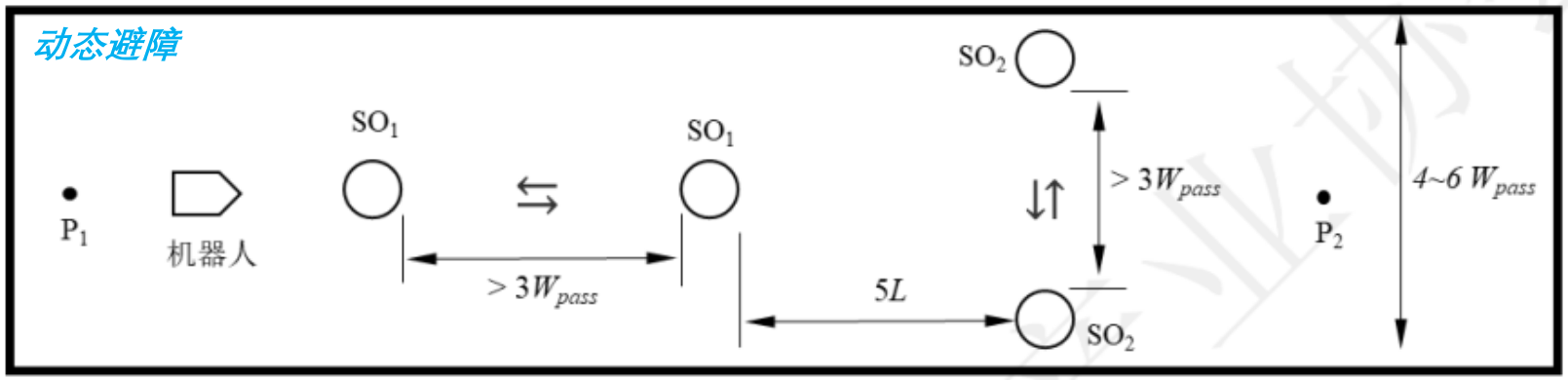
定位



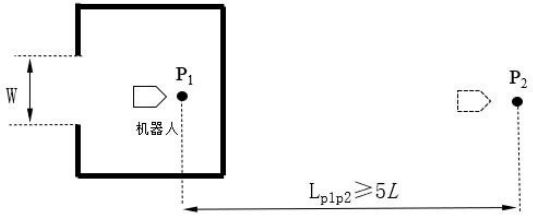
静态避障



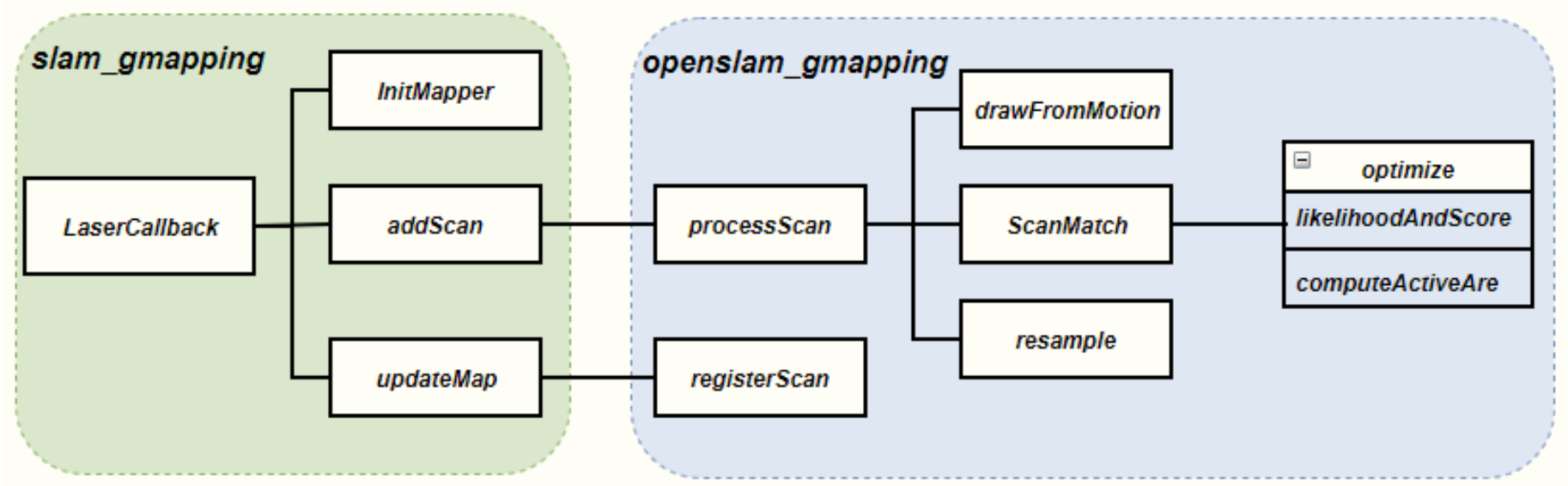
动态避障

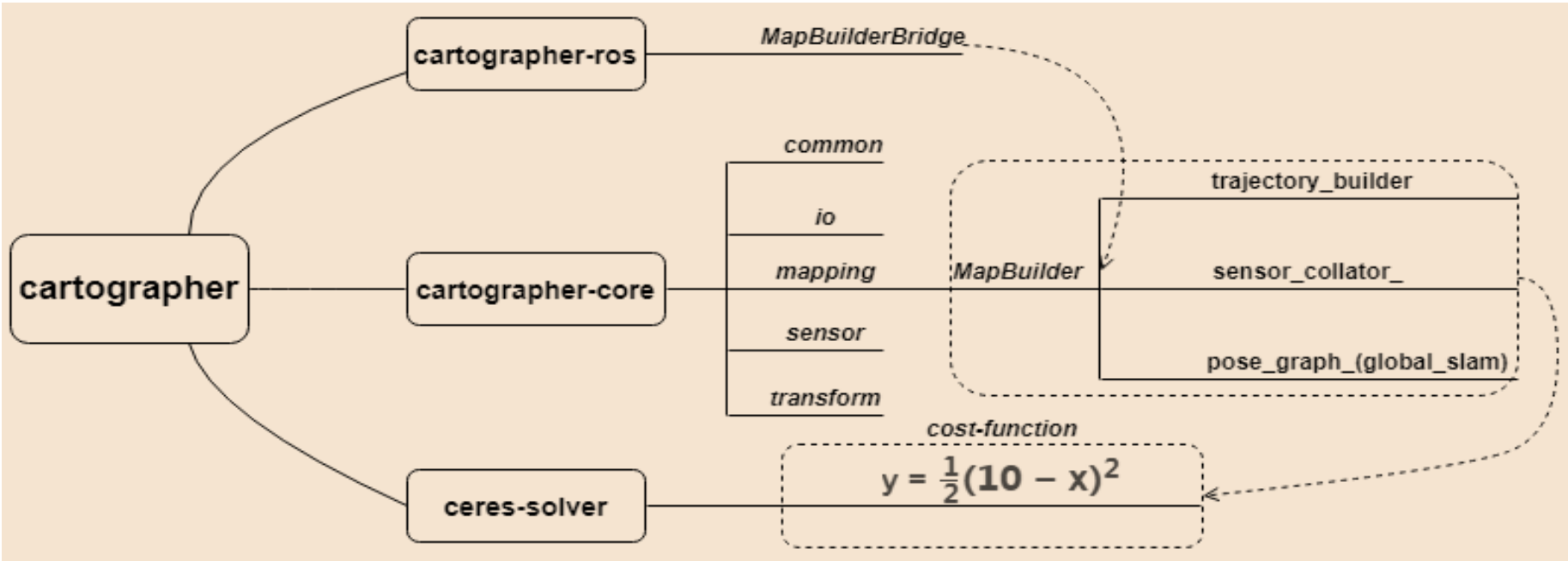


脱困

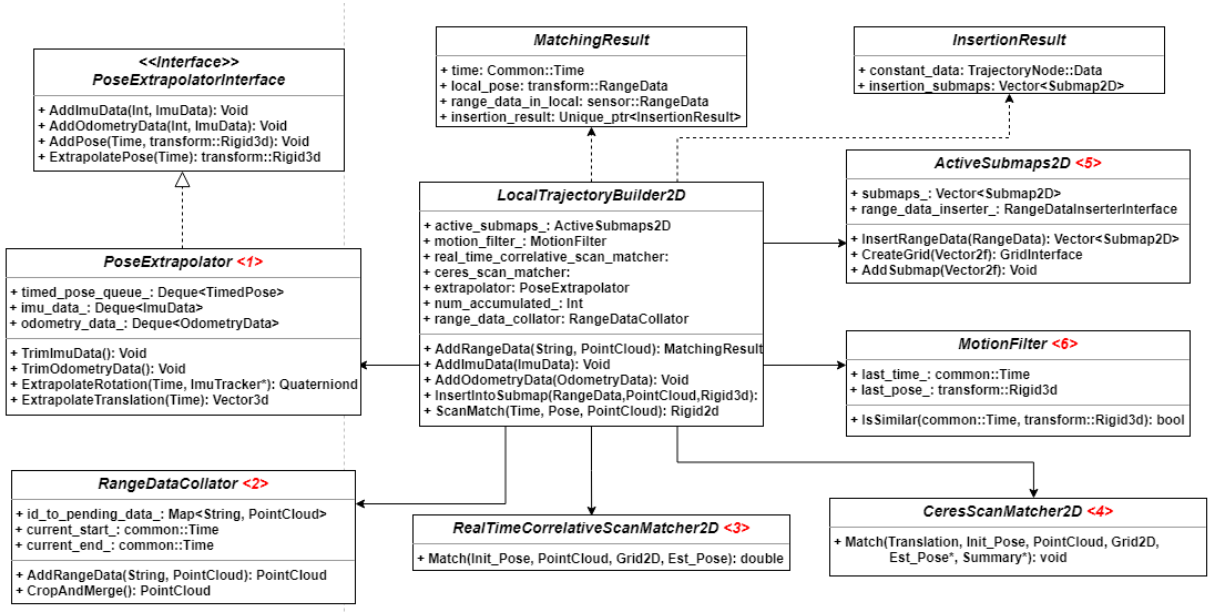


