



中国科学院大学

University of Chinese Academy of Sciences

# 博士学位论文

针对室内地图的拓扑表示与基于拓扑图的路径规划和地图匹配

作者姓名: 侯佳维

指导教师: Sören Schwertfeger 副教授

上海科技大学

学位类别: 工学博士

学科专业: 通信与信息系统

培养单位: 中国科学院上海微系统与信息技术研究所

2022 年 12 月



**Indoor Topological Representation and Topological  
Graph-Based Path Planning and Map Matching**

**A dissertation submitted to  
University of Chinese Academy of Sciences  
in partial fulfillment of the requirement  
for the degree of  
Doctor of Philosophy  
in Communication and Information Systems**

**By**

**Hou Jiawei**

**Supervisor: Sören Schwertfeger**

**Shanghai Institute of Microsystems and Information Technology**

**December, 2022**



**中国科学院大学**  
**学位论文原创性声明**

本人郑重声明：所提交的学位论文是本人在导师的指导下独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明或致谢。

作者签名：

日 期：

**中国科学院大学**  
**学位论文授权使用声明**

本人完全了解并同意遵守中国科学院有关保存和使用学位论文的规定，即中国科学院有权保留送交学位论文的副本，允许该论文被查阅，可以按照学术研究公开原则和保护知识产权的原则公布该论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存、汇编本学位论文。

涉密及延迟公开的学位论文在解密或延迟期后适用本声明。

作者签名：

日 期：

导师签名：

日 期：



## 摘要

地图在机器人领域中有着广泛应用，目前最常用的地图形式有占用栅格地图和点云，这两类地图形式都是以表示环境状态的非常小的单元（单元格、点）作为图的节点，地图分辨率越高，其对环境的表示就越精确，但是在应用中进行索引和计算时就会更耗时。一个好的解决办法是从地图中提取拓扑表示，这可提高语义地图构建、大型环境下的全局路径规划和地图匹配等许多应用的计算速度。因为拓扑地图中的每个节点不再是环境中的一个具体点，而可以是环境中的一片区域、一块空间、或一个地点。在人类眼中，区域比点能更好地描述一个地点，例如，当提及“时代广场”，人们常常想到一个广场的区域范围而非广场中的某一个点。对地图进行合理的分割是构建基于区域的拓扑地图的基础。

本篇论文首先提出了一种创新的拓扑地图表示方式——区域图 (Area Graph)——及从二维的室内栅格地图中提取这种拓扑地图的方法。区域图是以区域为节点的拓扑地图，为了将地图划分为有意义的区域，作者提出了基于维诺图 (Voronoi Diagram) 和房间检测的地图分割算法。通过将本文的区域构建方法与两种地图分割方法进行比较，实验显示本文方法的分割结果更接近人工给定的参考结果。并且，对迷宫地图的分割实验显示，本拓扑表示更适用于迷宫应用。

随后，本文作者提出了基于区域图的全局路径规划方法。此方法从区域图中构建出通道图 (Passage Graph)，然后以通道图的节点为搜索空间进行寻路。通过对比使用 A\* 搜索算法分别在二维栅格地图和由此栅格地图提取的通道图上搜索路径的时间，后者的路径规划计算时间比前者减少了两个数量级。路径规划是机器人领域的重要应用，作者通过此应用来展示拓扑地图相比于栅格地图在减少算法计算量上的巨大优势。

本文的另一重要贡献是提出了基于拓扑地图表示的两种地图匹配方法，分别是启发式聚类法 (HypoHypothesis Clustering) 和生长邻居法 (Neighbor Growing)，分别应用于地图间有、无全局一致变换两种情况下的地图匹配。启发式聚类法利用从区域中提取特征进行两幅地图间的区域匹配，在可能正确的区域匹配中通过聚类的方式投票选出正确的地图间变换。而生长邻居法的主要思路是不断通过节点的相邻关系把地图间成功匹配的区域加入公共子图，从而在地图间无一

致变换的情况下将地图的不同部分对应起来。针对两种方法，作者分别进行了对比实验：1) 对于有全局一致变换的地图间匹配，作者将启发式聚类法与常用的图像匹配方法、点云配准方法 ICP (Iterative Closest Point)、最近的基于区域匹配的地图匹配方法进行比较，实验结果表明启发式聚类法比其他方法有更高的地图匹配准确率，尤其是在不同模态的地图匹配上有显著优势。2) 对于损坏的地图，输入的地图间无一致的全局变换，作者将生长邻居法与最近的非刚体点配准方法作比较。实验显示生长邻居法可更好地匹配地图的不同部分。

除了二维地图的拓扑表示 Area Graph，本文还提出了以多楼层建筑的三维点云为输入构建的多层次三维拓扑表示。该拓扑表示主要的贡献在于解决了在三维地图进行区域分割并且抽象出拓扑表示的问题，其次，它是多层次的拓扑图结构：其高层级的拓扑节点是低层级的拓扑节点的父亲。该拓扑结构主要包含三个层级，分别代表的三维地图结构是：楼层-区域-空间快。在区域级拓扑图中，房间和走廊等区域为拓扑图的节点，门作为连接节点的边。该算法的核心是通过合并空闲空间块和检测门来获得区域。除此之外，本方法在这些区域的基础上使用区域图生成算法对这些区域进行进一步分割，使房间分割更具鲁棒性并且提高了机器人在走廊区域进行导航的实用性。实验显示，与另外两种最近的二维地图分割方法相比，本方法对地图的区域分割效果有极大的提升，在  $\frac{5}{6}$  的数据集中达到了 97% 以上的准确率。

本文提出了创新拓扑的表示——二维的区域图和三维的多层次拓扑-度量图，均以区域为节点，并提出了用于从地图提取区域的创新的地图分割方法。这两种拓扑表示方法及其构建算法丰富了对拓扑地图的理论研究。随即作者通过路径规划应用展示了拓扑地图在应用的计算速度上提供的巨大优势。基于该区域图，作者提出两种地图匹配方法，在准确率和计算速度上都优于同类的最新方法。这进一步展示了我们的区域图在机器人领域的实用性，为区域图的应用提供了技术支撑。

**关键词：**拓扑地图，地图分割，地图匹配，路径规划

## Abstract

Maps are widely used in the field of robotics. The most commonly used map representations are occupied grid maps and point clouds. These two types of maps use tiny units (cells, points) representing the environment's state as the graph's nodes. The higher the map resolution, the more accurate it is to describe the environment, but the more time-consuming the searching and computation are in the applications. A good solution is to extract topological representations from maps, which can improve the computational speed in many applications, such as semantic map construction, global path planning and map matching in large environments, because each node in the topological map is no longer a specific point in the environment, but an area, a space, or a place. In human intuition, areas are better than points to describe a place. For example, when referring to "Times Square", people often think of the area of the square rather than a specific point in the square. Reasonable segmentation of the map is the basis for constructing area-based topological maps.

The dissertation firstly proposes an innovative topological map representation, Area Graph, and a method for extracting this topological map from a two-dimensional indoor grid map. An area Graph is a topological map whose nodes are areas. In order to segment the map into meaningful areas, the author proposes a map segmentation algorithm based on Voronoi Diagram and room detection. By comparing our area generation method with another two map segmentation methods, the experiments show that the segmentation results of our method are closer to the manually given reference results. Moreover, the segmentation experiments on maze maps show that our topological representation is more suitable for maze applications.

Then the author proposes a global path planning method based on the Area Graph. This method constructs the Passage Graph from the Area Graph and then uses the nodes of the Passage Graph as the search space for the path planning algorithm. By comparing the time to search for paths on the two-dimensional grid map and on the Passage map extracted from this grid map using the A\* search algorithm, respectively, the path

planning computation time of the latter is reduced by two orders of magnitude compared to the former. Path planning is an important application in the field of robotics. The author uses this application to demonstrate the great advantage of topological maps in reducing computation compared to grid maps.

Another important contribution of this dissertation is that it proposes two map matching methods based on the topological representation, namely Hypothesis Clustering and Neighbor Growing, which are applied to map matching with and without global consistent transformation between maps, respectively. The Hypothesis Clustering method uses features extracted from areas to perform area matching between two maps, and votes the correct transformation between maps by clustering the potential correct matchings. The main idea of the Neighbor Growing method is to add the successfully matched areas between the maps one after another to the common subgraph through the adjacent relationship between the nodes, so as to correspond different parts of the pair of maps which without consistent transformation between. For the two methods, the author conducts comparison experiments: 1) For the matching between maps with global consistent transformation, the author compares the Hypothesis Clustering method with a common image registration method, the point cloud registration method ICP (Iterative Closest Point), and a state-of-the-art map matching methods based on region matching. The experiment results show that the Hypothesis Clustering method has higher map matching accuracy than other methods, especially when maps are in different modalities. 2) For broken maps, there is no consistent global transformation between input maps. The authors compare the Neighbor Growing method with a recent non-rigid point registration method. Experiments show that the Neighbor Growing method is better to match the different parts of the maps.

In addition to Area Graph, the topological representation of two-dimensional maps, this thesis proposes a hierarchical topometric representation of 3D maps with multi-storey building point clouds as input. The main contribution of it is to solve the problem of region segmentation for 3D maps and extracting the topological representation. Secondly, it is a hierarchical topological structure: its high-level topological nodes are the parents of the low-level topological nodes. The topological structure mainly includes

three levels, which respectively represent the three-dimensional map structure: storey, region and volume. In the region level graph, rooms and corridors are the nodes of the topology, and doors are the edges to connect nodes. The core of this algorithm is to obtain the areas by merging the free volumes and detecting the doors. In addition, this method uses the Area Graph generation algorithm to further segment these regions, which makes the room segmentation more robust and improves the practicability of robot navigation in the corridor regions. Experiments show that compared with the other two state-of-the-art two-dimensional map segmentation methods, our method has greatly improved the map segmentation, and has reached more than 97% in accuracy in the dataset of  $\frac{5}{6}$ .

The 2D Area Graph the hierarchical 3D topometric representation proposed in this dissertation takes areas as nodes, and provides innovative map segmentation methods for extracting areas from the maps. The representations and their construction algorithm enrich the theoretical research of topological maps. Afterward, the author demonstrated the large advantage provided by the topological map in the computation speed through the path planning application. Based on our topological map, the author proposes two map matching methods, which outperform the state-of-the-art methods in both accuracy and computation speed. This further demonstrates the practicability of our area map in the field of robotics and provides technical support for the application of the Area Graph.

**Keywords:** Topological Map, Map Segmentation, Map Matching, Path Planning



<b>目 录</b>	
<b>第 1 章 绪论</b> .....	<b>1</b>
1.1 研究背景和意义 .....	1
1.1.1 Area Graph 简介及其研究意义 .....	2
1.1.2 地图匹配算法的研究意义 .....	4
1.1.3 多层次拓扑-度量图的研究意义 .....	5
1.2 相关问题研究现状 .....	7
1.2.1 拓扑地图的构建 .....	7
1.2.2 基于拓扑地图的路径规划与导航 .....	11
1.2.3 地图匹配 .....	12
1.3 主要工作与文章结构 .....	15
<b>第 2 章 地图与拓扑图</b> .....	<b>19</b>
2.1 地图及其应用 .....	19
2.1.1 地图的表示方式 .....	19
2.1.2 地图的使用 .....	21
2.2 图与拓扑图 .....	22
2.2.1 图的基本元素 .....	23
2.2.2 维诺图 (Voronoi Diagram) .....	24
2.2.3 Topology Graph .....	27
2.3 本章小结 .....	33
<b>第 3 章 Area Graph: 从二维栅格地图提取的拓扑表示</b> .....	<b>35</b>
3.1 基本概念及元素介绍 .....	35
3.1.1 Voronoi Diagram .....	35
3.1.2 Topology Graph .....	36
3.1.3 临时图 .....	37
3.1.4 Area Graph .....	38
3.2 Area Graph 的创建过程 .....	39
3.2.1 生成多边形 .....	41
3.2.2 区域的生成 .....	46
3.2.3 创建区域图 .....	49
3.3 参数 $\alpha$ 的确定 .....	49
3.4 与地图分割相关工作的对比实验 .....	52

3.4.1 在普通的室内地图上的对比实验 .....	52
3.4.2 迷宫地图对比实验 .....	55
3.5 开源代码 .....	59
3.6 本章小结 .....	59
<b>第 4 章 基于区域图的路径规划方法 .....</b>	<b>61</b>
4.1 Passage Graph 的创建 .....	61
4.1.1 算法原理 .....	61
4.1.2 算法描述 .....	63
4.2 效率展示实验 .....	65
4.3 本章小结 .....	68
<b>第 5 章 基于区域图的地图匹配 .....</b>	<b>69</b>
5.1 算法简介 .....	69
5.1.1 算法流程 .....	69
5.2 算法的具体实现 .....	71
5.2.1 区域特征的提取与成本计算 .....	71
5.2.2 区域间的匹配成本 .....	75
5.2.3 方法一：启发式聚类法 .....	76
5.2.4 方法二：生长邻居法 .....	79
5.3 地图匹配实验 .....	81
5.3.1 数据集与评价标准 .....	82
5.3.2 特征选择实验 .....	83
5.3.3 ICP 误差对区域匹配成本的影响 .....	86
5.3.4 与最先进的地图匹配算法的比较实验 .....	86
5.4 本章小结 .....	92
<b>第 6 章 对三维环境的多层次拓扑-度量图表示 .....</b>	<b>95</b>
6.1 多层次拓扑-度量图结构总览 .....	95
6.1.1 多维度的度量信息 .....	95
6.1.2 多层次的拓扑结构 .....	96
6.2 构建算法 .....	97
6.2.1 点云预处理及楼层检测 .....	98
6.2.2 纵列的生成 .....	98
6.2.3 空间块与通道的生成 .....	99
6.2.4 区域的构建 .....	100
6.2.5 用 Area Graph 生成算法划分子区域 .....	103

6.3 实验评估 .....	105
6.3.1 算法性能实验 .....	105
6.3.2 对比实验 .....	105
第 7 章 总结与展望 .....	109
参考文献 .....	113
致谢 .....	123
作者简历及攻读学位期间发表的学术论文与研究成果 .....	125



## 图形列表

- 1.1 (a) 根据 Schwertfeger 等 (2015) 的方法由维诺图的修剪生成的 Topology Graph。Dead end edge 显示为浅蓝色线条，两端都连接非 dead-end 顶点 (junction) 的边显示为深蓝色，Topology Graph 的边大致描绘了地图的骨架。(b) 附着在 Topology Graph 的边上的多边形，是合并前的区域。(c) 使用 Alpha shape 检测从二维栅格地图中检测地图边界 (蓝线) 和房间 (红色)。(d) 属于同一 Alpha shape 内的多边形已被合并为一个区域。..... 3
- 1.2 区域图构建 (红色方框)、基于区域图的路径规划 (绿色方框) 与地图匹配 (蓝色方框) 的简要算法流程图，描述了这三项工作之间的关系。对于每幅给定地图，都会生成一个区域图表示。通过从区域图生成的通道图，可进行基于拓扑结构的路径规划。从这些区域中提取特征，并计算匹配成本进行区域匹配后，可通过启发式聚类法或生长邻居法这两种匹配方法进行地图匹配。..... 16
- 2.1 一个由二维栅格地图生成的维诺图的一部分区域的放大细节。这个维诺图是从网格地图 (被占用的单元格) 生成的，并选取其一小块区域 (绿色的小方块) 进行放大。图中展示了维诺图的顶点、维诺图的边和维诺图的区域 (face) 的示例。在邻居单元之间也会创建维诺图的边，此情况下生成的维诺边会穿过墙壁。..... 26
- 2.2 只修剪过一次的维诺图。为了最终得到一个简洁的 Topology Graph，上图中还有很多尽边 (浅蓝色) 需要被删除。射线 (浅绿色) 是指向无穷远的，因此无法从中创建出多边形，它们也将被移除。..... 28
- 2.3 Alpha Shape 通常用于检测点之间的开放空间。 $\alpha$ -shape 的大小取决于参数  $\alpha$ ，它是圆盘的平方半径，左图显示了点集内部的圆盘 (省略了外部圆盘以节省空间)。Alpha shape 的创建是通过将圆盘接触到的点用线段连接。右图为输出的  $\alpha$ -shape，外部  $\alpha$ -shape 显示为蓝色，内部  $\alpha$ -shape 显示为红色。..... 30
- 2.4 2D 栅格地图的预处理。(a) 未经去噪处理的原生二维栅格地图 (已将未知区域 (背景) 设置为白色)。(b) 经过 CGAL 的 REMOVE\_OUTLIERS 函数进行噪点去除的地图。(c) 经过去家具杂物的地图。(d) 使用 alpha shape 检测算法检测地图边界 (显示为红色)。..... 32