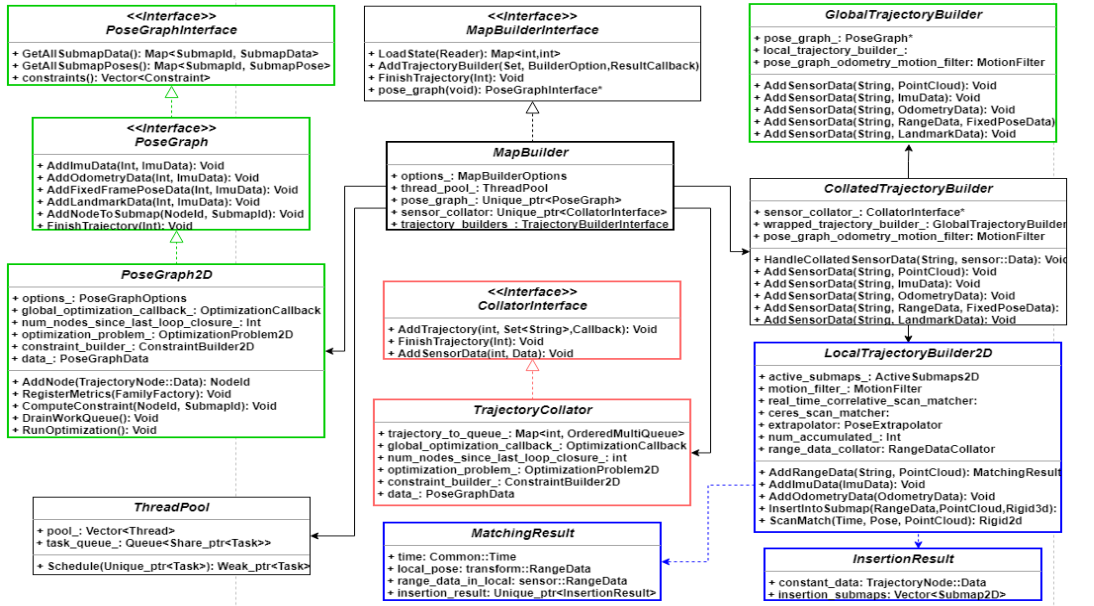
算法-架构

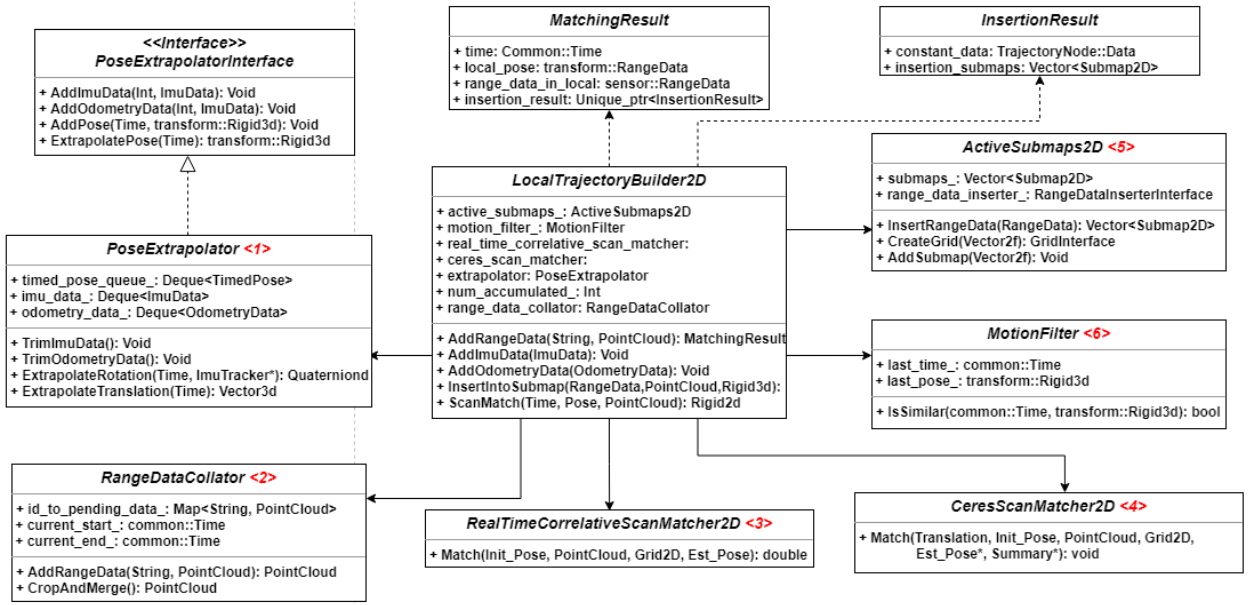

slam_gmapping
InitMapper
LaserCallback
addScan
updateMap
openslam_gmapping
processScan
registerScan
drawFromMotion
ScanMatch
resample
optimize
likelihoodAndScore
computeActiveAre