- 3.1 左示意图：绿色和紫色短箭头是相反方向的 Voronoi 半边。每个 Voronoi 半边都关联一个 face，其中仅包含一个 site。右示意图：按顺时针顺序将半边与 site 连接以创建半多边形。一对半多边形组成一个多边形，附加在其对应的 Topology Graph 边上。半多边形生成步骤（合并 face 的步骤）是在创建 Topology Graph 边  $e^T$  的 EDGE\_SKIPPING 步骤的同时完成的。创建的多边形在 CREATEPOLY 步骤中完成。 ..... 36
- 3.2 Topology Graph (a) 与 Area Graph (b) 生成流程。对比两个拓扑图的生成步骤，图 (b) 中的蓝色框是 Area Graph 生成算法相对于 Topology Graph 生成算法的改进部分，蓝色字体是完全新增的步骤。 ..... 40
- 3.3 将附加于尽边的多边形合并到其相邻半多边形中的两种情况（这里只显示  $e_{dj}$  的相邻半多边形而不是整个相邻的多边形）。其中，红点代表 Dead End Vertex，与 Dead End Vertex 相连的边就是尽边，紫色点代表 Junction 顶点。(a) 当其相邻边不是尽边，将尽边多边形  $poly(e_{dj})$  与其后置半多边形  $hpoly(h_{jn})$  合并。(b) 将尽边多边形  $poly(e_{dj})$  合并到它的前置半多边形  $hpoly(h_{ij})$  中，因为  $h_{dj}$  的后置邻居是一条尽边。... 43
- 3.4 (a) Topology Graph 的边附加了多边形，显示为彩色的区域。图中的 Dead End Edge 显示为浅蓝色线条，连接 junction 顶点的边显示为深蓝色，这些边形成地图的骨架。图上附加的多边形是合并之前的区域。(b) 使用 Alpha Shape 检测以从网格地图中检测地图边界（显示为蓝线）和开放空间（房间）（显示为红色斑点图案的多边形）。(c) 相同 Alpha Shape 内的区域已进行了合并。 ..... 44
- 3.5 上图放大了一条边及其多边形与两个 alpha shape 的边界相交的例子，图中红线是用于检测区域的 alpha shape 的边界，蓝色的是图  $G^M$  中一条横跨多个区域的边，此边将被进行两次切割生成三条分属于三个区域的新边及三个相应的新多边形。 ..... 48
- 3.6 根据不同参数  $W$  进行分割的结果使用 Mielle 等 (2017) 的标准 MCC 进行分割质量评估。图中横轴是参数  $W$ ，它决定了我们算法中的参数  $\alpha$ ，纵轴是由该参数产生的分割结果的 MCC 分数。对于每条曲线，在  $W = W_d$ （最宽的门宽）和  $W = W_c$ （最窄的走廊宽度）处的 MCC 分数分别用“x”和“o”标记。 ..... 50
- 3.7 为 Alpha Shape 检测算法设置三个不同的参数  $W$  值产生的对地图  $lab_d$  的分割结果。(a) 设置  $W = 0.5m$  导致地图欠分割，结果的 MCC 得分分别为 0.03; (b)  $W = 0.8m$  时分割结果良好，结果的 MCC 得分分别为 0.74; (c)  $W = 8.55m$  时 alpha shape 没有检测到任何房间，导致地图被过分割，结果的 MCC 得分分别为 0.23。 ..... 51
- 3.8 三种地图分割方法的结果对比。分数高的则效果好。 ..... 52
- 3.9 比较三种不同的地图分割方法的分割结果。上图将地图中分割好的不同区域涂为不同的颜色以便区分。此外，每个结果与参照结果（真值）的对比获得的 MCC 分数展示在每幅子图的右上角。 ..... 57

3.10 比较不同方法对迷宫地图的分割，遵循图 3.9 的风格进行结果展示。我们的算法使用了一个很大的参数 $W$ 值，有效地禁用了 alpha shape 房间检测算法。 .....	58
4.1 以图 3.4 的地图为例，展示生成 Passage Graph 的过程的示意图。(a) 根据 Topology Graph 的边生成的多边形。(b) 由 Alpha Shape 检测到的同一房间内的区域已被合并。(c) 通道图的边显示为红线。从图中可以看到深蓝色的大中心区域的所有通道之间是如何相互连接的，通道图的每条边都关联一条从 Topology Graph 的边获得的路径并记录路径的长度。(d) 图中的红色加粗线条展示了基于拓扑表示的路径规划算法搜索到的从起始点到终点的一条路径，显然它是通过将起始点和终点连接到线下保存的拓扑骨架路径上形成的。 .....	64
4.2 基于 Area Graph 拓扑表示构建的 Passage Graph 进行全局路径规划。随机给定起点与终点，由我们的算法获得的路径显示为加粗的红色线条。 .....	65
4.3 比较在栅格地图和通道图上的 $A^*$ 路径规划算法。(a) 在栅格地图上的 $A^*$ 算法搜索结果。(b) 在 Passage Graph 上的 $A^*$ 搜索结果。 .....	68
5.1 输入是由两个机器人分别对同一环境的扫描地图，两个地图之间有 $43.73^\circ$ 的相对旋转。第二行是由两个栅格地图生成的区域图。对于我们提出的启发式聚类方法，蓝线显示由我们的算法自动进行的区域匹配。该方法由该匹配结果估计得旋转变换为 $43.99^\circ$ 。对于我们提出的生长邻居法，蓝线显示最大匹配子图中的区域对，由此估计出的旋转变换为 $44.74^\circ$ 。对于有损坏的地图，我们不计算地图间的变换。可以看见使用两者中的任一个方法，两个粒子图都可以被很好地合并为一个完整的地图。 .....	70
5.2 机器人生成的地图（左）与相应的平面图（右）相匹配。地图分为不同的区域，用不同的颜色表示。通道点以黄色圆点突出显示，我们将属于同一区域的通道用红线连接起来，通道的距离特征是根据红线的长度计算的。 .....	73
5.3 (a) 计算两个区域之间的变换。(b) 将具有给定角度的区域 $A_i$ 及其匹配段的中点转换为其匹配区域 $B_j$ 。段落以红色突出显示。 .....	78
5.4 对一个主要包含走廊的环境的损坏地图的分割。地图拐弯处附近有一个区域重叠（用圆圈圈出）。 .....	82
5.5 实验结果展示了不同算法的匹配结果。结果表明，与其他算法相比，我们的算法具有更好的表现。 .....	90
5.6 我们的生长邻居法与非刚体点配准方法 (PRM-L2E) 的比较。我们的结果中相同颜色的区域表示这些区域属于同一个子图。 .....	93

6.1	输入三维点云 (a), 本算法产生不同维度的输出: (b) 0D 拓扑地图; (c) 使用记录了坐标的三维点作为节点的 1D 地图; (d) 以区域为节点的 2D 地图; (e) 记录空间块信息的 3D 地图。 .....	96
6.2	(a) 室内空间的纵列 ( <i>column</i> ) 示意图 (图中仅显示了三条纵列), 实际上空间中布满纵列。(b) 具有相似顶高的纵列组成一个空间块 ( <i>volume</i> )。空间块之间的接触面作为通道 ( <i>passage</i> )。(c) 图中展示了三个区域, 其中区域 2 是门所处的区域。区域 3 代表的是房间, 由图 (b) 的 volume 1 和 volume 2 合并而成。三个区域之间有两个通道。 .....	97
6.3	左: 点云侧视图及沿着点云 $Z$ 轴统计点数的直方图。中: 从上到下窗口大小分别为 $c \in \{2, 4, \dots, 10\}$ $cm$ 的窗口滤波结果。右: 检测到的地板与天花板 (标注为蓝色线条)。 .....	99
6.4	(a) 空间块级的拓扑图, 其中体积大于 $20m^3$ 的空间块的节点用蓝色表示。(b) 从三维体素占用地图生成的空间块。 .....	101
6.5	(a) 区域级的拓扑图。地图中的通道被标记为绿色。(b) 从三维体素占用地图生成的区域。 .....	101
6.6	从空间块生成区域。 .....	102
6.7	阈值 $a_{th}$ 对种子数的影响, 蓝色折现 (橙色点) 和橙色折现 (蓝色点) 分别为滤除多余种子前、后的种子数量, 红色水平线为真实房间区域数。 .....	103
6.8	(a) 列举了几个大区域使用 Area Graph 生成算法划分出子区域的示例。(b) 将一个大区域的二维分割结果应用到三维地图中。 .....	104

## 表格列表

2.1 用于 Topology Graph 生成的函数及其描述。一些函数在算法 1 中有出现。 .....	29
3.1 用于生成区域图的函数及其描述，函数的使用展示于算法 1 中。 ....	42
3.2 测试地图分为四种：(1) 完整的地图，家具少，很少有大的开放空间；(2) 完整的地图，家具少，且包含大的开放空间；(3) 区域中有很多家具的地图；(4) 不完整的地图。然后，对不同方法我们计算其对于不同类型地图的分割结果的 MCC 平均分数。 .....	55
3.3 三种地图分割方法的运行时长比较。除了计算它们的平均运行时间外，该表还显示了这些算法分割耗时最短或最长的地图。 .....	56
4.1 不同路径距离下基于网格地图的 $A^*$ 与基于通道图的 $A^*$ 的规划时间比较。 .....	67
5.1 匹配算法分别使用每项特征距离逐一运行，计算它们的区域匹配正确率以找出每项特征的影响以决定最后在我们的区域匹配算法中使用哪些特征。此外，我们比较了考虑和不考虑邻居的匹配成本两种情况下的表现。 .....	84
5.2 对于 Bormann、Carpin 和 Birk 的地图数据，匹配算法启发式聚类法 (HC) 和生长邻居法 (NG) 分别在有和没有 ICP 误差作为特征距离的情况下运行。我们看到 ICP 误差并没有改善匹配。由于 ICP 计算非常耗时，我们决定在区域匹配中不使用 ICP 误差。 .....	87
5.3 地图匹配结果比较。我们的启发式聚类法与基于 SIFT 特征的图像匹配法、ICP 点云配准法和 Shahbandi 等 (2019) 的工作进行了比较。我们的方法获得了最高的匹配正确性，并且比最先进的方法 (Shahbandi 等, 2019) 更快。 .....	88
5.4 我们的邻居生长 (NG) 方法与 Ma 等人的工作 (RPM-L2E) 比较对损坏地图的匹配能力，通过两个地图之间正确匹配的块的百分比来评估。 .....	91
6.1 不同分割方法在不同数据集上的 MCC 评估结果。 .....	106
6.2 对不同数据集的评估实验。 .....	107
6.3 与两种最先进方法 (Area Graph generation algorithm & MAORIS) 的对比实验。 .....	108



## 符号列表

## 符号

<b>Symbol</b>	<b>Description</b>
$G$	Topological graph
$V$	Vertex set of the graph
$E$	Edge set of the graph
$P$	Polygon set of the graph
${}^{VD}$	superscript for Voronoi Diagram elements
${}^T$	superscript for Topology Graph elements
${}^A$	superscript for Area Graph elements
$v$	Vertex of the graph
$h$	Half edge of the graph
$e$	Edge of the graph
$hp$	Half polygon of the graph
$p$	Polygon of the graph
$\mathcal{G}(\mathcal{V}, \mathcal{E})$	3D topological graph (volume level)
$\mathcal{V}$	Vertex set of the graph $\mathcal{G}$
$\mathcal{E}$	Edge set of the graph $\mathcal{G}$

## 算子

<b>Symbol</b>	<b>Description</b>
$deg()$	degrees of a vertex
$poly()$	polygon attached to the edge
$hpoly()$	half polygon attached to the half edge

## 缩写

SLAM	Simultaneous Localization And Mapping
------	---------------------------------------

VD	Voronoi Diagram
ICP	Iterative Closest Point

## 第1章 绪论

### 1.1 研究背景和意义

地图不仅在人类世界被广泛使用，也是机器人执行任务的重要工具。人类使用地图进行认路、寻路。随着机器人技术的发展进步，机器人可以执行越来越多的自主任务，机器人使用地图进行定位、导航、避障、巡逻等。随着传感器技术的发展，地图的精度越来越高，可以更好地描述环境细节；随着建图技术的发展，地图可以记录更大规模的环境信息。当前机器人最广泛使用的环境表达形式（地图）是栅格地图和点云，这类地图可以看作以许多描述环境细节的微小单元（单元格或点）作为节点的拓扑图，地图精度越高、规模越大，则节点越多，算法在地图上搜索或遍历时的计算量就会变大。因此，许多应用地图的算法正面临着计算量过大从而导致计算速度达不到任务要求的问题。

本文提出的以区域为节点的拓扑地图——区域图 (Area Graph) 和多层次三维拓扑-度量图可以很好地解决以上问题。因为环境中的房间、走廊结构通常是固定的，由于本文提出的拓扑地图以房间和走廊结构这类有意义的区域为节点，随着传感器的精度升高，其构建的地图的区域数量仍然是不变的，而且总是比栅格地图或点云的单元数少好几个数量级。随着地图规模的变大，以区域为节点的拓扑图中的区域数亦不会像栅格地图中的单元格数一样急剧上升而造成计算量的大幅上升。在本文接下来的章节中，作者在介绍了本文提出的拓扑地图的构建算法后，将通过一个机器人应用的例子——路径规划来体现拓扑地图相比于栅格地图在减少计算时间上的优势。

拓扑地图在机器人学领域的应用主要包括在环境中的全局路径规划与导航、地图评估、地图匹配等。区域图作为一种创新的拓扑地图表示，除了可用于传统的地图算法外，本文针对其特有的拓扑-度量结构提出了创新的基于拓扑地图的地图匹配算法。线上地图匹配主要用于定位，由于定位期间传感器数据的更新，机器人可以基于概率的算法来匹配其局部传感器数据到全局地图，该领域的研究已较为成熟。而离线地图无法通过时间上不断更新的数据进行地图匹配，离线地图匹配与线上地图匹配没有共通的算法。离线地图匹配在后文中将直接称为地图匹配。这里的地图匹配不同于线上建图应用中为了进行机器人定位而进行

的非常小范围的局部传感器信息与先前信息（已构建的地图）进行匹配，而是指将两张完整的、或已包含部分完整区域的全局地图进行匹配，该类匹配的意义主要是应用于地图合并、地图更新、地图评估等应用。由于地图存在特征稀疏、纹理单一、自相似性强等特点，传统的图像匹配迁移到地图匹配的效果并不好，因此，目前地图匹配仍然是个亟待研究的问题。本文提出的地图匹配方法结合了传统的图像匹配中的特征匹配的方法和拓扑地图的图结构的优势，使地图匹配的准确率和计算速度相比于之前的方法都有了显著的提升和进步。

本章将介绍拓扑地图与基于区域图的地图匹配的研究意义，然后介绍二维、三维拓扑地图的构建、基于拓扑地图的路径规划和地图匹配的相关研究工作，最后介绍全文的主要工作与文章结构安排。

### 1.1.1 Area Graph 简介及其研究意义

如前所述，地图和地图的构建对于几乎所有移动机器人应用程序都是必不可少的。地图可以用不同的方式表示，在机器人技术中被最广泛使用的是 2D 网格地图和 3D 点云。拓扑地图从地图的细节（像素、点）中抽象出更高级的表示（例如地点）来构建拓扑结构，因此提供了更紧凑的表示，使应用地图的算法的计算速度更快。拓扑地图是以地点或者区域作为顶点，且以这些地点之间的连接为边的图。对于许多机器人应用程序，拓扑地图中会记录一些度量信息 (metric information)，例如，使用顶点的二维地图坐标和边的路径信息，因此有时拓扑地图也称为拓扑-度量图 (topometric map)。

Area Graph（区域图）的创建和应用是本人博士阶段论文的主要贡献。Area Graph 是我们提出的一种新颖的二维拓扑表示，其顶点为区域，边为区域的连接，其整体是一个拓扑图结构，但图中每个顶点和每条边都记录了相应的环境度量信息，因此也是一种拓扑-度量图。Area Graph 的创建是基于 Schwertfeger 等 (2015, 2016b) 中呈现的 Topology Graph 结构发展而来，并且使用  $\alpha$ -shapes 的房间检测算法提取环境中的开放区域 Edelsbrunner 等 (1983)。图 1.1 展示了从二维网格地图中提取的 Topology Graph（图 1.1a）以及直接在其上直接附加多边形的效果（图 1.1b）、在网格地图中使用  $\alpha$ -shapes 检测到的开放区域（图 1.1c）、和最终形成的基于地图分割的 Area Graph（图 1.1d）。Area Graph 通过 Topology Graph 结构生成地图骨架来生成初始多边形，然后通过  $\alpha$ -shapes 检测算法检测开放区域来合并多边形，最终生成有意义的地图分割并从中建立拓扑地图。

从根本上说, 作为拓扑表示, Area Graph 的一个基本贡献是它将地图表示为一组互相连接的区域, 可以将其视为地图的分割。与纯粹的分割方法不同, 区域图还明确且可靠地表示区域之间的连接, 我们称之为通道 (*passages*)。这些通道代表了两个接触区域之间的共同边界, 允许移动主体 (人或机器人) 从一个区域转移到下一个区域。使用区域连接处来作为区域间转移的媒介为路径规划算法在该拓扑地图上的应用奠定了基础。在本人最近参与的工作 [He 等 \(2021\)](#) 中, 除了区域分割, 还明确地生成了 3D 地图表示中的通道, 并且该表示利用了 Area Graph 来为其提取更多的拓扑信息。本文提出的区域图是从 2D 网格地图中抽象出来的。与传统的拓扑地图相比, 它的表示更复杂, 因为顶点是由一个区域来描述的, 而不是仅仅一个点, 这是非常紧凑的拓扑地图表示方式。

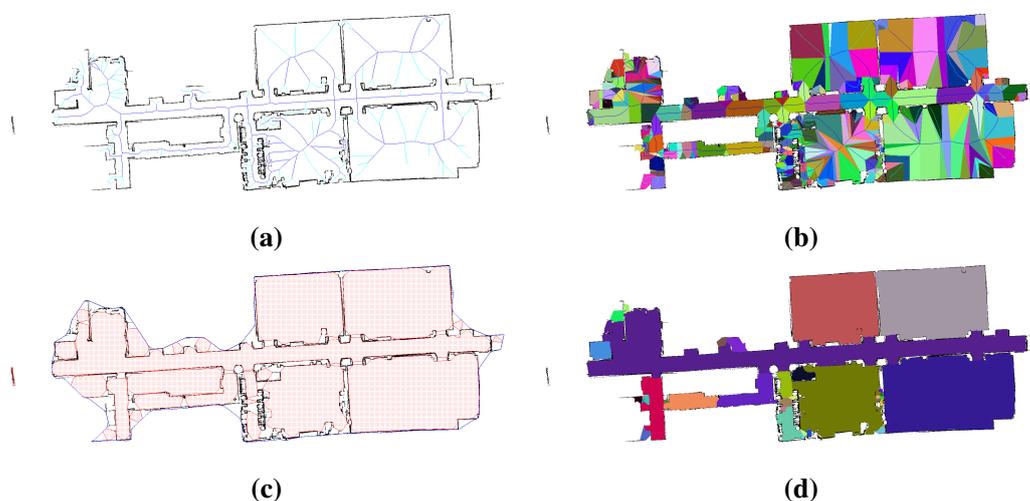


图 1.1 (a) 根据 [Schwertfeger 等 \(2015\)](#) 的方法由维诺图的修剪生成的 Topology Graph。Dead end edge 显示为浅蓝色线条, 两端都连接非 dead-end 顶点 (junction) 的边显示为深蓝色, Topology Graph 的边大致描绘了地图的骨架。(b) 附着在 Topology Graph 的边上的多边形, 是合并前的区域。(c) 使用 Alpha shape 检测从二维栅格地图中检测地图边界 (蓝线) 和房间 (红色)。(d) 属于同一 Alpha shape 内的多边形已被合并为一个区域。

Figure 1.1 (a) The Topology Graph obtained by pruning the Voronoi diagram using the method in [Schwertfeger 等 \(2015\)](#). Dead-end edges are shown as light-blue lines, edges connecting non-dead-end vertices (junctions) are shown in dark blue. The Topology Graph edges describes the skeleton of the map. (b) The polygons attached on the Topology Graph edges are areas before merged. (c) Alpha shape detection is used to detect the boundary (blue line) and rooms (red patterns) from the grid map. (d) Polygons within the same alpha shapes have been merged as an area.

传统的拓扑地图，例如许多基于 Voronoi 图创建的拓扑表示，将地点表示为带有附加坐标的节点，将这些地点之间的连接路径表示为带有附加路径信息的边。从人类的认知来看，用一个区域来代表一个地方是比仅仅使用一个点是更好的选择。因为当人们想到一个地点时，例如一张桌子、一个房间、一座建筑物、一个校园、一座城市等，他们会直觉地想到某个区域——而不仅仅是一个点。从机器人学的角度来看，使用区域表示的优势在于，对于给定的 2D 坐标（例如机器人当前位置或目标位置），机器人可以立即清楚它属于哪个区域/节点——而不是只知道一个坐标。区域的多边形边界也隐式地编码了障碍物的位置，因为根据定义，多边形边界的每个不是通道的部分都是不可穿越的障碍物。

Area Graph 的另一个优点是它的边表示。在传统的拓扑-度量图中，连接两个房间的边很可能从一个房间的中心（节点）连接到另一个房间的中心。但这不是很直观。对于人类的直觉来说，若让人类解释两个房间之间的联系，这个人很可能会说门连接了这两个房间。而 Area Graph 的边，代表的是通道，也就是代表这扇门。通道的宽度和位置也被编码在区域图的度量信息中。该信息可用于对从一个房间/区域到另一个房间/区域的路径进行规划。

因此，提出区域图 Area Graph 的表示方式的动机有两个：1) 我们认为这种表示更接近人类对地点及其联系的直觉；2) 其在机器人技术的应用方面有优势。本文的接下来章节中将详细介绍两种针对该拓扑表示的应用，即路径规划和地图匹配。相应地，提出区域图包含的创新点主要有两个：1) 提出了创新的以区域为节点的拓扑地图表示方式；2) 提出了创新的二维地图分割算法，且通过实验展示了其优越的分割性能。

### 1.1.2 地图匹配算法的研究意义

地图匹配是机器人领域中一项有趣且具有挑战性的任务。将机器人地图与不同模式的地图进行匹配或合并对许多机器人应用来说都有应用意义。举几个例子。在环境探索中，机器人传感器地图和平面设计图的匹配可以作为机器人导航的依据，提高探索效率 (Shahbandi 等, 2019)。匹配从不同楼层扫描的地图以构建建筑物的多层地图。解决因传感器的限制和 SLAM 算法 (Setalaphruk 等, 2003; Grisetti 等, 2007)，从机器人传感器数据（例如占用网格图）生成地图总是会失去对环境建图的一致性的问题。利用先验地图可以提高传感器地图的全局一致性，而先验地图的全局信息也可以提高 SLAM 应用中的定位精度，先验地图的

使用就要用到地图的匹配。此外，将机器人传感器地图与已知设计平面图匹配可以将设计平面图的语义信息提取到机器人地图，这对使用自然语言 (Kakuma 等, 2017) 进行机器人导航有很大帮助。例如，一个已存在的地图中已经知道其语义标签，则机器人可以将自己在另一个地图中进行语义定位。匹配不同模式的地图是机器人地图评估的一个关键问题 (Schwertfeger 等, 2015)。地图匹配的另一个重要应用是实现多机器人系统的地图融合，帮助多机器人系统中的协作 (Carpin, 2008)，或者提高人机交互的效率 (Georgiou 等, 2017)。

地图匹配问题还存在以下挑战：首先，1) 匹配两个不同地图不能使用图像对齐的传统方法。在实践中，机器人传感器扫描的地图可以被视为一张图片。然而，当一个占用网格地图被解释为图像时，它的特征远不如一般的数字图像，例如，地图无色彩、边角特征单一、纹理不丰富。因此，成熟的图像配准算法（例如，基于特征的方法）不适合机器人技术中的地图匹配问题。2) 匹配两个不同模态的地图很难使用图像匹配或以前的一些工作 (Birk 等, 2006; Carpin 等, 2004, 2005; Carpin, 2008) 来完成。此外，这些方法无法处理由不同模态引起的尺度变化问题。3) 由于其中一些方法是基于刚性变换的假设，如果地图发生变换失真的问题，则难以通过估计一个一致的地图变换来进行地图对齐。4) 在大规模地图之间进行地图匹配是一种挑战。耗时一直是二维地图匹配中的一个棘手问题，并限制了其应用。尽管 Shahbandi 等人的先进算法 (Shahbandi 等, 2019) 在一些具有挑战性的场景中具有良好的性能，但它需要大量的计算资源，这使得该方法难以应用于大规模场景。

在本文的工作中，作者提出了一种基于区域分割的拓扑表示的匹配算法来解决地图匹配问题的挑战，特别是解决耗时的问题。工作的主要贡献包括：a) 减少了地图匹配的时间。而且 b) 我们的算法可以处理不同的模态和尺寸的地图的匹配问题，c) 作者提出的其中一种方法不依赖于刚性变换假设。本文的算法已经在不同模式和场景的各种地图上实施测试。我们还与最先进的方法进行了比较，实验结果显示了我们算法的良好表现和计算效率，尤其是对大规模场景地图的匹配。

### 1.1.3 多层次拓扑-度量图的研究意义

Area Graph 是以区域为节点的拓扑-度量表示，该表示可以很好地将单层建筑的二维栅格地图抽象为对环境的一个拓扑-度量图。受限于其输入的二维地图

仅包含单个高度的环境信息，不能完整地描述建筑结构、易受环境中家具等杂物的影响，Area Graph 的节点区域仍有过分割的现象，与房间结构不完全吻合。因此，我们提出了针对三维地图提取拓扑表示的方法，拓扑图中记录三维度量信息，用以解决三维地图数据量过大造成的计算效率低下的问题。

考虑到以区域作为拓扑节点的好处，该三维拓扑表示仍使用符合语义的（三维）区域（例如，房间和没有分叉口的走廊）作为该拓扑图的节点。为此我们提出了创新的三维地图分割算法。在对三维地图进行区域分割时，通过检测天花板、门、墙壁等室内环境中特有的分隔结构，可以使算法对地图的分割与建筑本身的语义结构更吻合。

Area Graph 仅能以描述单层建筑结构的单幅二维栅格地图为输入，生成单个拓扑图结构，仅表示单层建筑环境。为此，我们提出了从描述多层建筑结构的三维地图生成多层次拓扑-度量表示的方法，作为一种改进的方法。在这样的结构中，更高级的拓扑图中的节点是拓扑子图，代表更大范围、更高级的区域。例如，在一栋建筑的多层拓扑图中，最高级拓扑图中的一个节点包含的信息是建筑中某一层生成的拓扑地图，其度量信息是一整层的建筑区域。而单个楼层生成的拓扑图的节点是代表房间和走廊的区域。而其中的完整走廊区域（包含分叉的贯通走廊）还可通过再分割创建一个拓扑子图，其每个节点是一段没有分叉口的走廊区域。使用多层拓扑图进行对多层建筑的表示，其意义不仅是（1）在空间上具有良好的解释性、更符合人类语义，而且（2）这样紧凑的表示提高了算法应用的计算效率，其多层结构使算法可以进行更快速高效的搜索和索引，进而允许机器人将其用于实践。

总的来说，多层次拓扑-度量图表示相对于 Area Graph 表示有以下改进：（1）该拓扑-度量图是对三维地图的抽象，最底层的拓扑区域节点包含的是三维区域空间的度量信息，对环境的描述更完整，解决应用三维地图时计算量过大的问题。（2）地图分割过程中通过环境的三维信息提取例如门、墙面、天花板等可用于识别房间的建筑结构，使地图分割结果与房间结构更吻合。（3）多层次拓扑-度量图可以表示多楼层建筑结构，允许机器人对多层建筑环境进行计算和应用。（4）多层次的拓扑结构，以低层拓扑子图作为高层拓扑图的节点，使对环境的表示结构更紧凑，进一步缩小了应用的搜索空间。

## 1.2 相关问题研究现状

在本章作者将介绍关于拓扑地图的研究现状，为此，亦会介绍地图分割算法的研究现状。随后，作者将介绍拓扑地图在路径规划和地图匹配这两方面的应用上的相关研究。

### 1.2.1 拓扑地图的构建

拓扑地图是对地图的抽象表示，其构建方法可以分为线上和线下两种。线上方法在对环境进行探索和建图时同步地进行拓扑地图的创建和更新，该类拓扑图常被称为增量式创建的拓扑地图，常对线上的（实时的）传感器数据有要求，且对执行任务的机器人有在线计算能力的要求，线上拓扑地图的创建在环境探索——尤其是无人机对三维环境的探索——这类应用上有重要的作用。线下拓扑地图的创建相对于线上构建拓扑图，减轻了线上计算量，通常也不要求记录线上实时的传感器数据，而是通过已经创建完全的全局二维或三维地图直接创建拓扑表示。线下创建的拓扑地图可用于全局路径规划、地图评估等需要全局环境信息的应用。

#### 1.2.1.1 从二维信息提取拓扑地图

二维地图是地面机器人与环境互动的重要工具，从二维地图中提取拓扑地图可以使这些应用的计算量下降、计算速度更快，因此，在机器人导航、定位等方面有重要的应用。同时，相比于构建三维拓扑地图，构建二维拓扑图的过程耗费的计算资源也更小。拓扑地图的构建分为线上和线下两种，分别适用于线上探索应用和全局应用。

线上拓扑地图主要是通过增量式的构建方法，在创建包含环境细节的栅格地图或点云的时候就同步地创建对应的拓扑地图。这种创建方式常常要求机器人的线上位姿、轨迹信息，适用于同步定位与建图、探索未知环境这类机器人未获得全局环境信息的应用场景。

Ge 等 (2011) 提出名为同时路径规划和拓扑映射的方法，并发、增量地构建地图，将探索环境的全局地图编码在层次拓扑图中，通过嵌入空间树，将环境中的自由空间描述为一组节点和图。该方法严格利用拓扑地图而非网格地图来解决未知环境中的路径规划问题。相似地，为了应用于地面机器人的导航和探索，也有一些研究是针对于二维环境的线上拓扑地图的构建。Yuan 等 (2019) 的算法

可以使地面机器人在为环境构建二维栅格地图的同时，基于与地图上障碍物的距离增量式地创建地图的拓扑信息，并根据新增的信息不断调整和更新，形成以环境岔口、角落、走廊尽头为节点的拓扑地图。以上提及的线上拓扑地图构建方法都是使用地点（或是空间中的特定点）作为拓扑图的节点。

离线构建的拓扑地图根据表示方法的不同主要分为两种情况，一种是使用特定点作为拓扑图的节点，一种是通过将地图进行分割，以区域作为拓扑图的节点。其中，作为拓扑节点的特定点有两种情况，一种是使用地标或人工标注的地点，一种是基于建筑结构的关键点（例如，交叉路口、出入关口等）。

与基于特定点的方法相比，使用基于结构（比如房间结构、走廊结构等建筑结构）的拓扑地图构建方法有几个优点：首先，不必依赖于特征检测，包括人工特征与自然特征。在基于人工特征进行地点识别的情况下，人工特征必须以相对较高的密度放置在环境中。而依赖于自然特征的算法是针对特定应用的，因为这些特征通常仅在特定环境和特定规模中以足够的密度存在。

其次，相比于基于位置的地图（占用栅格地图），基于结构的方法搜索和计算效率更高。因为拓扑信息通常是地图的用途：关注感兴趣的地点，寻找感兴趣的地点之间的连接关系。基于位置的地图构建方法依赖于足够好的地图分辨率来识别特征，而基于结构的表示将从单元格级别抽象出环境结构，生成基于结构的拓扑地图，允许算法使用图中的拓扑信息，将搜索空间降低为拓扑节点，以更高的效率进行搜索和计算。

从占用网格图中提取维诺图来构建拓扑图是一种常用的方法 (Thrun, 1998; Setalaphruk 等, 2008; Beeson 等, 2005; Saeedi 等, 2014a; Friedman 等, 2007)。通过维诺图，可以基于结构自动化地从网格地图提取点或区域作为拓扑节点。目前对该方法的研究大致有三类。第一种方法，Thrun (1998) 和 Kai 等 (2008)，在出入口（例如，门）处将地图划分为不相交的区域。可以通过关键点 (critical point) 找到出入口，其中，关键点在几何上是 Voronoi 的边中比近邻更靠近障碍物的点。这是最经典的方法。第二种方法 Setalaphruk 等 (2008); Schwertfeger 等 (2016a); Wallgrün (2004) 从 Voronoi 顶点构建拓扑图的节点，因此顶点表示地图中的位置或坐标。第三种方法是分割区域作为拓扑节点。例如，Friedman 等 (2007) 使用 Voronoi 随机场来估计 Voronoi 图中每个点的地点类型作为标签，然后根据标签对环境进行分割以构建拓扑图。该方法旨在估计室内环境的场所类型。这些方法

构建拓扑图需要很长时间。

对地图进行分割形成的地图拓扑表示已经在 1998 年就开始使用：Thrun (1998) 提出使用 Voronoi Graphs 的关键点进行区域分割。该文章展示了一种基于 Voronoi 图的分割方法，通过关键点产生区域单元来创建拓扑表示，每个区域单元表示房间或房间的一部分。Kai 等 (2008) 中提出了基于 Voronoi 图的分割，并进行了一些优化以使算法仅在真实的门处选择关键点。并且为了分割得到完整的房间，该算法还使用一些启发式方法来合并小区域。Bormann 等 (2016) 介绍了几种流行的房间分割方法：形态分割 (Fabrizi 等, 2002; Buschka 等, 2002)、基于距离变换的分割 (Diosi 等, 2005)、图形分割 (Brunskill 等, 2007; Zivkovic 等, 2006)、基于特征的分割 (Ekvall 等, 2007; Mozos 等, 2007) 和基于 Voronoi 图的分割 (Thrun, 1998; Beeson 等, 2005; Kai 等, 2008)，并比较了这些方法的分割结果。Bormann 等 (2016) 的对比实验结果表明，基于 Voronoi 的方法的分割结果与理想分割结果的逼近度最高。Bormann 在文中使用来自 Thrun (1998) 的算法实现了基于 Voronoi 的地图分割，并添加了启发式的区域合并算法。作者提出的地图分割方法也包含了区域合并算法，但是不是根据启发式，而是通过检测开放空间来寻找房间等有意义的区域。

MAORIS (Mielle 等, 2017) 通过计算距离图像来分割地图。距离图像中的每个像素都有一个值，表示该像素到与其最近障碍物的距离，具有相似值的像素点进行合并。这种方法有助于防止走廊的过度分割，但容易受到沿墙壁的障碍物突然变化的影响（例如平整的一面墙壁前有一根柱子，则该方法容易产生过分割）。