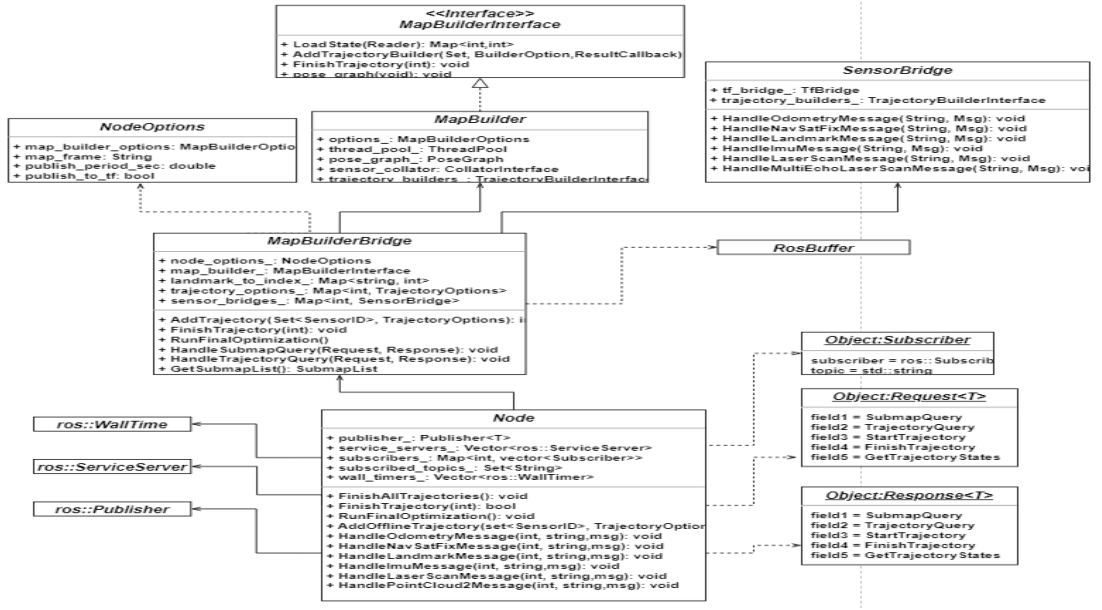

cartographer
cartographer-ros
MapBuilderBridge
cartographer-core
common
io
mapping
sensor
transform
MapBuilder
trajectory_builder
sensor_collator_
pose_graph_(global_slam)
ceres-solver
cost-function
$y = \frac{1}{2}(10 - x)^2$

算法-类图

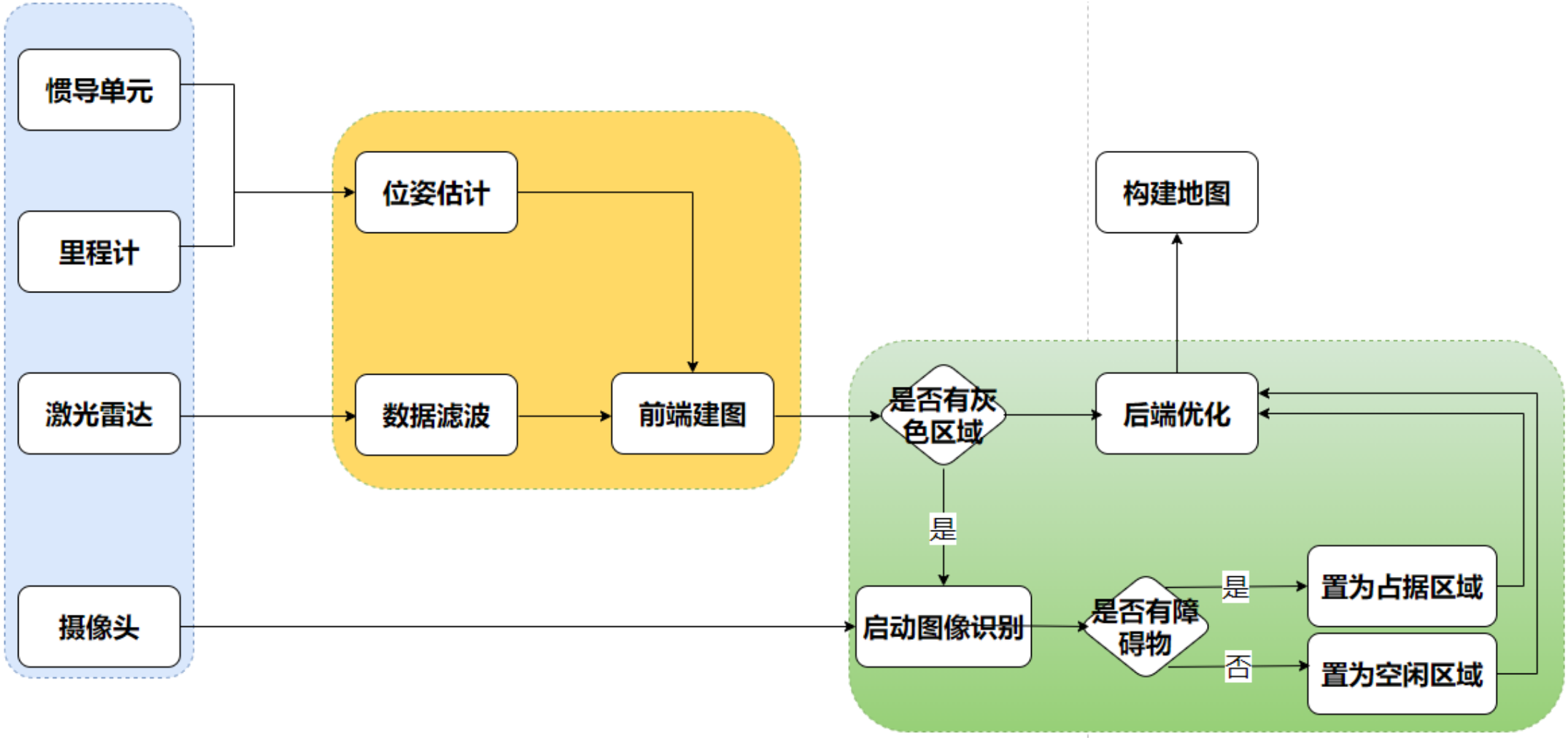

<<Interface>>
PoseExtrapolatorInterface
+ AddImuData(Int, ImuData): Void
+ AddOdometryData(Int, ImuData): Void
+ AddPose(Time, transform::Rigid3d): Void
+ ExtrapolatePose(Time): transform::Rigid3d
MatchingResult
+ time: Common::Time
+ local_pose: transform::RangeData
+ range_data_in_local: sensor::RangeData
+ insertion_result: Unique_ptr<InsertionResult>
InsertionResult
+ constant_data: TrajectoryNode::Data
+ insertion_submaps: Vector<Submap2D>
LocalTrajectoryBuilder2D
+ active_submaps_: ActiveSubmaps2D
+ motion_filter_: MotionFilter
+ real_time_correlative_scan_matcher:
+ ceres_scan_matcher:
+ extrapolator: PoseExtrapolator
+ num_accumulated_: Int
+ range_data_collator: RangeDataCollator
+ AddRangeData(String, PointCloud): MatchingResult
+ AddImuData(ImuData): Void
+ AddOdometryData(OdometryData): Void
+ InsertIntoSubmap(RangeData,PointCloud,Rigid3d):
+ ScanMatch(Time, Pose, PointCloud): Rigid2d
ActiveSubmaps2D <5>
+ submaps_: Vector<Submap2D>
+ range_data_inserter_: RangeDataInserterInterface
+ InsertRangeData(RangeData): Vector<Submap2D>
+ CreateGrid(Vector2f): GridInterface
+ AddSubmap(Vector2f): Void
PoseExtrapolator <1>
+ timed_pose_queue_: Deque<TimedPose>
+ imu_data_: Deque<ImuData>
+ odometry_data_: Deque<OdometryData>
+ TrimImuData(): Void
+ TrimOdometryData(): Void
+ ExtrapolateRotation(Time, ImuTracker*): Quaternion
+ ExtrapolateTranslation(Time): Vector3d
MotionFilter <6>
+ last_time_: common::Time
+ last_pose_: transform::Rigid3d
+ IsSimilar(common::Time, transform::Rigid3d): bool
RangeDataCollator <2>
+ id_to_pending_data_: Map<String, PointCloud>
+ current_start_: common::Time
+ current_end_: common::Time
+ AddRangeData(String, PointCloud): PointCloud
+ CropAndMerge(): PointCloud
RealTimeCorrelativeScanMatcher2D <3>
+ Match(Init_Pose, PointCloud, Grid2D, Est_Pose): double
CeresScanMatcher2D <4>
+ Match(Translation, Init_Pose, PointCloud, Grid2D,
Est_Pose*, Summary*): void


<<Interface>>
PoseGraphInterface
+ GetAllSubmapData(): Map<SubmapId, SubmapData>
+ GetAllSubmapPoses(): Map<SubmapId, SubmapPose>
+ constraints(): Vector<Constraint>
<<Interface>>
PoseGraph
+ AddImuData(Int, ImuData): Void
+ AddOdometryData(Int, ImuData): Void
+ AddFixedFramePoseData(Int, ImuData): Void
+ AddLandmarkData(Int, ImuData): Void
+ AddNodeToSubmap(NodeId, SubmapId): Void
+ FinishTrajectory(Int): Void
PoseGraph2D
+ options_: PoseGraphOptions
+ global_optimization_callback_: OptimizationCallback
+ num_nodes_since_last_loop_closure_: Int
+ optimization_problem_: OptimizationProblem2D
+ constraint_builder_: ConstraintBuilder2D
+ data_: PoseGraphData
+ AddNode(TrajectoryNode::Data): NodeId
+ RegisterMetrics(FamilyFactory): Void
+ ComputeConstraint(NodeId, SubmapId): Void
+ DrainWorkQueue(): Void
+ RunOptimization(): Void
ThreadPool
+ pool_: Vector<Thread>
+ task_queue_: Queue<Share_ptr<Task>>
+ Schedule(Unique_ptr<Task>): Weak_ptr<Task>
<<Interface>>
MapBuilderInterface
+ LoadState(Reader): Map<int,int>
+ AddTrajectoryBuilder(Set, BuilderOption,ResultCallback)
+ FinishTrajectory(Int): Void
+ pose_graph(void): PoseGraphInterface*
MapBuilder
+ options_: MapBuilderOptions
+ thread_pool_: ThreadPool
+ pose_graph_: Unique_ptr<PoseGraph>
+ sensor_collator: Unique_ptr<CollatorInterface>
+ trajectory_builders_: TrajectoryBuilderInterface
<<Interface>>
CollatorInterface
+ AddTrajectory(int, Set<String>,Callback): Void
+ FinishTrajectory(Int): Void
+ AddSensorData(int, Data): Void
TrajectoryCollator
+ trajectory_to_queue_: Map<int, OrderedMultiQueue>
+ global_optimization_callback_: OptimizationCallback
+ num_nodes_since_last_loop_closure_: int
+ optimization_problem_: OptimizationProblem2D
+ constraint_builder_: ConstraintBuilder2D
+ data_: PoseGraphData
MatchingResult
+ time: Common::Time
+ local_pose: transform::RangeData
+ range_data_in_local: sensor::RangeData
+ insertion_result: Unique_ptr<InsertionResult>
GlobalTrajectoryBuilder
+ pose_graph_: PoseGraph*
+ local_trajectory_builder_:
+ pose_graph_odometry_motion_filter: MotionFilter
+ AddSensorData(String, PointCloud): Void
+ AddSensorData(String, ImuData): Void
+ AddSensorData(String, OdometryData): Void
+ AddSensorData(String, RangeData, FixedPoseData)
+ AddSensorData(String, LandmarkData): Void
CollatedTrajectoryBuilder
+ sensor_collator_: CollatorInterface*
+ wrapped_trajectory_builder_: GlobalTrajectoryBuilder
+ pose_graph_odometry_motion_filter: MotionFilter
+ HandleCollatedSensorData(String, sensor::Data): Void
+ AddSensorData(String, PointCloud): Void
+ AddSensorData(String, ImuData): Void
+ AddSensorData(String, OdometryData): Void
+ AddSensorData(String, RangeData, FixedPoseData):
+ AddSensorData(String, LandmarkData): Void
LocalTrajectoryBuilder2D
+ active_submaps_: ActiveSubmaps2D
+ motion_filter_: MotionFilter
+ real_time_correlative_scan_matcher:
+ ceres_scan_matcher:
+ extrapolator: PoseExtrapolator
+ num_accumulated_: Int
+ range_data_collator: RangeDataCollator
+ AddRangeData(String, PointCloud): MatchingResult
+ AddImuData(ImuData): Void
+ AddOdometryData(OdometryData): Void
+ InsertIntoSubmap(RangeData,PointCloud,Rigid3d):
+ ScanMatch(Time, Pose, PointCloud): Rigid2d
InsertionResult
+ constant_data: TrajectoryNode::Data
+ insertion_submaps: Vector<Submap2D>


<<Interface>>
PoseExtrapolatorInterface
+ AddImuData(Int, ImuData): Void
+ AddOdometryData(Int, ImuData): Void
+ AddPose(Time, transform::Rigid3d): Void
+ ExtrapolatePose(Time): transform::Rigid3d
MatchingResult
+ time: Common::Time
+ local_pose: transform::RangeData
+ range_data_in_local: sensor::RangeData
+ insertion_result: Unique_ptr<InsertionResult>
InsertionResult
+ constant_data: TrajectoryNode::Data
+ insertion_submaps: Vector<Submap2D>
LocalTrajectoryBuilder2D
+ active_submaps_: ActiveSubmaps2D
+ motion_filter_: MotionFilter
+ real_time_correlative_scan_matcher:
+ ceres_scan_matcher:
+ extrapolator: PoseExtrapolator
+ num_accumulated_: Int
+ range_data_collator: RangeDataCollator
+ AddRangeData(String, PointCloud): MatchingResult
+ AddImuData(ImuData): Void
+ AddOdometryData(OdometryData): Void
+ InsertIntoSubmap(RangeData,PointCloud,Rigid3d):
+ ScanMatch(Time, Pose, PointCloud): Rigid2d
ActiveSubmaps2D <5>
+ submaps_: Vector<Submap2D>
+ range_data_inserter_: RangeDataInserterInterface
+ InsertRangeData(RangeData): Vector<Submap2D>
+ CreateGrid(Vector2f): GridInterface
+ AddSubmap(Vector2f): Void
PoseExtrapolator <1>
+ timed_pose_queue_: Deque<TimedPose>
+ imu_data_: Deque<ImuData>
+ odometry_data_: Deque<OdometryData>
+ TrimImuData(): Void
+ TrimOdometryData(): Void
+ ExtrapolateRotation(Time, ImuTracker*): Quaternion
+ ExtrapolateTranslation(Time): Vector3d
MotionFilter <6>
+ last_time_: common::Time
+ last_pose_: transform::Rigid3d
+ IsSimilar(common::Time, transform::Rigid3d): bool
RangeDataCollator <2>
+ id_to_pending_data_: Map<String, PointCloud>
+ current_start_: common::Time
+ current_end_: common::Time
+ AddRangeData(String, PointCloud): PointCloud
+ CropAndMerge(): PointCloud
RealTimeCorrelativeScanMatcher2D <3>
+ Match(Init_Pose, PointCloud, Grid2D, Est_Pose): double
CeresScanMatcher2D <4>
+ Match(Translation, Init_Pose, PointCloud, Grid2D,
Est_Pose*, Summary*): void