近年来，由于深度学习的火热发展，也产生了一些使用深度卷积神经网络进行室内地图分割的方法，例如 Foroughi 等 (2021)。这类算法的地图分割结果与理想分割结果相似度通常较高，但严重依赖于可用的训练数据。因为这类方法使用有监督训练，需要使用人类标记好的数据集进行训练。另一种深度学习分割方法 Zheng 等 (2021) 通过距离变换来进行分割，然后利用神经网络将地图标记为“房间”、“走廊”和“门口”等类别。

### 1.2.1.2 从三维信息提取拓扑地图

近年来，由于无人机在商业、农业、救援、仓储物流等各方面的应用的兴起，对在三维环境中的无人机探索和导航的研究大量增加。许多研究工作通过对三维环境构建线上拓扑地图 (Ge 等, 2011; Oleynikova 等, 2018; Mozos 等, 2005;

Wang 等, 2020), 极大地简化了环境, 明确了三维环境的拓扑结构, 提高了无人机探索和导航的效率。

对于基于视觉进行建图的应用 (视觉 SLAM) 来说, 当机器人在探索环境时, 使用包含巨大数据量的图像集合图像直接进行建图、定位和导航是一项巨大的挑战。从图像数据中提取环境的拓扑信息, 利用视觉数据建立拓扑地图不失为一种良好的解决办法。比如 Fraundorfer 等 (2007) 通过将图像添加到数据库来维护一个图, 在机器人探索期间在线创建拓扑地图表示, 使机器人得以在包含数百万张图像的图像集中进行匹配、检测闭环, 并在拓扑地图中以良好的性能进行路径规划和导航。Mozos 等 (2005) 是一项基于视觉数据进行拓扑图构建的工作, 它是基于学习的方法, 该方法需要来自机器人的在线数据, 这意味着需要大的存储空间和在线计算能力。

Blochlinger 等 (2018) 提出了一种从视觉同步定位与映射 (VSLAM) 系统中提取 3D 拓扑图的算法 TopoMap。该方法是以体素簇 (区域) 为节点的线上拓扑地图构建方法, 是基于无人机在三维空间的应用。他们的工作取得了不错的效果, 包括在区域分割和拓扑地图表示构建上的良好性能, 其算法需要线上机器人的传感器信息和轨迹。该方法创建一组图的空闲空间簇, 作为拓扑地图的顶点。正是受其启发, 本文的三维拓扑-度量图以此为雏形创建以可通行的开阔空间 (房间、走廊等) 的三维区域为拓扑节点的拓扑地图。

对语义 SLAM 的研究近来引起了广泛的关注, 这一类算法是在建图的过程中在地图标注语义信息。在工作 Kimera 中, Kimera-Semantics 环节使用一些现有的二维语义分割工具 (比如一些基于深度学习的方法) 对地图进行语义标注, 将语义标签附加到点云中的每个三维点 (Rosinol 等, 2020)。而在今年出现的语义 SLAM 方法 Hydra (Hughes 等, 2022) 则可实时构建多层次的三维场景图 (3D scene graph), 该工作与我们的多层次的地图表示方式非常接近。其最低层次表示的节点是度量-语义三维网格, 其次层节点是识别的物体, 第三层的节点表示的是地点, 第四层节点是房间, 最高层节点是楼层。该方法与本文的多层次三维拓扑地图表示方法的区别在于, 一是, 它是一种实时的方法, 二是, 由于建图和语义标注是实时的, 它不以全局地图作为输入进行分割, 其房间的分割并非完整的, 过分割较为严重, 其更倾向于对地图做大致的区域分割。

### 1.2.2 基于拓扑地图的路径规划与导航

拓扑地图在二维和三维环境中都被大量应用于机器人的路径规划与导航,二维拓扑地图可应用于地面机器人,尤其是无人车,三维拓扑地图则应用于无人机等空中机器人。

Guo 等 (2012) 提出基于二维拓扑地图的用于自动导引车 (AGV) 的路径规划方法。该工作通过将环境简单地用直线分割形成拓扑地图,使用改进的经典 Dijkstra 算法在拓扑地图中寻找全局最优路径,再使用 A\* 算法在全局路径的每两个相邻节点之间搜索局部最小路径。为了使农业机器人在大环境中进行路径规划并确保安全路径和与植物之间的最大距离, Santos 等 (2019) 从 2D 网格图中提取拓扑图时,因农田是规则的行列排列方式,可直接从每行农田建立一串相连的点作为拓扑节点,该方法可以减少路径规划算法所需的内存总量并减少路径搜索空间。可见,即便使用最粗糙的拓扑地图构建方法和最简单的图搜索算法,从栅格地图中提取拓扑地图都能大大降低算法的搜索空间、提高搜索效率、减少计算量。

Han 等 (2020) 提出了一个基于视觉构建拓扑地图的无人机导航与定位方法。通过视觉拓扑图,无人机可以利用当时捕获的特定图像在节点之间定位和导航,拓扑图上每一个节点是一个特定的地点。该算法用于范围较大的室外环境仍可高效地导航,可见拓扑地图提升了机器人在大范围环境中应用的能力。

基于拓扑表示的全局路径规划比直接在大规模的二维网格地图上直接进行搜索规划更快,因为搜索的节点有数量级上的下降。Thrun (1998) 中的工作使用 Voronoi 图构建拓扑图,并形成了规划的路线图网络,这个方法与我们的方法比较接近。但是,它没有考虑在区域分割步骤中创建穿过房间的路径。相反,他们在一个区域中建立最短路径以连接其两个相邻区域。而我们的方法将 Voronoi 边作为机器人遵循的路径。这样,我们的方法更充分地利用了生成的 Voronoi 图,节省了避障计算的时间和内存空间。

Zivkovic 等 (2006) 中的方法基于图划分 (graph partition) 对网格地图进行分割以构建拓扑图。然后,利用马尔可夫决策过程在分割结果上进行路径规划。这是一种基于拓扑表示的动态规划方法,可以为多个目标规划路径。但与 Thrun (1998) 中的方法相比,它的计算时间和路径长度更长。

基于图的规划,也称为基于路线图的规划。路线图 (road map) (Canny, 1988)

是从二维网格图构造的一维图，通过连接兴趣点作为路径，即离线保存顶点之间的路径，然后在线查询中通过简单的搜索就可以快速从该图中提取路径。基于路线图的方法可以根据其底层方法的多样性进行区分，例如可见性图 (Latombe, 1991)、概率路线图方法 (PRM) (Den Berg 等, 2005; Kavraki 等, 1996; Holleman 等, 2000) 和快速探索随机树 (RRT) 算法 (Kuffner 等, 2002; Martin 等, 2007; Yang, 2011)。其思路是，将预先计算的路径保存在路线图中，则在线上进行路径规划的步骤中，规划算法的工作只是将起点和目标点连接到路线图中的最近点，然后在路线图中搜索路径即可。我们在本文中提出的基于拓扑地图的路径规划即是使用了预存路径的思路。使用修剪过的维诺图创建的路径可以有效避障。例如，规划算法 (Hoff 等, 2000; Rohnert, 1988) 使用 Voronoi 图来获得从障碍物到机器人自身的距离最大的路径。本文提出的路径规划方法所预存的路径也是基于修剪的维诺图形成的。

在游戏环境中应用于路线图的导航方法，导航网格法 (navigation mesh methods) (Geraerts, 2010; Van Toll 等, 2016; Hale 等, 2008) 将可步行环境划分为一组 2D 区域，以便虚拟角色可以在每个区域 (Van Toll 等, 2012) 内选择其运动。广义 Voronoi 图 (GVD) 通常用于将自由空间分解为区域以构建导航网格 (Geraerts 等, 2008)。因此，导航网格方法比基于路线图的方法提供了更大的灵活性。然而，导航网格主要是为 2D 多边形环境 (Toll 等, 2011) 设计的，要求环境的自由空间可以使用几何区域进行表示，适用于游戏场景。相比之下，真实的环境地图由形状不规则的噪声和障碍物组成。因此，导航网格法不太适合机器人扫描的真实环境的地图。

### 1.2.3 地图匹配

地图匹配问题在机器人建图中可作为数据关联问题，匹配的方法可分为线上匹配和线下匹配两种。线上的地图匹配一般用于解决机器人定位、多机器人合并建图等问题，而线下地图匹配则可用于地图合并（多机器人的地图融合）、地图纠正、地图评估等应用。

线上地图匹配的常用方法 (Howard 等, 2006) 是，在多机器人任务期间，当机器人相遇时将地图进行融合，以完成地图合并的工作，该方法需要知晓机器人的相对位姿。虽然在线地图融合算法的效果很好，但离线地图匹配算法也很有意义。当线上传感器数据不可用时，离线地图匹配算法可以仅基于网格地图进行匹

配。因为它不需要原始的传感器数据，只需要较少的存储空间和计算能力，并允许使用不同的会话中收集的多幅环境地图进行融合以重新建图。需要进行多地图融合的可能情况有：多个机器人合作收集的环境数据、或是由于环境范围过大，机器人不得不分多次收集环境数据。在这种情况下，对地图的合并就是一个关键而重要的问题。除了匹配使用单种传感器数据创建的地图，如果离线匹配算法还可匹配使用完全不同的传感器（例如立体摄像头、带 IMU 的单目视觉、深度摄像头、2D 或 3D 激光、超声波）创建的地图、或者通过完全不同的方式（例如 CAD 模型、静态扫描）创建的地图，则该地图匹配算法还可用于地图纠正的应用。

对于离线进行的地图匹配，由于地图也可以看作图片，地图匹配可以被认为是图像匹配问题的一种特殊情况。早期的工作中常将图像匹配方法用于地图匹配。SIFT (Lowe, 2004; Baker 等, 2004; Lucas 等, 1981) 等基于特征和优化的方法是用于匹配的主流。然而，由于特征太少和室内地图的自相似性，图像匹配方法在地图匹配上的表现不如在普通图像上那么好。此外，与基于拓扑地图的方法相比，它们无法处理损坏或弯曲的地图（常发生于建图时产生的传感器数据漂移）。

Carpin 等 (2004) 和 Carpin 等 (2005) 将地图匹配问题建模为优化问题，然后通过随机搜索算法求解。这些优化方法容易收敛于局部最小值并且需要初始猜测。其后，工作 (Birk 等, 2006) 通过采用自适应随机游走算法避免了局部最小值。他们通过在六维空间中搜索相对变换来解决地图成对匹配的优化问题，目标是最小化用于测量地图之间相似性的启发式函数。该算法需要大量的迭代，并在每次迭代中遍历地图的所有像素。因此，该算法的计算量是巨大的。我们的方法在对节点进行特征匹配的基础上进行基于旋转匹配的启发式搜索，从而避免了大量的遍历，而且也不要求用户给定（匹配对的）初始猜测。

Carpin (2008) 提出了一种使用霍夫变换对搜索空间进行建模并将变换分解为平移和旋转估计的方法，用于合并多机器人系统中的占用栅格地图。由于对变换估计的分解，这些方法通常是快速的、非迭代的和确定性的。然而，这些方法仅限于基于一些假设的情况，例如待匹配的地图为相同的模态，且之间仅有刚性变换。相比之下，本文的方法允许在不同模态的地图之间进行匹配。此外，本文的其中一种方法在匹配损坏或漂移的地图方面具有优势。

解决地图匹配问题的一种常见方法是将输入地图建模为抽象表示，然后根

据几何相似性或图的结构进行匹配。Huang 等 (2005) 提出了一种基于图匹配和图像对齐的算法, 将包含不同部分环境的地图通过内嵌的拓扑地图进行合并。通过类似的方法, Wallgrün 等人提出了一种使用基于 Voronoi 图 (Wallgrün, 2010) 的图匹配的地图匹配算法。Schwertfeger 等人开发了一种地图匹配方法, 用于地图质量评估, 其算法是一个基于顶点匹配的自动化过程, 该算法通过比较顶点的描述子和它们的邻居的相似性进行匹配 (Schwertfeger 等, 2016b, 2015, 2013)。Schwertfeger 等人的算法充分考虑了图结构, 但不考虑几何属性。而本论文中提出的方法充分利用了几何属性和图结构, 这允许匹配仅具有部分重叠的地图。

在三维点云匹配方面, 也有对应的基于特征描述子匹配的研究和基于拓扑表示的匹配方法。Yuan 等 (2021) 利用其提出的描述子进行点云匹配, 虽然改进了描述子使匹配效率变高, 但整个算法的计算复杂度仍达到了  $O(N^2)$  ( $N$  为点云中的点数)。Yue 等 (2020), 提出了一种用于多机器人定位的自动定位的地图匹配方法, 即利用从带有注释语义的点云中获得的语义地图进行匹配。

与本文最相似的工作是 Saeedi 等人的工作 (Saeedi 等, 2012), 他们通过区域分解, 将地图抽象为二维排列, 用于表示开放空间的边界和区域, 使用该方法能够匹配不同模式和比例的地图。随后使用拓扑结构通过随机变换找到两个地图之间的相对变换, 并通过边匹配的方法 Saeedi 等 (2012, 2014a,b) 找到地图之间的平移变换。他们的方法在不同的模态图匹配任务中实现了最先进的性能, 但是为了避免错过正确的结果, 他们遍历了搜索空间, 这使得他们的算法非常慢。为了解决这些问题, 本文的方法利用区域的几何特征筛选出可能正确的匹配, 使搜索空间更小。此外, 虽然他们将地图分割并构造成拓扑图, 但它们只是匹配区域的边界框 (bounding box), 而本文利用了区域的几何特性, 这使得算法对包含大量相似房间的环境的地图进行匹配更加鲁棒。以不同的方式匹配地图, 例如机器人地图和平面设计图, 是地图匹配领域的重要要求。一些最先进的方法, 例如 Georgiou 等 (2017) 和 Shahbandi 等 (2019), 为这项任务提出了算法, 他们的方法可以匹配不同模态的地图, 这也是本文所提出的方法的优势。这方面的例子可以在图 5.5 的 lab\_a 和 lab\_f 这两个例子中看到。

### 1.3 主要工作与文章结构

本文所介绍的内容是针对章节1.1阐述的当前地图应用所面临的问题进行解决，也是本人博士期间撰写、发表的与 Area Graph 相关的会议与期刊论文的总结与详细描述：1) 本人在会议论文Hou 等 (2019b) 中介绍了 Area Graph 的创建。2) 在 Hou 等 (2018) 中本人强调，Area Graph 非常适合快速的路径规划。3) 地图匹配的工作在 Hou 等 (2019a) 和 Hou 等 (2022) 中介绍。4) 本人与同课题组同学合作的论文 He 等 (2019, 2021) 介绍了多层次的三维拓扑-度量图的构建。这些内容在本文被分为四大章，分别对应本文的第三章到第六章。第三章与第五章的研究工作可以串联成一套创建区域图并进行路径规划、地图匹配的流程，如图1.2所示，该图亦解释了这三项工作之间的关系。第六章中的三维拓扑-度量图是对第三章介绍的 Area Graph 的进一步改进和发展。

本文全文共分为七个章节，下面将根据文章结构来介绍各章内容及本文创新点：

第一章作为绪论，主要介绍了本论文研究的相关背景，即当前机器人领域对地图的应用与所面临的问题，并综述了构建拓扑地图的研究意义。接着，阐述了拓扑地图构建、基于拓扑地图的路径规划和地图匹配的相关研究。最后，叙述了全文的主要内容及结构安排。

第二章主要是对地图和图的相关知识的介绍。拓扑地图是一种地图形式，也是一种图结构。本文提出的区域图 (Area Graph) 是基于维诺图、尤其是基于 Topology Graph 进行构建的。作者在第二章首先介绍地图的表示方式与应用的基本知识，随后介绍图和维诺图的基础概念，接着详细介绍 Topology Graph 的构建过程。这样的安排有助于读者理解区域图和多层次拓扑-度量图中涉及的概念及其构建过程。

第三章介绍了本文的第一项主要贡献：提出的一种创新的、以区域为节点的拓扑表示——区域图，主要解决了章节1.1中提到的算法在大规模、高精度的栅格地图中计算量大的问题。区域图构建算法中包含的地图分割算法则是本文的第二项贡献，作者将该算法作为一种地图分割方法与其它方法进行了对比实验以展示算法优势。这一章首先列举并介绍了区域图 Area Graph 表示和构建过程中会涉及的概念和元素符号。接着，详细描述了区域图的构建过程，其中包括区域生成、房间检测和拓扑表示创建这三个主要步骤。值得注意的是，用于生成区