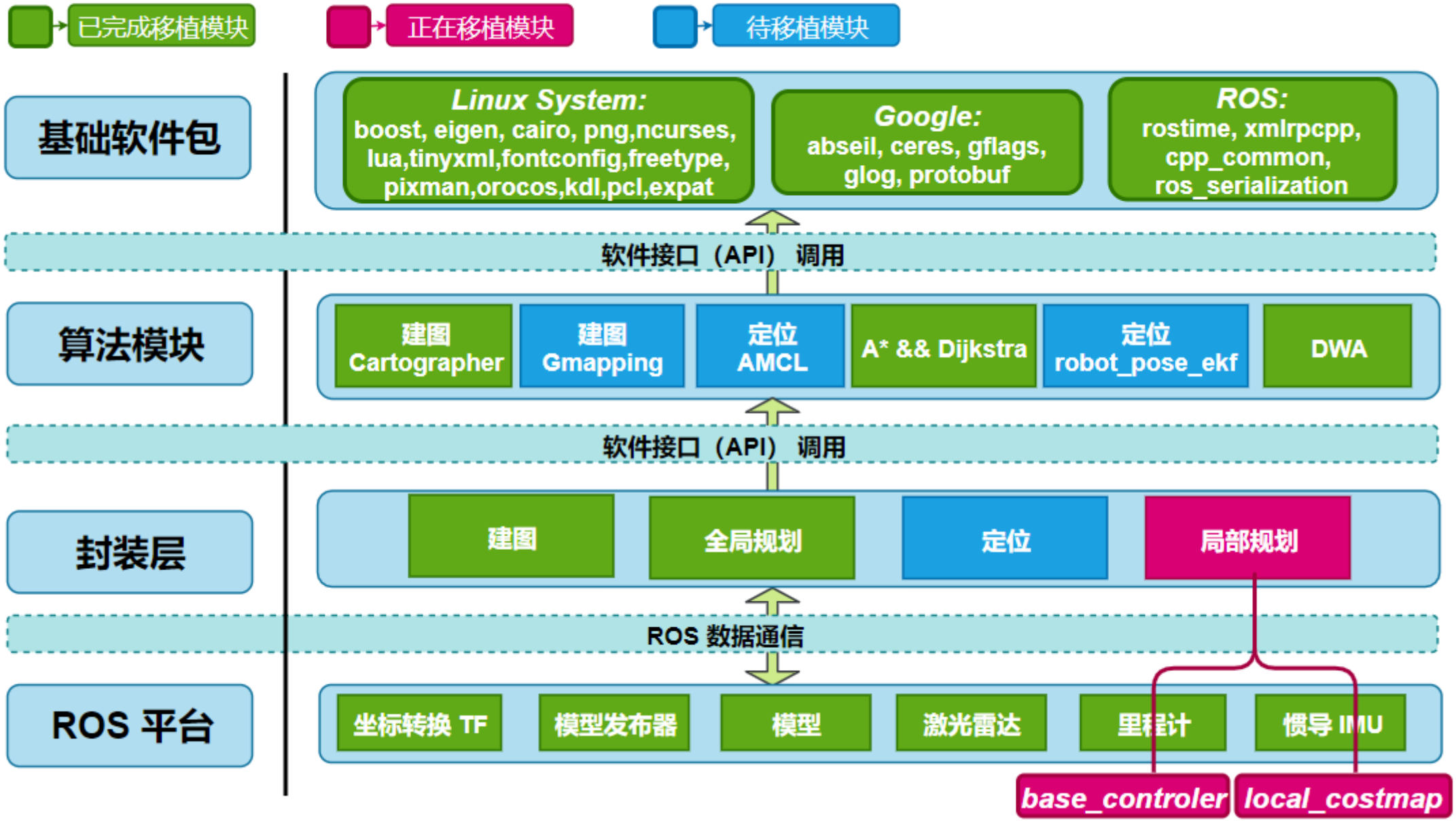

<<Interface>>
MapBuilderInterface
+ LoadState(Reader): Map<int,int>
+ AddTrajectoryBuilder(Set, BuilderOption,ResultCallback)
+ FinishTrajectory(int): void
+ pose_graph(void): void
SensorBridge
+ tf_bridge_: TfBridge
+ trajectory_builders_: TrajectoryBuilderInterface
+ HandleOdometryMessage(String, Msg): void
+ HandleNavSatFixMessage(String, Msg): void
+ HandleLandmarkMessage(String, Msg): void
+ HandleImuMessage(String, Msg): void
+ HandleLaserScanMessage(String, Msg): void
+ HandleMultiEchoLaserScanMessage(String, Msg): voi
NodeOptions
+ map_builder_options: MapBuilderOptio
+ map_frame: String
+ publish_period_sec: double
+ publish_to_tf: bool
MapBuilder
+ options_: MapBuilderOptions
+ thread_pool_: ThreadPool
+ pose_graph_: PoseGraph
+ sensor_collator: CollatorInterface
+ trajectory_builders_: TrajectoryBuilderInterfac
MapBuilderBridge
+ node_options_: NodeOptions
+ map_builder_: MapBuilderInterface
+ landmark_to_index_: Map<string, int>
+ trajectory_options_: Map<int, TrajectoryOptions>
+ sensor_bridges_: Map<int, SensorBridge>
+ AddTrajectory(Set<SensorId>, TrajectoryOptions): i
+ FinishTrajectory(int): void
+ RunFinalOptimization()
+ HandleSubmapQuery(Request, Response): void
+ HandleTrajectoryQuery(Request, Response): void
+ GetSubmapList(): SubmapList
RosBuffer
Object:Subscriber
subscriber = ros::Subscrib
topic = std::string
Object:Request<T>
field1 = SubmapQuery
field2 = TrajectoryQuery
field3 = StartTrajectory
field4 = FinishTrajectory
field5 = GetTrajectoryStates
Object:Response<T>
field1 = SubmapQuery
field2 = TrajectoryQuery
field3 = StartTrajectory
field4 = FinishTrajectory
field5 = GetTrajectoryStates
ros::WallTime
ros::ServiceServer
ros::Publisher
Node
+ publisher_: Publisher<T>
+ service_servers_: Vector<ros::ServiceServer>
+ subscribers_: Map<int, vector<Subscriber>>
+ subscribed_topics_: Set<String>
+ wall_timers_: Vector<ros::WallTimer>
+ FinishAllTrajectories(): void
+ FinishTrajectory(int): bool
+ RunFinalOptimization(): void
+ AddOfflineTrajectory(set<SensorId>, TrajectoryOptio
+ HandleOdometryMessage(int, string,msg): void
+ HandleNavSatFixMessage(int, string,msg): void
+ HandleLandmarkMessage(int, string,msg): void
+ HandleImuMessage(int, string,msg): void
+ HandleLaserScanMessage(int, string,msg): void
+ HandlePointCloud2Message(int, string,msg): void

- 一种提升扫地机器人地图清晰度的方法

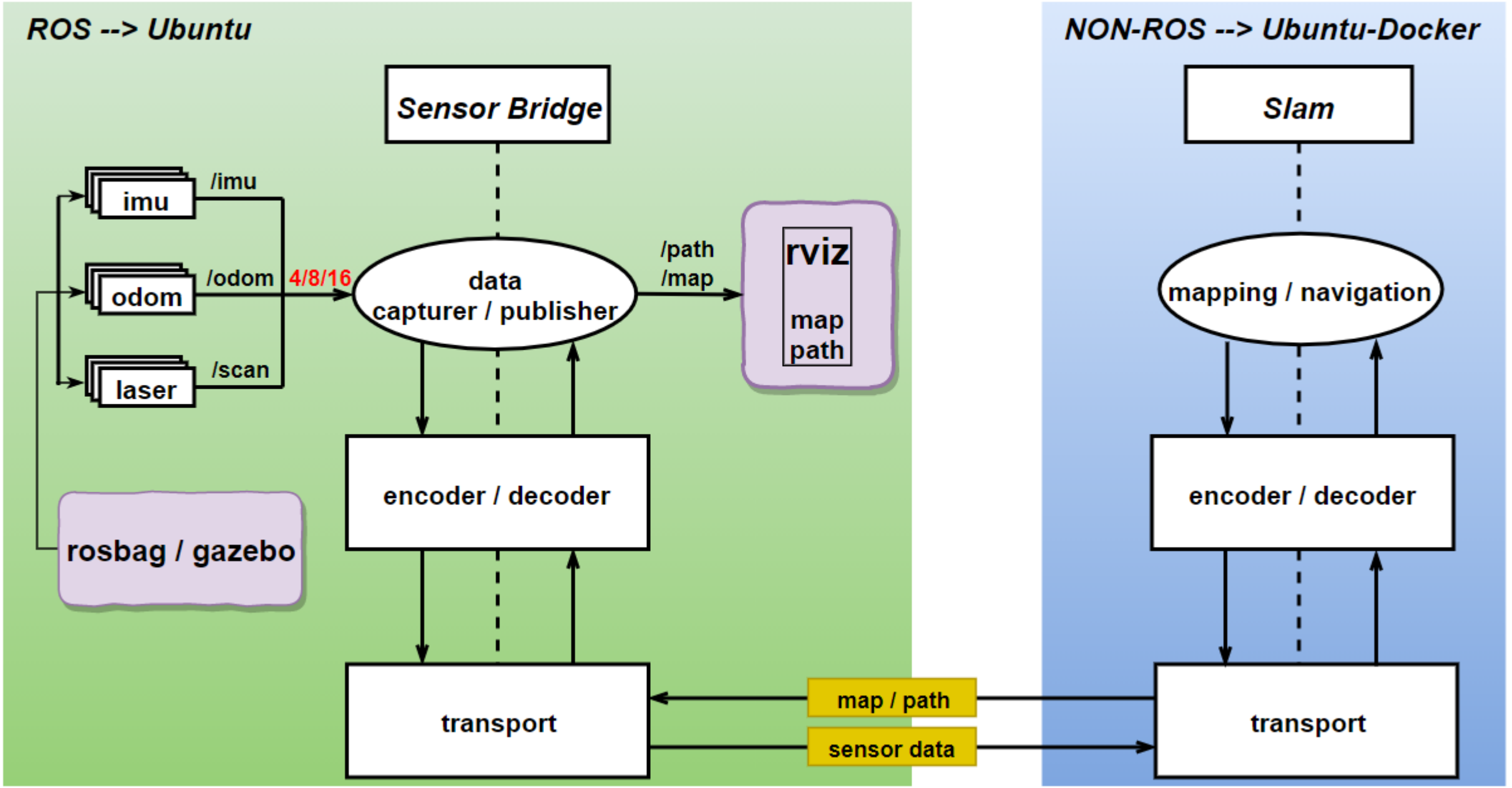


地图优化

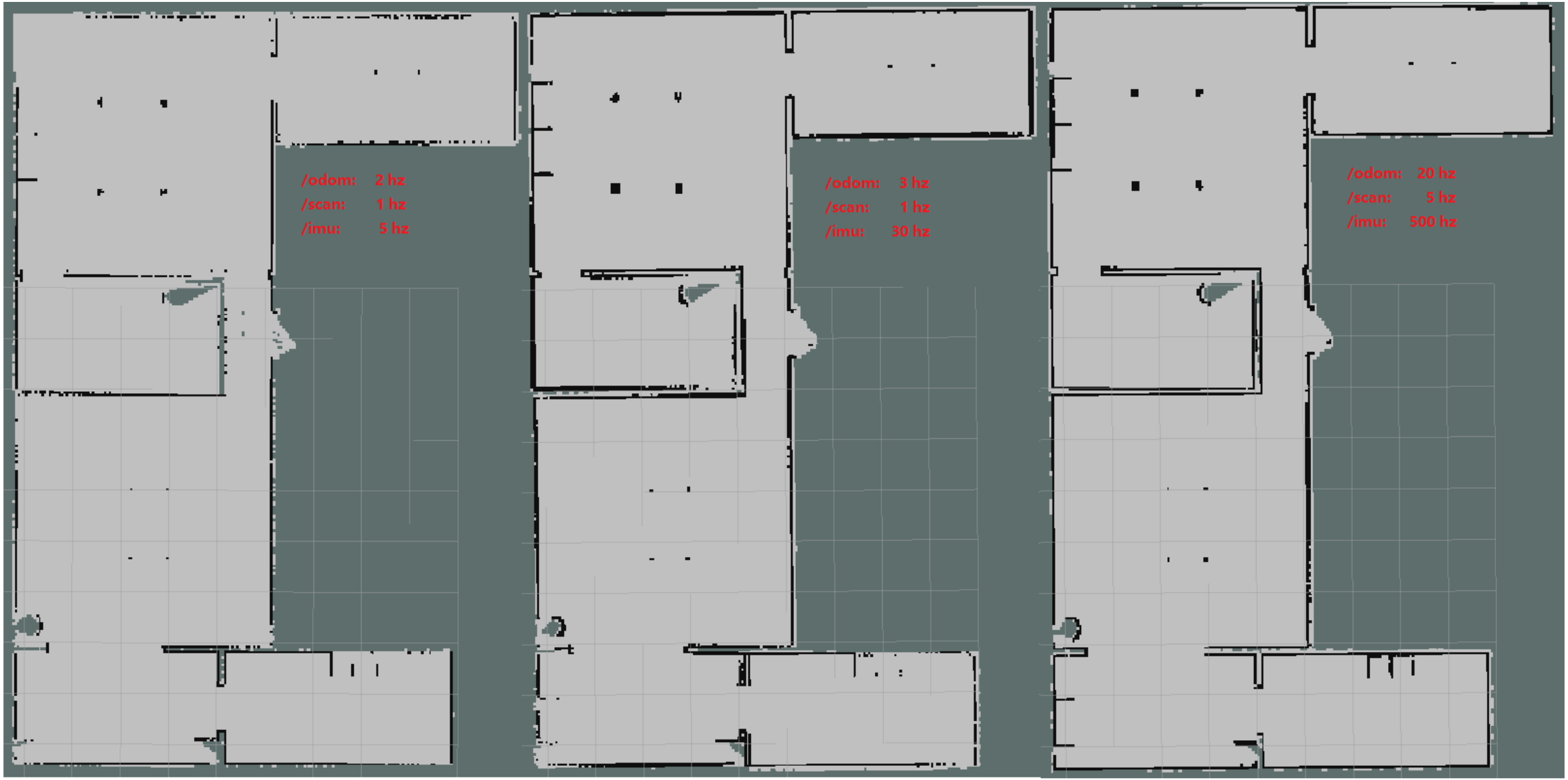
去 ROS 化


已完成移植模块
正在移植模块
待移植模块
基础软件包
Linux System:
boost, eigen, cairo, png,ncurses,
lua,tinyxml,fontconfig,freetype,
pixman,orocos,kdl,pcl,expat
Google:
abseil, ceres, gflags,
glog, protobuf
ROS:
rostime, xmlrpcpp,
cpp_common,
ros_serialization
软件接口（API） 调用
算法模块
建图
Cartographer
建图
Gmapping
定位
AMCL
A* && Dijkstra
定位
robot_pose_ekf
DWA
软件接口（API） 调用
封装层
建图
全局规划
定位
局部规划
ROS 数据通信
ROS 平台
坐标转换 TF
模型发布器
模型
激光雷达
里程计
惯导 IMU
base_controler
local_costmap