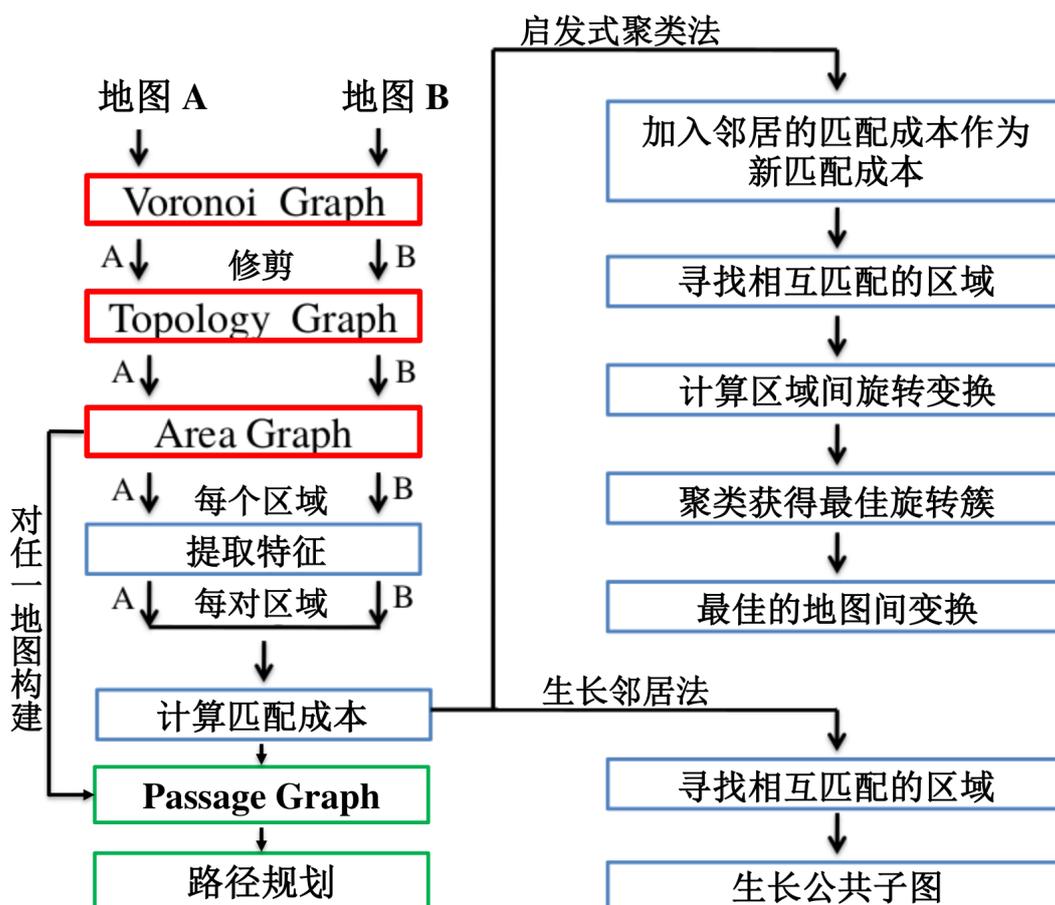


图 1.2 区域图构建（红色方框）、基于区域图的路径规划（绿色方框）与地图匹配（蓝色方框）的简要算法流程图，描述了这三项工作之间的关系。对于每幅给定地图，都会生成一个区域图表示。通过从区域图生成的通道图，可进行基于拓扑结构的路径规划。从这些区域中提取特征，并计算匹配成本进行区域匹配后，可通过启发式聚类法或生长邻居法这两种匹配方法进行地图匹配。

Figure 1.2 The pipeline of our Area Graph generation (red boxes), Area Graph-based path planning (green boxes) and map matching algorithm (blue boxes), which shows the relationship of the three works. For each map an Area Graph representation is generated. Graph-based path planning can be run on the Passage Graph generated from the Area Graph. After extracting features from the areas, computing matching cost and area matching, the map matching can be calculated either by using Hypotheses Clustering or Neighbor Growing.

域图的节点——区域的算法，亦是一种创新的地图分割方法。作者通过实验讨论区域图构建过程中会影响地图分割效果的重要参数  $\alpha$  的设置及其影响。最后，作者通过与一个基于维诺图的地图分割方法和一个最近的地图分割研究工作进行比较，并对结果进行了量化的评估。

第四章是对本文提出的区域图的一个应用示例——基于区域图的全局路径规划，作为本论文的第三项贡献，解决了基于拓扑地图的路径规划相关方法没有预存离线路径的问题。为此，作者提出了区域图的对偶拓扑图通道图 Passage Graph，Passage Graph 中存储的度量信息是离线预存的、用于穿梭区域节点的路径。这一章首先介绍通道图 Passage Graph 的原理和构建算法，接着，作者通过比较 A\* 搜索算法在通道图和栅格地图上的路径规划运算时间，展示了通道图在路径规划这一应用的计算量减少上的巨大优势。

第五章阐述了本文的第四项主要贡献，作者提出了两种基于拓扑地图的地图匹配算法——启发式聚类法与生长邻居法，方法的亮点是结合了基于特征和基于图结构的匹配方式。这一章首先简要地列出了两种地图匹配方法的算法流程，接着，具体地描述了算法的每个步骤：两个方法共有的算法步骤包括区域的特征提取与区域匹配，之后，a) 启发式聚类法通过对可能正确的匹配进行聚类的方法来获得地图间的变换关系；b) 生长邻居法通过节点间的邻居关系生长公共子图以进行地图不同部分的匹配。随后，作者通过对比不同区域特征对区域匹配正确率的影响来确定最终用于匹配算法的区域特征。最后作者为两种方法分别设计了对比实验：对于地图间有全局一致变换的情况，使用启发式聚类法与图像匹配方法和其它地图匹配方法进行地图匹配对比实验，通过计算正确率量化地评估算法；对于地图损坏导致的地图间无全局一致变换的情况，使用生长邻居法与非刚体点配准方法进行对比实验。

第六章介绍了一种创新的多层次拓扑地图表示，它是基于区域图进行进一步发展的拓扑表示。这是一种多层次的拓扑图，分别从楼层-区域-子区域-空间块提取对应的拓扑结构，高层拓扑图的节点是低层子拓扑图，低层拓扑图的节点是三维空间区域。该表示方法是从三维地图提取、构建的，允许对三维空间提取拓扑结构，包含三维的度量信息，并可以表示多楼层的建筑结构。这一章首先介绍了多层拓扑-度量图的整体结构及其涉及的相关概念，随后详细介绍了各层级拓扑图的构建算法。该拓扑图以三维空间区域为节点，因此涉及对三维地图的区域分割，则此拓扑地图构建算法包含了地图分割算法。作者在本章的实验部分通过与其它地图分割算法比较，展示了该分割算法的对区域分割的准确性。

第七章总结了全文的主要工作，并展望了下一步的工作方向。



## 第 2 章 地图与拓扑图

### 2.1 地图及其应用

地图是环境几何数据的视觉表示，它记录了环境的结构和障碍物等信息来描述环境。创建地图的过程被称为建图 (*mapping*)，其通常是指一个移动平台在环境中穿梭时使用传感器收集数据，并生成地图的过程。在这本章节中，首先介绍地图的表示方式，包括栅格地图、点云地图和拓扑地图等。随后介绍地图的应用。

#### 2.1.1 地图的表示方式

从维度上区分，地图主要分为二维和三维地图，也有特殊的 2.5 维地图（高程图）。显然，用于建图的传感器种类会影响由其创建的地图的表示维度，例如，使用单线激光测距传感器扫描生成的是二维地图，该地图仅包含三维环境中的某一高度平面的环境信息；而使用多线雷达传感器，则可获得三维地图，该地图则包含了三维环境中多个高度平面的信息。

地图可以用许多不同的方式表示。在机器人学中，最流行的地图表示是使用网格来描述环境中所有被观察区域的状态或以点云来描述被占用的空间，但也可以使用例如线 (Garulli 等, 2005) 或大表面块 (Birk 等, 2010) 等几何形状来表示环境中的障碍物。

在机器人领域中，使用范围传感器生成环境的二维表示，二维栅格地图，从很早开始就被使用 (Moravec 等, 1985; Elfes, 1989; Thrun, 2002; Durrant-Whyte 等, 2006)。这种方法通常使用超声波测距传感器或激光测距传感器 (Laser Range Finder, LRF) 通过距离测量来寻找离传感器最近的障碍物的位置。地图通常以网格表示，其中每个单元格使用概率表示该单元格是否存在障碍物（1 则表示该单元格确定被占用，0 则表示未被占用，介于 0 与 1 之间则表示不确定或未被探索的区域）。通常，这些测距传感器会在二维平面上收集大量样本来确定每个单元格所代表的环境状态。地图可以以图片的形式进行可视化呈现，即，每个像素点表示一个单元格，黑色像素点表示被占用的单元格，而白色像素点表示未被占用的单元格，未知的单元格则用灰色像素点表示。三维地图的创建使用的是不同于二维地图的传感器。最开始的三维地图构建的方法是通过立体相机完成的，这种

情况下，三维数据是从两个同步摄像机的视差信息中提取的 (Barnard 等, 1982)。除使用立体相机之外，市场上还有现成的三维激光扫描仪 (3DSOURCED, 2022)。使用激光创建的三维地图常常用点云表示，点云中的点的位置是激光碰到障碍物的位置，表示空间中的障碍物表面。栅格地图和点云表示之间的一个主要区别是，栅格地图可以区分环境中未被占用的空间与未观察到的空间，而点云只表示被占用的空间。点云不会受到网格单元的分辨率的限制。另一方面，栅格地图会自动合并同一单元格的多个观测值，而点云对同一位置观察所得的多个点并不会合并。由于网格还存储未观察到的单元格，因此它们对内存的需求可能会变得非常大，尤其是在三维地图中。通过使用四叉树或八叉树可以在一定程度上缓解这个问题。

还有的方法是使用地形分类作为单元格的值 (Triebel 等, 2006)，称为多层次表面地图 (Multi-Level Surface map, MLS map)。该方法可在网格的每个单元格中存储多个表面。这使移动机器人能够对具有桥梁、地下通道、建筑物或矿山等复杂结构的环境进行建模。

把障碍物相对于水平参考平面（地面）的高度存储在单元格中，这样表示的地图称为高程地图。高程地图也被称为 2.5D 地图，因为它们使用二维网格，但是提供有限的三维信息，而又不是完整的三维地图表示，它们只在每个单元格中存储障碍物的高度信息。高程图的一个典型应用是可以用来表示地球表面的地形 (ElevationMap.net, 2022)，因其在二维地图的基础上记录了高度信息，并且为不同的高度渐变地标记不同的颜色以进行可视化，我们可以在高程图上方便地区分高山、平原与海洋。除此之外高程地图的另一个广泛应用是可以用于地面移动机器人（尤其是足式机器人）的崎岖地形导航 (Fankhauser 等, 2018)。目前一个被广泛使用的在崎岖地形建图的开源算法是由苏黎世联邦理工学院 (ETHZ) 发布的 `elevation_map` (Fankhauser 等, 2016)。

除了上述几类基于点或网格单元的地图表示之外，还有其他地图表示不使用点或单元格，而是使用更复杂的对象，例如线、多边形或平面来表示环境。这种基于几何表示的地图通常可以更精细地抽象出环境的细节，且地图数据大小通常要小得多，使许多算法在此类表示上表现得更好。

拓扑地图则完全从传感器数据中抽象出来，保存更通用的数据。它们的节点表示地点、地标、或某一区域等，连接节点的边反映了两个地点之间的连通性。

拓扑图的常见示例是抽象的公共交通站点图（地铁线路图）。本课题提出的 Area Graph（区域图）即是一种地图的拓扑表示，Area Graph 的节点是地图中的一个区域，代表的通常是一个房间，或是没有分叉的走廊，图中连接节点的边则是相邻区域的连接之处，即门或分叉口。拓扑图更注重的不再是环境中的占用或空闲状态，也不再是环境的几何测量信息，而是环境中各个地点（或是环境结构）之间的关系，通过节点相连的边表示出来。拓扑地图通常是基于网格地图或点云之上生成的，而非由传感器数据直接生成。拓扑表示把地图的搜索空间由单元格压缩成抽象节点，因此使许多算法的搜索空间有数量级上的大幅下降。本论文即将介绍的基于拓扑地图的路径规划和地图匹配方法就是立足于拓扑图的这个优点提出的。

### 2.1.2 地图的使用

移动机器人生成的地图被用于许多不同领域的各种应用。地图的两个主要使用者是对地图中表示的信息感兴趣的人类，和需要地图或有定位信息才能执行任务的机器人。例如，人类操作员可以在开车时使用地图进行定位和导航，他还可以在地图上标注兴趣点或其他信息，或者用地图为自主机器人设置目标点。对于机器人任务，地图通常是评估当前情况、定义目标和执行任务的重要工具。例如，当涉及多机器人系统时，机器人共享他们的传感器数据以及更新对自己的定位估计，并将其它机器人扫描的地图与他们自己的传感器数据合并到一张全局地图中。执行半自主或全自主任务的机器人都非常依赖于地图，地图是许多算法必要的基础。

大多数的移动机器人应用，可以从以下几个方面使用地图：1) 地图可用于机器人对自身进行定位。当机器人被放置于环境中而不告知它所处何处，它可以通过对比自身的传感器数据与已有的地图来确定自身在环境中所处的位置。这其实可以看作是一项线上地图匹配的工作，只是一帧的传感器数据常常不足以获得可信的定位。若机器人没有已知的环境地图，而是一边对环境建图，一边在地图中定位自身，采用的技术则是同时定位和地图构建 (Simultaneous Localization and Mapping, SLAM) (Thrun, 2002; Thrun 等, 2005; Frese, 2006)。2) 地图为机器人提供环境信息以用于机器人避障或帮助机器人通过环境。当远程操作机器人，或机器人进行自主导航时，机器人可以使用避障算法根据地图中的环境信息检测到障碍物并避开。这是移动机器人应用中一个非常重要的功能。或者当使用某

些半自主功能（例如开门、爬楼梯）来通过环境时，机器人也非常依赖地图中的环境信息。3) 地图可用于路径规划，为机器人找到到达目标点的最佳路径。从起点到终点的路径常常不止一条，根据任务的实际需求，机器人可以根据地图中的环境信息选择最符合要求的一条。最常见的是要求在地图中规划出成本最低且避开障碍物的路径。也可能考虑其它因素，比如在探索任务中，机器人的目标是绘制某个未知区域的地图。这通常意味着机器人必须至少扫描到所有可以到达的未知位置一次，以便可以为环境建图。则机器人需要在节省能源的情况下尽可能地靠近未探索的区域。而在某些谨慎的应用中则相反，机器人需要尽量避免靠近未知的区域。而在远程任务或多机器人协作中，地图是实现沟通和协作的关键工具，机器人需要保持在主机或其它机器人的通信传输范围内，以进行合作探索、保持通信、保持编队等。

除了以上常见应用，地图还用于移动机器人的其他应用领域。用于军事应用，如营地安全巡逻时，则机器人基于先验地图进行巡逻时不再追求成本最低；再如机器人侦察任务，可使用 SLAM 探索未知领域。或是应用于服务机器人，则必须构建长期地图，再在日常任务中存储和更新短期障碍物的位置（例如室内环境地图中需要及时更新家具位置）。服务机器人还常常在意地图中的语义信息，比如不同功能的房间对应的是地图的哪一片区域，这是通过地图分割可以实现的功能，本文将介绍一种地图分割方法对地图进行区域分割，并建立基于区域的环境拓扑表示。应用于搜救任务时，搜救任务所涉及的环境常具有非常非结构化的地形，因此在这种环境中建图可以被认为是 SLAM 算法最困难的挑战之一，因为此环境下生成的地图通常包含很大的错误（例如，传感器数据漂移或包含噪点等错误）。若能通过地图匹配来纠正该场景下所构建的地图，将会使机器人在搜救任务中有更好的表现。本论文中将有涉及该场景的地图匹配实验，实验中使用到本论文作者提出的基于图结构的地图匹配方法。

由此可见，地图的使用对于很多移动机器人的应用来都是非常重要和关键的。而我们基于地图进行的地图分割和地图匹配也是具有实际应用意义的。

## 2.2 图与拓扑图

本章将介绍图、维诺图等几个即将用到的图结构及其基本元素，作为后续算法介绍的铺垫。本课题提出的 Area Graph 是基于 [Schwertfeger 等 \(2015\)](#) 的

Topology Graph 生成的，因此，在本章作者会花一定篇幅介绍 Topology Graph 的生成算法，以便帮助读者理解 Area Graph 的生成算法。

本课题对拓扑地图的构建与实现使用了开源的计算几何算法库 (Computational Geometry Algorithms Library, CGAL)([The CGAL Project, 2022](#))。CGAL 是一个基于 C++ 语言的开源库，它提供丰富的几何计算算法和数据结构，其中包括生成和存储 Voronoi 图 (Voronoi Diagram, 也称维诺图) 的算法包，即 CGAL 的 `Voronoi_diagram_2` 类 ([Karavelas, 2022](#))。随后本课题对拓扑地图的计算和存储使用的数据结构都是基于 CGAL 生成的维诺图的数据结构的。因此，本章会涉及许多 CGAL 的几何结构概念，接下来会详细地一一介绍。

### 2.2.1 图的基本元素

本论文所使用的名词“图”特指 graph，是图论中的概念，不等同于图像 (image) 或地图 (map)。在介绍拓扑地图之前首先对图论的一些重要元素、其存储结构、即将使用的概念进行简短的介绍。

图的基本元素和其在 CGAL 的实现中涉及的元素：

- 图 (Graph): 一般来说，图由顶点 (vertex) 的集合  $V$  和连接顶点的边 (edge) 的集合  $E$  组成。图  $G$  可以表示为  $G = (V, E)$ ，其中有  $V = \{v_1, v_2, \dots, v_n\}$ ,  $E = \{e_1, e_2, \dots, e_m\}$ ，即  $V$  是顶点  $v$  的集合， $E$  是边  $e$  的集合。如果图中的边都有规定方向，则该图是有向图，如果图中的边都是没有方向的，则该图是无向图。无向图可以看作所有边都是双向的有向图。本论文涉及的拓扑地图都是无向图，但是在数据结构的实现上是有向图，图中每条边都是由两条方向相反的有向边组成。本课题通过从二维栅格地图提取 Voronoi Diagram 来构建拓扑结构，对图的结构实现也是基于其数据结构发展的。