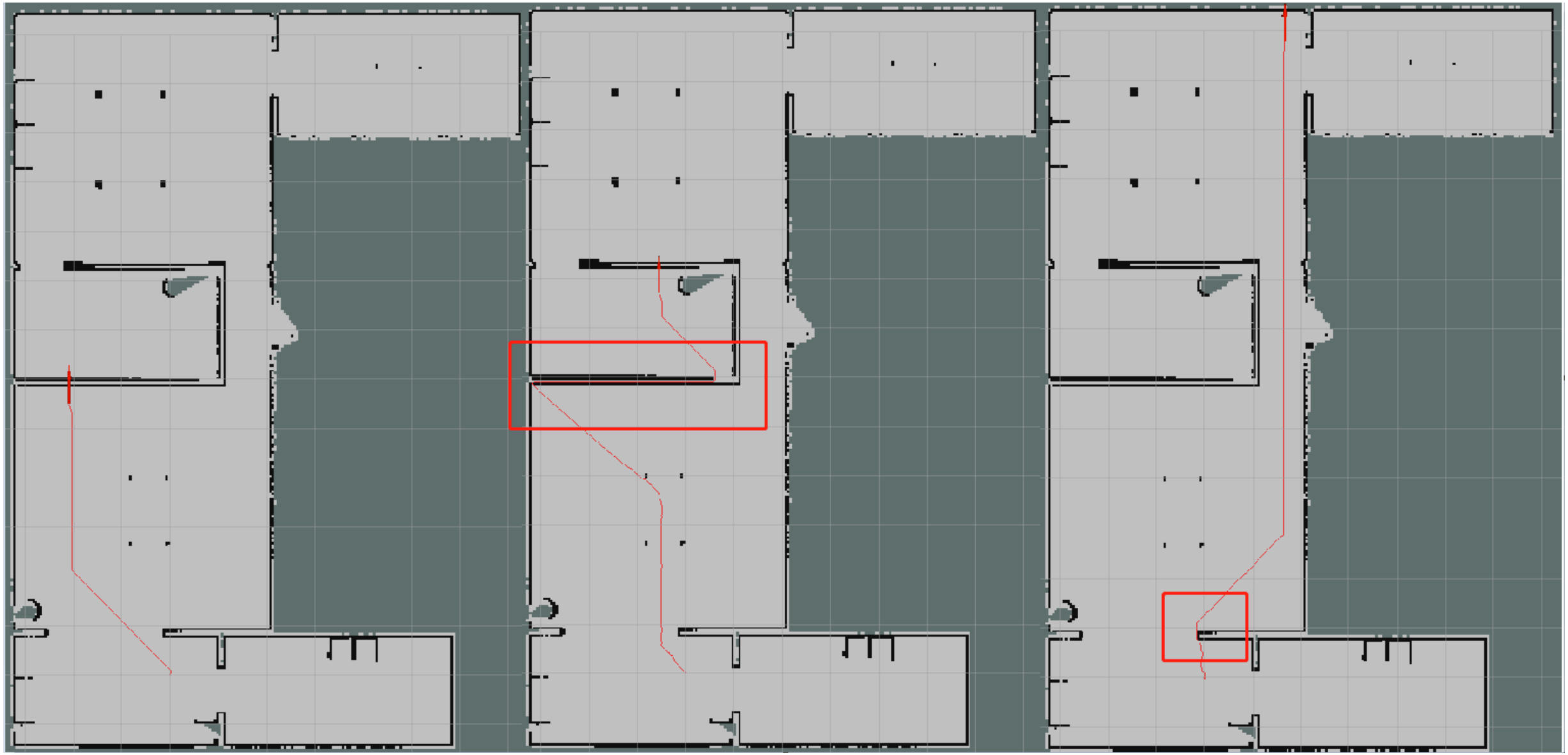
Sensor Bridge



传感器帧率@建图


/odom: 2 hz
/scan: 1 hz
/imu: 5 hz
/odom: 3 hz
/scan: 1 hz
/imu: 30 hz
/odom: 20 hz
/scan: 5 hz
/imu: 500 hz

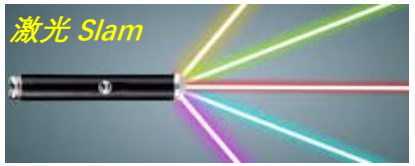
全局路径规划



未完成工作

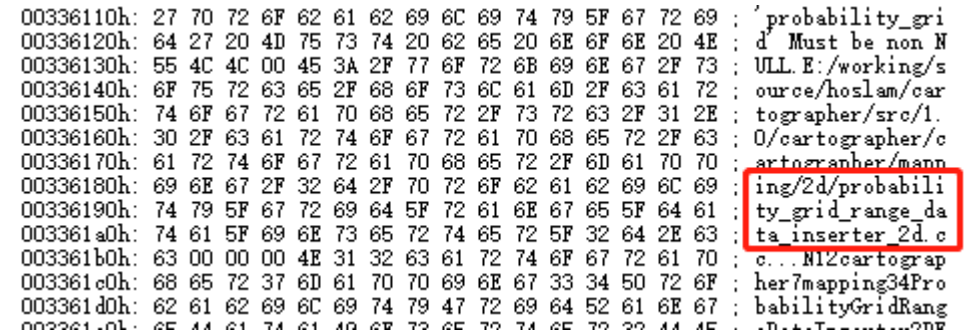
工作项	总体状态	说明
集成研究院 USLAM 到实体机器人	工作未开展	必要性不大
选定合适的算法，集成到实体机器人	工作未开展	没有找到合适的机器人硬件
对导航功能进行两项效果明显的优化	完成一项优化	有一项地图显示效果优化，暂未找到更多优化项
文档，技术分享	已完成 8 篇文档	文档可以持续补充完善

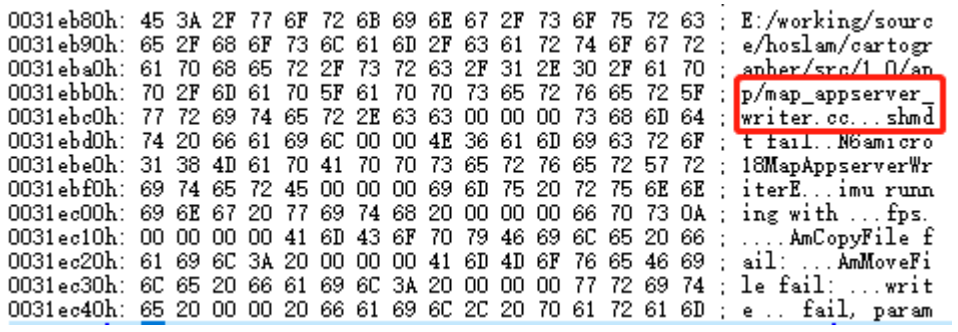
下年主要工作

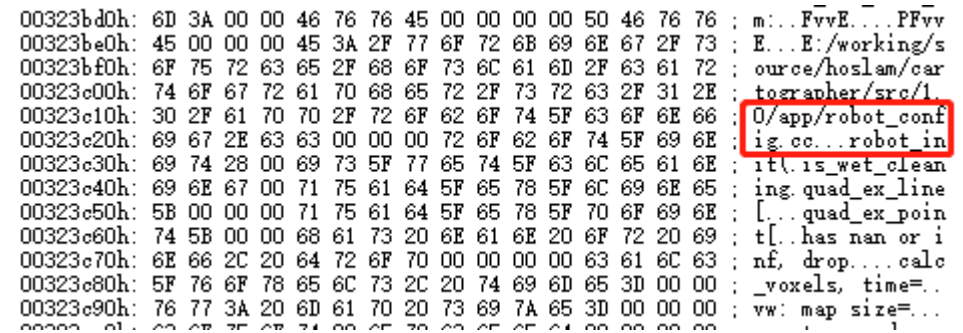


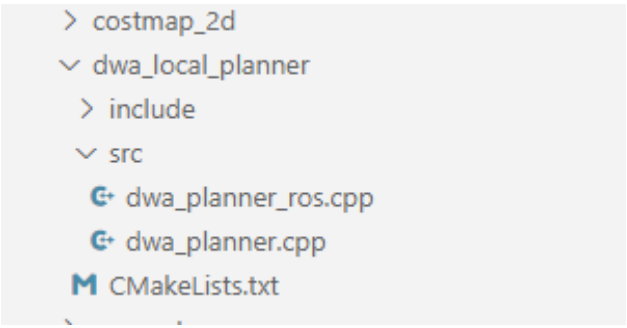
1. 局部路径规划软件包去 ROS 化，并通过传感器数据测试验证
2. 更多软件包（gmapping，amcl, rrt）去 ROS 化移植
3. 设计规范的 ROSA-SLAM 工程，参考其他导航解决方案设计对外接口
4. 对去 ROS 化后的各个零散软件包整合成一个完整的 SLAM 功能包
5. 对 ROSA-SLAM 功能包测试，验证，优化
6. 移植 ROSA-SLAM 功能包到扫地机，与 AM 导航功能对比测试
7. 路径探索，机器人移动控制等实现

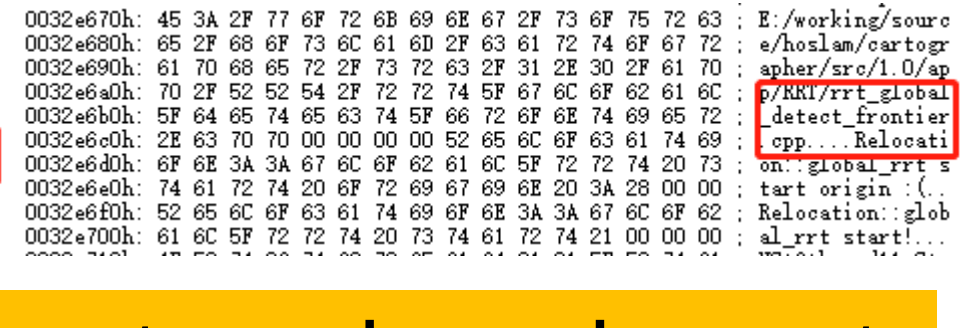
AMICRO





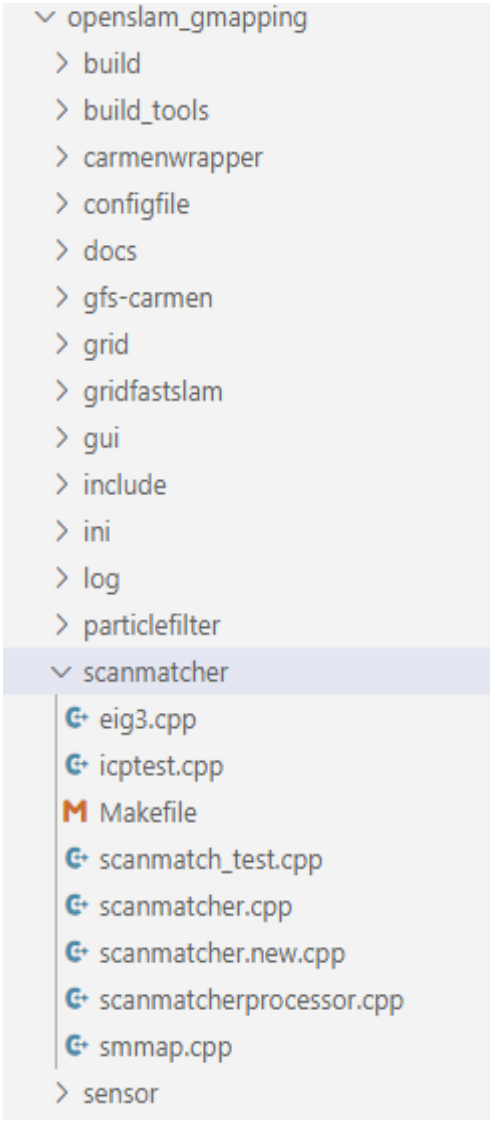






Hoslam: cartographer + dwa + rrt

USLAM



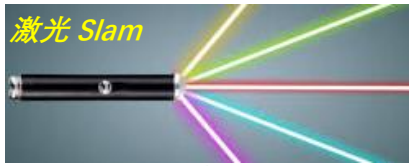

```
untu:~/project/temp/install/lib/uslam$ ldd uslam
 linux-vdso.so.1 (0x00007ffee9521000)
 libSDL-1.2.so.0 => /usr/lib/x86_64-linux-gnu/libSDL-1.2.so.0 (0x0
 libSDL_image-1.2.so.0 => /usr/lib/x86_64-linux-gnu/libSDL_image-1
 libyaml-cpp.so.0.5 => /usr/lib/x86_64-linux-gnu/libyaml-cpp.so.0.
 libboost_system.so.1.65.1 => /usr/lib/x86_64-linux-gnu/libboost_sy
 libboost_thread.so.1.65.1 => /usr/lib/x86_64-linux-gnu/libboost_th
 libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007
 libgridfastslam.so => /usr/local/lib/libgridfastslam.so (0x00007fb
 libscanmatcher.so => /usr/local/lib/libscanmatcher.so (0x00007fb2e
 libsensor_range.so => /usr/local/lib/libsensor_range.so (0x00007fb
 libsensor_odometry.so => /usr/local/lib/libsensor_odometry.so (0x0
 libutils.so => /usr/local/lib/libutils.so (0x00007fb2ea4e6000)
```

```
6E 74 41 63 63 75 6D 75 6C 61 74 6F 72 45 : ointAccumulatorE
30 45 45 44 31 45 76 00 5F 5A 4E 38 47 4D : L0EED1Ev._ZN8GM
70 69 6E 67 31 37 47 72 69 64 53 6C 61 6D : apping17GridSlam
6F 63 65 73 73 6F 72 32 34 73 65 74 4D 6F : Processor24setMo
6F 6E 4D 6F 64 65 6C 50 61 72 61 6D 65 74 : tionModelParamet
73 45 64 64 64 64 00 5F 5A 4E 38 47 4D 61 : ersEdddd._ZN8GMa
69 6E 67 31 37 47 72 69 64 53 6C 61 6D 50 : pping17GridSlamP
63 65 73 73 6F 72 34 69 6E 69 74 45 6A 64 : rocessor4initEjd
64 64 52 53 74 36 76 65 63 74 6F 72 49 61 : ddddRSt6vectorIa
49 61 45 45 4E 53 5F 31 33 6F 72 69 65 6E : SaIaEENS_13orien
64 70 6F 69 6E 74 49 64 64 45 45 00 5F 5A : tedpointIddEE._Z
63 76 66 6C 61 6E 6E 37 61 6E 79 69 6D 70 : N7cvflann7anyimp
36 73 6D 61 6C 6C 5F 61 6E 79 5F 70 6F 6C : l16small_any_pol
79 49 6A 45 39 67 65 74 5F 76 61 6C 75 65 : icyIjE9get_value
```

```
mike@ubuntu:~/project/temp/install/lib$ ls *.so
libamcl_map.so                  libFaceTracker.so          liblayers.so                        librealsense2_camera.so
libamcl_pf.so                   libfollow_unit.so          liblocal_mapper.so                  libreset_costmap_recovery.so
libamcl_sensors.so              libglobal_planner.so       libmap_server_image_loader.so       librosa.so
libbase_local_planner.so        libgridfastslam.so         libmove_base.so                     librotate_and_forward_recovery
libcarrot_planner.so            libicpPointToPlane.so      libmove_slow_and_clear.so           librotate_and_move_recovery.so
libcatmullrom_spline.so         libicpPointToPoint.so      libnarrow_corridor_behavior.so      librotate_recovery.so
libclear_costmap_recovery.so    libiMemory.so              libnavfn.so                         libscanmatcher.so
libcostmap_2d.so                libjsoncpp180.so           libnavi_unit.so                     libsensor_base.so
libcostmap_converter.so         libjsoncpp.so              libnew_rotate_recovery.so           libsensor_odometry.so
libdwa_local_planner.so         libjson_parse_lib.so       libpath_planner.so                  libsensor_range.so
libFaceRecWrapper.so            liblaser_scan_filters.so   librange_sensor_layer.so            libspatio_temporal_voxel_layer
```

Uslam: amcl + dwa + global_planner + gmapping

目标



Now

Long Term