- 节点 (Vertex/Node): 图结构中的节点，用变量  $v$  表示。在拓扑地图中，顶点通常或多或少包含语义信息，例如作为走廊分叉口、房间、特定地点等。

- 度 (Degree): 与一个顶点  $v$  相连的边的数量称为该顶点的度，记作  $deg(v)$ 。对于有向图来说，一个顶点的度可以分为入度和出度，入度即指向该顶点的边的数量，相应地，出度即以该顶点为起点的边的数量。一个顶点的度，是其出度与入度之和。

- 边 (Edge): 图中两个顶点 (节点) 之间的连接，即  $e_{ij} = (v_i, v_j)$ 。边通常表示一种连通关系，即两个节点代表的位置或地点之间是可互相到达的。

- 半边 (*Half Edge*): 在一个双连通的边列表 (Doubly Connected Edge List, DCEL) 的数据结构 (Muller 等, 1978) 中, 半边是两个顶点之间的有向边, 用  $h$  表示。图中的每条 edge (无向边) 由两条相反方向的半边组成。在本论文中, 所有边 (edge) 都可表示为:

$$e_{ij} = (v_i, v_j) = \{h_{ij}, h_{ji}\}, \quad (2.1)$$

其中, 半边  $h_{ij}$  是从顶点  $v_i$  到顶点  $v_j$  的有向边。然而, 为简单起见, 在接下来的整篇论文中, 我们提到“边”即指两个半边结合, 可看做无向边。

- 孪生关系 (*Twin*): 所有半边都有一个与其是孪生关系的半边, 该孪生半边与其连接的顶点相同、方向相反。即, 一条半边的起始节点是其半边孪生的终点, 而其指向的终点是其孪生半边的起点。

- 射线 (*Ray*): 射线是一种特殊的边, 它仅连接一个节点, 其另一端指向无穷。除此之外, 因其仅有一个方向, 射线仅包含一条半边。由 CGAL 生成的原生维诺图中是不包含射线的, 当我们对维诺图进行修剪操作时, 由于一些边被切割、其一端的顶点被移除, 这些边才变成了射线。不过, 由于我们最终的拓扑地图中不需要射线, 这些射线最后也会被移除。

• 面 (*Face*): 在几何中, face 是一个多边形区域。在基于双连通边列表的数据结构实现中, 一个面是由多条半边绕顺时针方向围成的区域, 半边关联的面是在其右侧、由该半边所参与围成的多边形区域。半边的孪生半边以另一侧的区域作为它关联的面。在 Voronoi 图中, 每个面 (Voronoi 单元) 恰好包含一个点 (site)。在这个本课题的应用中, 这个点对应于网格地图中的障碍物单元。

• 子图: 图的子集称作该图的子图, 子图的顶点集是该图顶点集的子集, 且子图的边集是该图的边集的子集。

### 2.2.2 维诺图 (Voronoi Diagram)

广义维诺图 (Generalized Voronoi Diagram, GVD), 也称为维诺分解 (Voronoi decomposition) 或狄利克雷镶嵌 (Dirichlet tessellation) (Aurenhammer 等, 2000, 2013), 是一种分割空间的几何结构, 广泛用于许多应用领域 (Pehlivanoglu, 2012; Lebedeva 等, 2018; Wan 等, 2019)。在二维栅格地图的可视化表示 (图片) 中, 障碍物被表示为黑色的像素点, 以这些点为基点 (site), 我们可以生成维诺图 (Voronoi Diagram, 后续将简称为 VD, 或 Voronoi 图)。维诺图由顶点 (维诺点) 和边 (维诺

边) 组成, 将空间划分为包围输入点的单元 (cell)。每个由维诺边围成的单元内, 恰好包含一个基点, 而维诺边的交点则是维诺图的顶点。对于一条维诺边所涉及的两个相邻区域内包含的两个基点, 该维诺边与这两个基点之间的欧式距离相等, 维诺点与其涉及的三个区域内的基点的距离相等。即, 与基点关联的 Voronoi 单元内的任意一点与该基点的距离不大于与所有其他基点的距离。图2.1展示了一个示例维诺图的部分放大视图, 其中黑色的点代表原图片中的一个像素点。基于这些黑色的点生成的维诺图用红色的点和线段分别表示其顶点和边。在本课题的使用中, 地图中黑色的像素点代表的是障碍物所占用的位置, 本文将取 2D 网格地图中所有的占用网格 (黑色点) 作为维诺图的基点来生成维诺图。本课题的维诺图生成是使用计算几何算法库 CGAL, 把给定的二维栅格地图以图片的方式作为输入生成的。

在 Reem (2011) 中, Reem 探讨了维诺图生成的稳定性。Reem 发现, 当输入的基点位置发生微小的变化, 维诺图的单元也仅产生微小的变化。这对于地图应用是十分具有意义的, 因为机器人扫描得到的地图总是略有不同的。由栅格地图直接生成的维诺图不可直接用于地图应用, 比如生成拓扑地图以进行路径规划, 因为基点之间通常彼此非常接近, 生成的原生维诺图的边因此会穿过障碍物 (比如墙体作为不可穿越的障碍物由许多障碍点组成, 生成的维诺边为了分割这些点则会穿越墙体), 而这些边是绝不可用于路径规划的。在本文后续的算法中, 将利用简化过的维诺边和维诺单元生成我们提出的拓扑地图结构。

为了生成和存储维诺图, 我们使用 CGAL 的 *Voronoi\_diagram\_2* 类 (Karavelas, 2022)。为了使用该算法包, 我们将网格地图中所有“占用”的单元格以点的形式放入向量中, 形成点集, 使这个 2D 点集作为是 CGAL 的 *Voronoi\_diagram\_2* 类的输入。在计算并生成维诺图之后, *Voronoi\_diagram\_2* 类提供对 Voronoi 的顶点、边以及由这些边作为边界的单元 (在 CGAL 中 cell 单元变量被存储为 face) 和单元内的基点 site 的访问。由 CGAL 从 2D 地图创建的维诺图  $VD$  及其数据结构是我们接下来要生成的拓扑地图的基础, 以数学形式表达为:

$$VD = (V^{VD}, E^{VD}, F^{VD}), \quad (2.2)$$

其中, 维诺图中的每条边  $e$  都由两条半边  $h$  (halfedge) 组成, 其下标为两端顶点  $v$  的下标:

$$e_{ij}^{VD} = (v_i^{VD}, v_j^{VD}) = \{h_{ij}^{VD}, h_{ji}^{VD}\} \in E_{VD}. \quad (2.3)$$

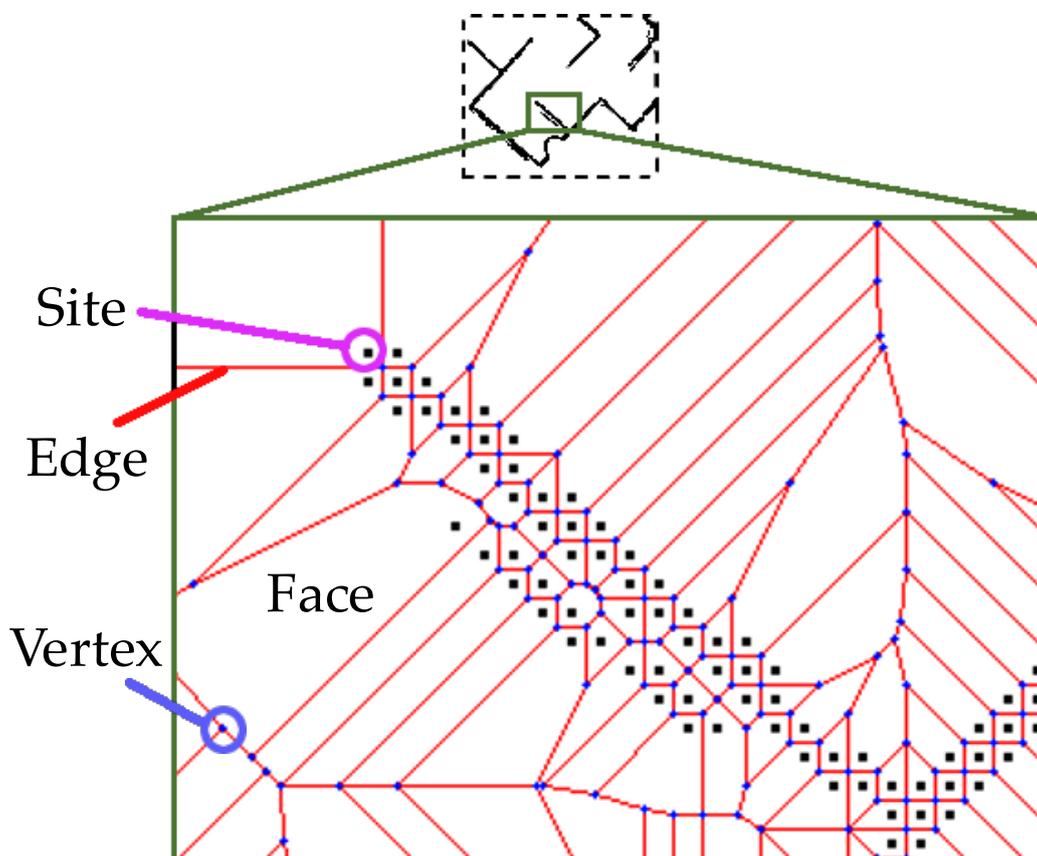


图 2.1 一个由二维栅格地图生成的维诺图的一部分区域的放大细节。这个维诺图是从网格地图（被占用的单元格）生成的，并选取其一小块区域（绿色的小方块）进行放大。图中展示了维诺图的顶点、维诺图的边和维诺图的区域 (face) 的示例。在邻居单元之间也会创建维诺图的边，此情况下生成的维诺边会穿过墙壁。

Figure 2.1 The details of a part of a Voronoi Diagram, which is generated from a 2D grid map. The Voronoi Diagram is generated from the sites of the grid map (occupied cells), a patch of which is zoomed in from the small green square. It points out examples of a Voronoi Diagram vertex, a Voronoi Diagram edge and a Voronoi Diagram face. Voronoi edges are created even between neighboring occupied cells, which means the edges go through the walls in this situation.

在 CGAL 生成的维诺图中，每条半边都关联一个 face。face 由几条 halfedge 围成，因此每个 face 即一个小区域，维诺图的所有 face 都保存在集合  $F_{VD}$  中。每个 face 都与一个 site（网格图中的占用单元格（输入点））相关联。这些 face 将在后续被用于创建多边形作为拓扑地图的区域。

### 2.2.3 Topology Graph

Topology Graph 是由 Schwertfeger 等 (2015) 提出的一种二维拓扑地图。其顶点表示二维空间中的位置，边表示这些位置之间的（可互相到达）路径。图 1.1a 展示了一个 Topology Graph 的示例，可见 Topology Graph 中的边描述了二维地图的骨架。本课题提出的 Area Graph 是在 Topology Graph 的结构基础上经过一系列操作生成的。这一小节将介绍 Topology Graph 的生成算法，并对其图结构通过变量进行描述。

首先，我们介绍在 Schwertfeger 等 (2015) 中为 Topology Graph 定义的元素，图 2.2 是一个在修剪过程中的维诺图，其中展示的这些元素，可以通过图文对照来看以帮助理解：

- 尽头顶点 (*Dead End Vertex*): Topology Graph 中度为 1 的顶点，即只有一条边与之相连。这类顶点常出现在走廊的尽头。
- 尽头边 (*Dead End Edge*): 与尽头顶点相连的边。
- *Spurious Edge*: 与尽头顶点相连的边，并且此边的长度小于某一阈值。这些边在后续操作中会被修剪，以精简 Topology Graph。
- 主边 (*Major Edge*): 不是 *Spurious Edge* 的边则成为主边。
- 交叉点 (*Junction*): 与至少三条主边相连的顶点。
- 主节点 (*Major Vertex*): 度为 1 的顶点并且与其相连的边为主边。或者与其相连的主边数量大于 2 的顶点。即，作为 junction 或 dead end vertex 的节点，这类顶点最终会被保留作为 Topology Graph 的节点。
- *Spurious Vertex*: 与该顶点相连的至少一条边为 *Spurious Edge*，将在 *Spurious Edge* 被修剪后该顶点恰好与两条主边相连，这类节点会被跳过，无法保留成为 Topology Graph 的节点。

由 2D 网格图生成的原生 VD 中，每个 site 都会被维诺边包围，形成一个维诺单元。因此，对于非常靠近的两个 site，维诺边仍会从两点之间穿过（参见图 2.1），则造成例如墙内有边穿过的现象。因此，我们要从图中删除与被占用单元

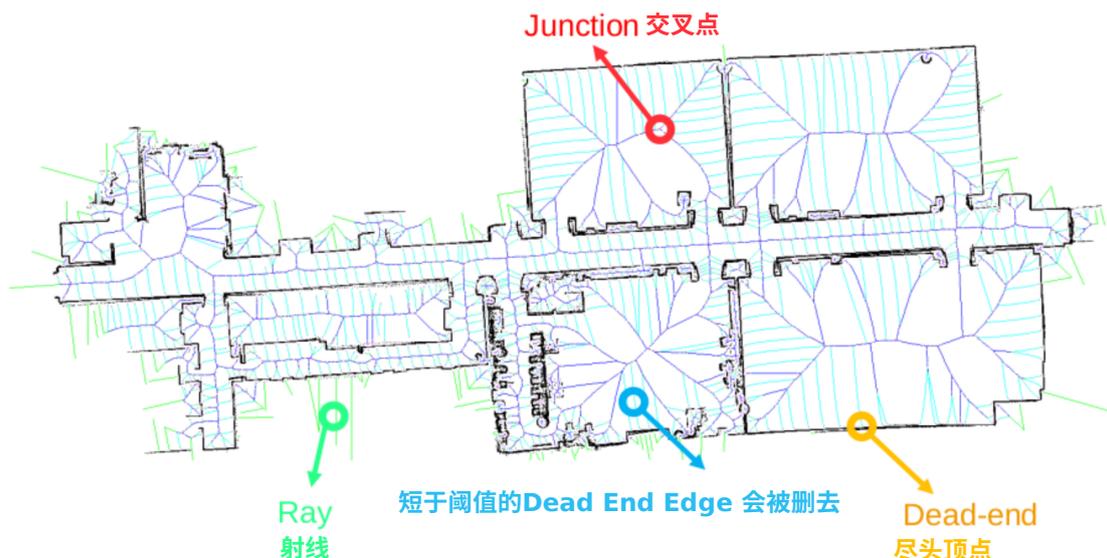


图 2.2 只修剪过一次的维诺图。为了最终得到一个简洁的 Topology Graph，上图中还有很多尽边（浅蓝色）需要被删除。射线（浅绿色）是指向无穷远的，因此无法从中创建出多边形，它们也将被移除。

Figure 2.2 A Voronoi Diagram that is only pruned once. There are still a lot of dead-end edges (light blue) to be deleted. Rays (light green) are unbounded, so no polygon can be created from it, they will be removed, too.

的距离小于阈值的所有边，即对 Voronoi 图进行剪枝。对维诺图经过一系列的剪枝和整理，最终形成的 Topology Graph  $G^T = (V^T, E^T)$ ，其边清晰而不杂乱地描述了地图的大致骨架。这一系列剪枝过程较为复杂，为了方便读者理解，这里将由二维栅格地图生成 Topology Graph 的一系列步骤（函数）列于表 2.1 中。

接下来，本文作者将依照步骤首次执行的先后顺序详细介绍各个步骤。但是在此之前，我们首先介绍有三个步骤中都要涉及到的 Alpha Shape。

### 2.2.3.1 Alpha Shape ( $\alpha$ -shape)

Alape shape ( $\alpha$ -shape) 是一个多边形，从输入的点集中检测  $\alpha$ -shape 常常意味着要检测点集中没有点聚集的空间，如果输入点集是地图的占用单元则检测  $\alpha$ -shape 直观上即是寻找环境中的开阔空间（没有障碍物的空间）。图 2.3 显示了输入一个点集从中获得 alpha shape 的原理：这些点相当于是 2D 网格图的占用单元。使用用户设定的值  $\alpha = r^2$ ，任意取该集合中的两个点，可以定义了两个半径为  $r$  的圆，仅当这两个圆中的至少一个不包含集合中的任何其他点时，在这两

步骤 (函数名)	功能描述
噪点去除 (REMOVE_OUTLIERS)	去除二维栅格地图中的噪点。
杂物去除 (ALPHASHAPEREMOVAL)	删去二维地图中的家具。
创建 Voronoi 图 (CREATEVORIGRAPH)	由栅格地图生成维诺图。
移除尽边 (REMOVEDEADENDS)	删去长度短于阈值的尽边。
Edge Skipping (EDGESKIPPING)	跳过 Spurious Vertex 生成 Topology 边。
Remove Outside (REMOVEOUTSIDE)	删除维诺图在地图边界外的部分。
保留最大子图 (KEEPLARGESTGROUP)	仅保留边的长度总和最大的连通子图。
去除射线 (REMOVERAYS)	删去图中指向无穷远的边 (射线)。

表 2.1 用于 Topology Graph 生成的函数及其描述。一些函数在算法 1 中有出现。

Table 2.1 Functions for Topology Graph generation and their descriptions. Some functions also appear in Algorithm 1.

个点之间连一条线段作为即将创建的 alpha shape 的边缘。测试输入点集中的所有点对，将创建的边连接在一起形成的多边形即是 alpha shape。由此可见，形成 alpha shape 的区域内是没有点占用的，因此，alpha shape 的检测可以用于检测点集内部的开阔空间、点集的外边界、以及开阔空间中的小簇点（在地图中通常是杂物）。而且，参数  $\alpha$  设得越大，也即圆盘的半径越大，使用 alpha shape 检测出来的开阔空间的最小面积越大。

在本文的实现中，我们使用 CGAL 的 *Alpha\_shape\_2* 类 (Da, 2022) 来进行 alpha shape 的检测，在本算法中 alpha shape 检测算法被用于杂物去除、地图边界检测、房间检测等三种功能。

### 2.2.3.2 去除地图中的噪点与杂物

过度分割 (Over-Segmentation) 是地图分割的一个常见问题，通常由家具和地图噪点引起。在 He 等 (2019) 中，我们展示了使用 Chen 等 (2020) 中介绍的机器人创建无家具地图的工作。该方法要求使用两个垂直放置的激光雷达进行扫描，通过在三维地图中寻找墙壁和天花板来去除对环境中杂物的扫描，形成无杂物的、仅包含环境结构的地图。本课题涉及的算法主要是用于离线处理已经获得的二维栅格地图，地图中通常已经包含有家具等杂物。对于这种情况，则需要对输入的地图进行一些预处理，包括去除地图扫描过程中产生的噪点，还有地图中的

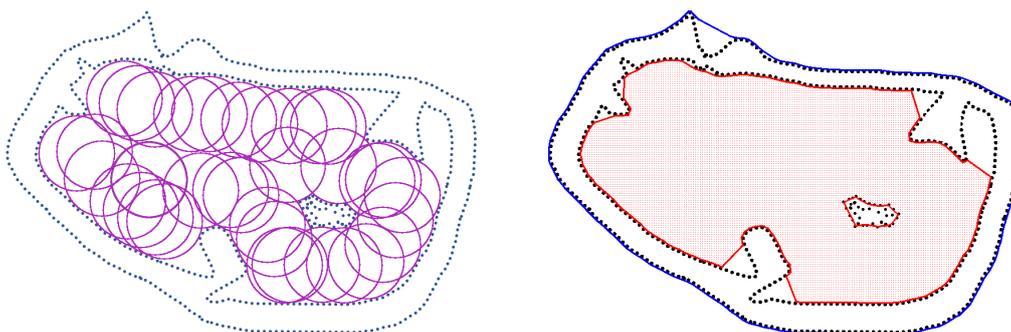


图 2.3 Alpha Shape 通常用于检测点之间的开放空间。 $\alpha$ -shape 的大小取决于参数  $\alpha$ ，它是圆盘的平方半径，左图显示了点集内部的圆盘（省略了外部圆盘以节省空间）。Alpha shape 的创建是通过将圆盘接触到的点用线段连接。右图为输出的  $\alpha$ -shape，外部  $\alpha$ -shape 显示为蓝色，内部  $\alpha$ -shape 显示为红色。

Figure 2.3 Alpha shape is usually used to detect open space between points. The alpha shape size depends on the parameter  $\alpha$ , which is the squared radius of disks (outside ones omitted to save space). An alpha shape is created by connecting the disk-reached points with line segments. The output on the right shows the outer  $\alpha$ -shape in blue and the inner  $\alpha$ -shapes as red.

家具等杂物。

为了减少地图噪点的影响，我们使用 CGAL 的 `REMOVE_OUTLIERS` 函数 (Alliez 等, 2022) 从输入图像中初步去除地图中散落的小噪点，该函数出现在算法 1 中。它检测噪点的原理是计算每个点与其最近邻居的平方距离的平均值，删除计算结果最大的点。用户可指定最近邻居的数量或指定邻域球体的半径作为近邻阈值，以及指定要删除的点占总点数的百分比。当地图中存在一簇点，其包含的点数量极少，则很有可能是噪点。噪点簇中的各个点由于紧挨着的近邻较少，这些点与其近邻的平方距离值算出来就会比较大。图 2.4b 展示了去除噪点后的地图，较原生地图更干净一些。

随后，我们检测地图中的杂物（例如家具等）并进行去除，此处我们使用  $\alpha$ -shapes Edelsbrunner 等 (1983) 算法来做检测，其用于生成 alpha shape 的参数记为  $\alpha_{robot}$ ，这个参数由机器人的宽度决定：使用比机器人宽度的一半略宽的尺寸作为 alpha 圆盘的半径，即机器人可通过的空间都被包括在 alpha shape 内，被包裹在此参数下检测出的 alpha shape 内的小点簇（点数小于某阈值的点簇）则被认为是可以去除的杂物，此功能在函数 `ALPHASHAPEREMOVAL` 中实现，该函数也

出现在算法 1 中。图 2.4c 展示了使用 alpha shape 去除杂物后的地图。

### 2.2.3.3 使用 Alpha Shape 检测地图边界

由于原始的维诺图延伸到地图之外的无穷远处,因此需要找到地图的边界以删除不需要的数据,即在边界之外的维诺图。为此,再次使用 CGAL 的 alpha shape 检测算法 (Da, 2022) 来帮助我们检测边界 (实现于函数 `PERFORMALPHASHAPE`)。如同作者在算法1中描述的,使用 alpha shape 算法对地图的边界检测和开阔空间 (房间) 检测是在同一函数中进行的。此次检测 alpha shape 的参数  $\alpha_m$  设置在后续的章节将会讨论,因为该参数要兼顾检测房间的功能,直觉上来说会设置该参数使 alpha 圆盘直径小于要检测的区域的宽度,同时大于区域连接通道 (通常是门) 的宽度,这样可以确保要检测的房间内可容纳下 alpha 圆盘,即可检测到 alpha shape,同时又不会使不同的区域互相连通,即当圆盘可以通过区域的连通处,两个区域会被检测为一个区域。在此设置下,除了地图内部的开阔区域可以检测到 alpha shape 之外,地图的外部 (延伸向无穷远) 也会检测到一个巨大的 alpha shape,则此 alpha shape 可以作为地图的边界。图 2.4d 展示了使用 alpha shape 检测到的地图边界。

在检测到地图边界之后,需要删除掉地图边界之外的 Voronoi 顶点和边。首先删除边界外所有 Voronoi 点,随后删除所有从这些点出发的半边 (有向边),则得到如下结果:如果一条边 (如上所述,一条无向边由一对双向的孪生半边组成) 两端连接的都是边界外的 VD 顶点,则完全删除这条边,即删除这对双向的孪生半边;若边的一端连接边界内的 VD 点,另一端连接边界外的 VD 点,则删除从边界外的 VD 点出发的半边,保留从边界内 VD 出发的半边,此时,形成了一种特殊的边结构 射线 (ray) (Brönnimann 等, 2022)。CGAL 中的射线结构是从一个点出发,指向无穷的有向边。通过上述边删除操作我们也可以看出,射线结构仅包含一条半边,因此射线是有向边,且仅连接一个顶点。在后续的操作中,这些射线也会被删除,但此时它们因算法只删除从被删顶点出发的半边而暂时存在。

### 2.2.3.4 删除短于阈值的 Dead End Edge

随后,将使用函数 `REMOVEDEADENDS` 删除与地图中的点距离小于设定阈值长度以下的 dead end edge (尽边),此过程仅删除尽边及与尽边相连的尽头顶点,并不删除与该尽边相连的非尽头顶点。该尽边删除过程会被重复执行多次 (一般

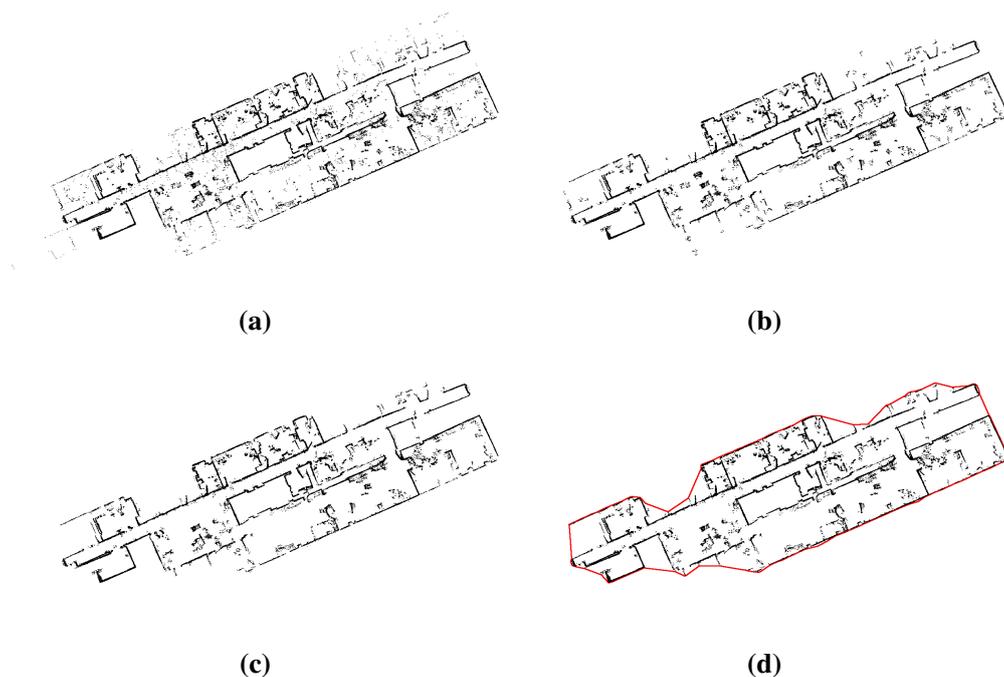


图 2.4 2D 栅格地图的预处理。(a) 未经去噪处理的原生二维栅格地图（已将未知区域（背景）设置为白色）。(b) 经过 CGAL 的 `REMOVE_OUTLIERS` 函数进行噪点去除的地图。(c) 经过去家具杂物的地图。(d) 使用 `alpha shape` 检测算法检测地图边界（显示为红色）。

**Figure 2.4 Preprocessing for a 2D map. (a) Map before noise removal (the unknown region (background) is set to white). (b) Map after noise removal with the CGAL `REMOVE_OUTLIERS` function. (c) Map after furniture removal. (d) Map boundary (shown in red) detected by alpha shape detection.**

是 2-3 次)。每次删除尽边会设定不同的长度阈值：第一次会设定一个非常大的长度阈值，因为算法希望删除所有指向地图中障碍点的边，它们通常都不是地图的骨架，对导航等地图应用意义不大。图 2.2 显示了经过一次修剪的 Voronoi 图。而第二、三次删除尽边的目的则是尽量删除骨架上的短毛刺、以及与地图中的点太靠近的维诺顶点，使生成的 Topology Graph 尽量简洁。对于每轮删除 dead end edge，都会伴随一次 *Edge Skipping* 过程的执行，其具体步骤在下一子章进行介绍。

### 2.2.3.5 跳过非 Junction 的顶点

在对图进行剪枝的过程中，一些尽头顶点被移除，其边也被一起删去，则导致图中一些本来是交叉点 (junction) 的顶点失去了与它连接的一条边（尽边）从而不再是一个交叉点，即，顶点的度由 3 变为 2 或更少，成为 Spurious Vertex。

Topology Graph 通过 *Edge Skipping* 过程仅保留 junction 和 dead end 两种顶点，其原理是：只连接两条边的所有顶点 (Spurious Vertex) 都被跳过，与 Spurious Vertex 相连的两条边被合并为一条。跳过一个顶点是指，该顶点在 Topology Graph 中不再作为一个节点，而仅仅是 Topology 边的一部分。即，该过程使用被跳过的 VD 顶点作为中间点 (waypoint) 将与这些 VD 顶点相连的 VD 边连接在一起生成 Topology Graph 中的边。下文中，Topology Graph 中的边将被称为 Topology 边，记作  $e^T$ 。Topology 边  $e_{ij}^T = (v_i^T, v_j^T) \in E^T$  两端连接的节点  $v^T$ ，要么是交叉点 ( $\text{deg}(v^T) > 2$ )，要么是尽头顶点 ( $\text{deg}(v^T) = 1$ )。

Topology Graph 的 edge skipping 过程可以被描述为

$$\begin{aligned} e_{ij}^T &= \{e_{ik_1}^{VD}, e_{k_1k_2}^{VD}, \dots, e_{k_mj}^{VD}\} \\ &= (v_i^{VD}, v_{k_1}^{VD}, v_{k_2}^{VD}, \dots, v_{k_m}^{VD}, v_j^{VD}) \\ &= \{h_{ij}^T, h_{ji}^T\}, \end{aligned}$$

其中， $\{v_{k_1}^{VD}, v_{k_2}^{VD}, \dots, v_{k_m}^{VD}\}$  是来自维诺图的顶点，它们不再是 Topology 边的端点，仅是其途经的点。

### 2.2.3.6 删除射线并保留最大子图

删除地图边界外的维诺图之后，图中会包含射线，即只有一个顶点并指向无穷的边，这些边是在删除穿过地图边界的顶点连接边时形成的。这一步将删除所有图中的射线，但是保留连接这些射线的维诺图顶点（在函数 REMOVE\_RAYS 中实现）。

由于只有可到达区域的图在地图应用中才是有意义的，因此最终只保留长度和最大的连通图作为 Topology Graph，并删除所有不属于最大连通图的维诺顶点和边（这一步实现于函数 KEEP\_BIGGEST\_GROUP）。

生成 Topology Graph 的函数总结在表 2.1 中。图 1.1 的第一幅图展示了由 2D 栅格地图生成的最终 Topology Graph，其尽边为浅蓝色，其他边为深蓝色。其顶点仅有 junction 顶点及尽头顶点两种。

## 2.3 本章小结

本章主要分为两部分。第一个子章介绍地图的种类和应用。从信息量来分，地图可以分为二维地图、三维地图，和处于两者中间的 2.5 维地图（高程地图）；

从表示方式来分，地图的表示形式可分为栅格地图、点云、树状图、几何表示等。第二个子章首先介绍了图的基本概念和元素，随后介绍了维诺图和 Topology Graph 的结构和创建过程。这些作为背景知识，有助于读者了解拓扑图的基本结构，进而更容易理解下一章提出的区域图的结构和构建过程。

## 第3章 Area Graph: 从二维栅格地图提取的拓扑表示

### 3.1 基本概念及元素介绍

绪论中的章节1.1.1曾简要介绍到，区域图 (Area Graph) 的生成算法的输入是一个二维占用网格图，算法的输出是一个拓扑-度量图，即区域图，我们将它记为一个无向图  $G^A = (V^A, E^A)$ 。

区域图使用的地图分割是通过在 Topology Graph 的基础上添加和合并多边形发展而来的，可以看作是从 Topology Graph 添加了多边形的变体构成的。而 Topology Graph 基于修剪后的 Voronoi Diagram (VD)。本章节从描述维诺图的结构和符号开始。随后，将介绍 Topology Graph (Schwertfeger 等 (2015))  $G^T = (V^T, E^T)$  的符号以及如何在 Topology Graph 每个边上附加多边形。带多边形的 Topology Graph 变体称为临时图或者过渡图 (Temporary Graph 或 Middle Graph)  $G^M = (V^M, E^M, P^M)$ ，其顶点集和边集在最开始时与  $G^T$  的相同，随后在区域合并阶段更新。接着，通过房间检测算法合并一些多边形，得到一个分割良好的地图，这些多边形定义的区域被称为 *areas*，它们将成为区域图中的顶点。最后，在此基础上，将介绍定义区域图的符号。

#### 3.1.1 Voronoi Diagram

CGAL 的维诺图算法将 2D 网格地图中的所有占用单元作为维诺图的基点 site (在章节2.2.2中我们曾介绍 site 与维诺图的顶点与边之间的关系) 来生成维诺图 (Voronoi Diagram)  $VD$ 。图 2.1 显示了维诺图的某个部分的放大细节图。如前所述，维诺图可以表示为：

$$VD = (V^{VD}, E^{VD}, F^{VD}), \quad (3.1)$$

其中，

$$e_{ij}^{VD} = (v_i^{VD}, v_j^{VD}) = \{h_{ij}^{VD}, h_{ji}^{VD}\} \in E_{VD}. \quad (3.2)$$

由于统一使用了 CGAL 进行图结构的算法实现，本文使用的所有图的所有边 (edge) 都由一对相反方向的孪生半边 (half edge) 组成。因此，对所有图 (包括  $VD, G^M, G^A$ ) 的所有边都具有以下结构：

$$e_{ij} = (v_i, v_j) = \{h_{ij}, h_{ji}\}, \quad (3.3)$$

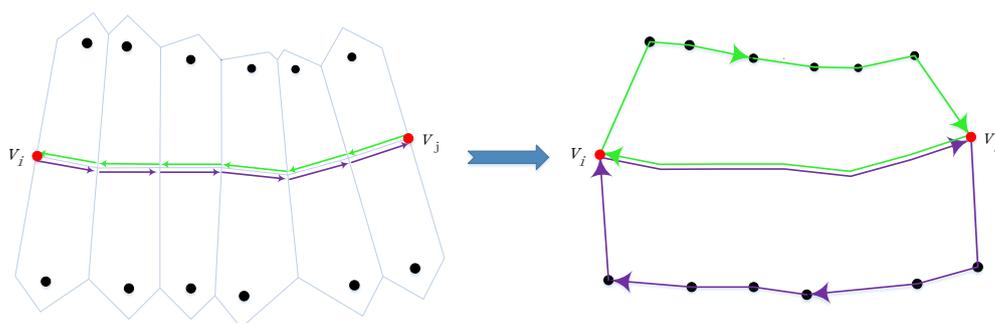


图 3.1 左示意图：绿色和紫色短箭头是相反方向的 Voronoi 半边。每个 Voronoi 半边都关联一个 face，其中仅包含一个 site。右示意图：按顺时针顺序将半边与 site 连接以创建半多边形。一对半多边形组成一个多边形，附加在其对应的 Topology Graph 边上。半多边形生成步骤（合并 face 的步骤）是在创建 Topology Graph 边  $e^T$  的 EDGE\_SKIPPING 步骤的同时完成的。创建的多边形在 CREATEPOLY 步骤中完成。

Figure 3.1 Left schematic: The green and purple short arrows are the Voronoi halfedges in opposite directions. Each Voronoi halfedge has a face, which contains exactly one site inside. Right schematic: Connect halfedges with sites in clockwise order to create half-polygons. A pair of half-polygons make up a polygon, attaching on its corresponding Topology Graph edge. This is done during the EDGE\_SKIPPING step that creates the Topology edge  $e^T$ . The polygon created is done in the CREATEPOLY step.

其中，半边  $h_{ij}$  是 CGAL 的数据结构概念，是从顶点  $v_i$  到顶点  $v_j$  的有向边。然而，为简单起见，在整篇论文中，术语“边”是指两条半边的结合，作为一条无向边。

在维诺图生成时，每个半边都有一个 face。VD 的所有 face 都保存在 face 集  $F_{VD}$  中。每个 face 都与一个 site（网格地图中的占用单元（点））相关联。图 3.1 左示意图显示了一组 VD face，可以合并成半多边形。一个 face 由其周围的 VD 半边围成，内部包含一个 site。

### 3.1.2 Topology Graph

从 2D 网格图生成的原生 VD 有很多边，有在两个靠近的障碍点之间穿过的边（参见图 2.1）将会导致墙内有图结构的边穿过。因此，从网格图生成维诺图后，算法将对维诺图进行剪枝，生成 Topology Graph  $G^T = (V^T, E^T)$ 。剪枝过程称为 *Dead End Removal*，并伴随 *edge skipping*。Dead End Removal 从图中删除与被占用单元的距离小于阈值的所有边。在那之后，一些 VD 顶点失去了它的一条尽

边并且不再是一个交叉点。交叉点 (junction) 即一个具有两个以上连接边的顶点。edge skipping 过程通过将非交叉点之间的边首尾连接成为一个 Topology Graph 的边。

经过 edge skipping 步骤的处理, Topology Graph 的边  $e_{ij}^T = (v_i^T, v_j^T) \in E^T$  由一系列维诺边组成, 而维诺顶点成为 Topology Graph 边附加的度量路径的途经点。Topology Graph 可以描述为

$$G^T = (V^T, E^T), \quad (3.4)$$

其顶点可表示为

$$V^T = \{v \in V^{VD} | \deg(v) = 1 \vee \deg(v) > 2\}, \quad (3.5)$$

其中, 顶点的度数  $\deg(v)$  表示连接到顶点的边数。而 Topology Graph 边的构建可表示为

$$\begin{aligned} e_{ij}^T &\in E^T \\ &= (v_i^{VD}, v_{k_1}^{VD}, v_{k_2}^{VD}, \dots, v_{k_m}^{VD}, v_j^{VD}) \\ &= \{e_{ik_1}^{VD}, e_{k_1k_2}^{VD}, \dots, e_{k_mj}^{VD}\} \\ &= \{h_{ij}^T, h_{ji}^T\} \\ &= (v_i^T, v_j^T), \end{aligned} \quad (3.6)$$

其中  $\{v_{k_1}^{VD}, v_{k_2}^{VD}, \dots, v_{k_m}^{VD}\}$  是来自原生维诺图的顶点, 作为 Topology Graph 边的途经点。它们不是边  $e_{ij}^T$  的端点。

### 3.1.3 临时图

区域图的区域是由附加到 Topology Graph 边上的多边形生成的。为此, 定义附加了多边形的 Topology Graph 变体为临时图  $G^M = (V^M, E^M, P^M)$ , 用以解释为了构建 Area Graph 的区域而添加和合并多边形的处理过程。

在 CUTEDGES 步骤之前 (如算法 1 所示), 用 Topology Graph 初始化临时图的顶点和边, 即:

$$V^M = V^T, E^M = E^T, \quad (3.7)$$

随后, 经过添加 face 为多边形的操作, 临时图的每条边  $e^M$  恰好有一个多边形  $p^M \in P^M$ 。也就是说, 图  $G^M$  是带有附加多边形信息的 Topology Graph 变体。

创建多边形的算法（通过连接 Voronoi face）是基于 edge skipping 算法发展而来的。对于每个半边  $h_{ij}^M$ ，按顺时针顺序连接 face 中的 site 和对偶 site 来构建半多边形  $hpoly(h_{ij}^M)$ 。该过程如图 3.1 所示。一对孪生半多边形  $\{hpoly(h_{ij}^M), hpoly(h_{ji}^M)\}$  被看做是一个多边形，关联一条 Topology Graph 边  $e_{ij}^M$ ，表示为

$$\begin{aligned} p_{ij}^M &= \{hpoly(h_{ij}^M), hpoly(h_{ji}^M)\} \\ &= poly(e_{ij}^M) \in P^M. \end{aligned} \quad (3.8)$$

因此，临时图  $G^M$  的结构可以表示为

$$G^M = (V^M, E^M, P^M), \quad (3.9)$$

其中

$$\begin{aligned} V^M &= \{v \in V^{VD} \mid deg(v) = 1 \vee deg(v) > 2\}, \\ E^M &= \{e_{ij} = e_{ji} = (v_i, v_j) = (h_{ij}, h_{ji}) \mid v_i, v_j \in V^M\}, \\ P^M &= \{p_{ij} = poly(e_{ij}) = (hpoly(h_{ij}), hpoly(h_{ji}))\}. \end{aligned} \quad (3.10)$$

#### 3.1.4 Area Graph

在合并 face 的步骤之后，临时图  $G^M$  中的最终多边形被定义为区域 (area)，以区域为节点来生成区域图  $G^A$ ，表示为

$$G^A = (V^A, E^A), \quad (3.11)$$

其中区域图的每个顶点  $v^A \in V^A$  是最终临时图中的一个区域，其记录的度量信息即代表该区域的多边形，及该区域的几何特征；而区域图中连接两个顶点的边  $e^A \in E^A$  记录区域之间的连接，其记录的度量信息包括连接点和在两个区域之间的共同边界。

**算法 1** 区域图的生成**算法 1** Generate the Area Graph**Require:** 2D Grid Map  $map$ ; parameters  $noise\_percent$ ,  $\alpha_{robot}$ ,  $\alpha_m$ **Ensure:** Area Graph  $G^A = (V^A, E^A)$ 


---

```

1:  $map \leftarrow REMOVE\_OUTLIERS(map, noise\_percent)$ 
2:  $map \leftarrow ALPHASHAPEREMOVAL(map, \alpha_{robot})$ 
3:  $VD \leftarrow CREATEVORIGRAPH(map)$ 
4:  $G^M \leftarrow CREATEPOLY(VD)$ 
5:  $AS_{biggest}, alphaShapes \leftarrow PERFORMALPHASHAPE(map, \alpha_m)$ 
6:  $G^M \leftarrow REMOVEOUTSIDE(G^M, AS_{biggest})$ 
7:  $repeat\_count = 3$ 
8: for  $i = 1$  to  $repeat\_count$  do
9:    $G^M \leftarrow REMOVEDEADENDS(G^M)$ 
10:   $G^M \leftarrow MERGEPOLYS(G^M)$ 
11: end for
12:  $G^M \leftarrow KEEPBIGGESTGROUP(G^M)$ 
13:  $G^M \leftarrow REMOVRAYS(G^M)$ 
14:  $G^M \leftarrow CUTEDGES(G^M, alphaShapes, AS_{biggest})$ 
15:  $G^A \leftarrow MERGEAREAS(G^M, alphaShapes)$ 

```

---

## 3.2 Area Graph 的创建过程

Area Graph 的创建是基于 Topology Graph (Schwertfeger 等, 2015) 的图结构发展而来的, 因此在阅读本节之前, 可通过回顾章节2.2.3来帮助理解, 其简要介绍了二维栅格地图的预处理步骤 (包括背景去除、边界检测、噪点去除和家具去除), 并简要概述了生成 Topology Graph 的边及其修剪步骤。

在本节中, 作者将详细描述为区域图生成区域的算法。该过程主要包括两个方面: 1) 初步生成多边形。其算法主要是基于创建 Topology Graph 的算法进行修改, 在生成 Topology Graph 的边时同步生成多边形, 并对多边形随边的变化进行相应的操作, 以确保边的合并或删除不会导致区域空洞或丢失; 2) 使用 Alpha Shape 检测开阔区域 (房间) 并将同一 Alpha Shape 中的多边形合并在一起成为

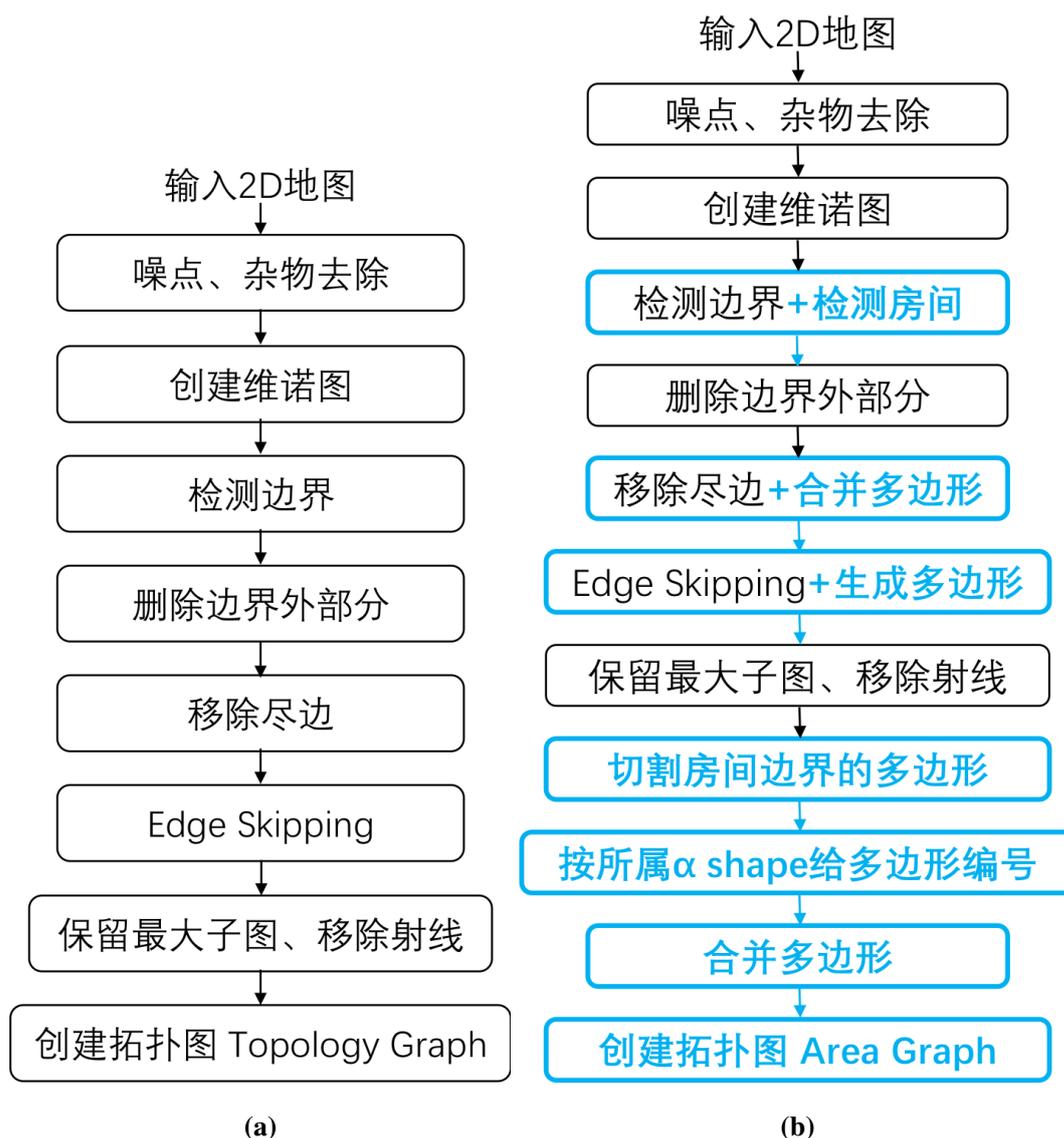


图 3.2 Topology Graph (a) 与 Area Graph (b) 生成流程。对比两个拓扑图的生成步骤，图 (b) 中的蓝色框是 Area Graph 生成算法相对于 Topology Graph 生成算法的改进部分，蓝色字体是完全新增的步骤。

Figure 3.2 The generation pipeline of the Topology Graph (a) and the Area Graph (b). Comparing the generation steps of the two graphs, the blue boxes in figure (b) are the improved steps of the Area Graph generation algorithm compared to the Topology Graph generation algorithm, and the blue fonts are completely new steps.

一个区域。这一部分是 Topology Graph 构建过程所没有的，因为区域图要以符合语义（房间和走廊）的区域作为拓扑节点，因此算法需要将地图分割为以房间或走廊为单位的区域。图3.2展示了 Area Graph 生成算法对比 Topology Graph 生成算法进行的改进，蓝色框是进行了大幅修改的算法步骤，蓝色字体是新增的算法

步骤。区域图中的一个区域对应于最终的  $G^M$  中的一个多边形。创建 Area Graph 的算法流程如算法1所示，表格3.1列举了算法中用到的函数。

### 3.2.1 生成多边形

对比图3.2的流程可以看出，Area Graph 的地图预处理步骤、维诺图生成和外围处理函数与在 Topology Graph 创建过程中使用的一致，这些都是对地图的预处理、对图的初始化及善后工作。而区域图构建的核心步骤，相对于 Topology Graph 是做了巨大改进的。对比表格2.1与表格3.1可以看出，Area Graph 中用于生成多边形的算法（函数），是基于 Topology Graph 的边创建函数发展的，主要是增加了对多边形的操作：1) 使用合并附加多边形（创建附加多边形）的函数 CREATEPOLY 替代了 Topology Graph 创建过程中仅合并（创建）边的函数 EDGESKIPPING，Area Graph 的初始多边形在这一步生成；2) 在删除特定尽头顶点及其连接边后并不删去其多边形，而是将这些多边形与邻居合并来进行保留，这样才不会导致最终生成的区域有空洞或缺失。也就是说，本算法修改了 Topology Graph 算法中进行边生成和尽头顶点删除的步骤，新增了多边形的相应操作。接下来，作者将使用两个小节详细阐述在生成 Area Graph 区域的算法中的这两个步骤。

#### 3.2.1.1 边跳过 (Edge Skipping) 与多边形生成

首先，算法使用从二维栅格地图生成的原生维诺图 VD 进行临时图  $G^M$  的初始化，并且删除与障碍物距离过近的维诺边。此时维诺图中的许多顶点失去了一条边，成为非 junction 顶点，在 Topology Graph 的生成过程中，算法通过跳过这些非 junction 顶点、串行连接其维诺边来生成 Topology Graph 的边。而我们在生成  $G^M$  图中的边时还需考虑其附加多边形的生成，因此，我们的算法提出在生成边的同时进行其关联的多边形的创建，这一操作过程在函数 CREATEPOLY 中实现，是改进了 edge skipping 算法。其中 edge skipping 步骤曾在章节2.2.3进行过详细的描述，此处侧重介绍其在 CREATEPOLY 中的改进方面。首先，我们仍然是以非 junction 的维诺顶点为途经点、将连接这些顶点的短半边（维诺边）首尾相连，来生成  $G^M$  的边，这一步与 edge skipping 中的步骤相同，可表达为式3.6。除此之外，算法还依序连接这些维诺边对应的 VD face，以生成多边形  $p_{ij} \in P^M$  附加到 Topology Graph 边上。该处理过程在图 3.1 中展示了示意图，以方便读者理解：算法按顺时针顺序将 VD 半边（有向的小短边）串连，再将末尾的 junction

步骤 (函数名)	功能描述
噪点去除 (REMOVE_OUTLIERS)	去除二维栅格地图中的噪点。
杂物去除 (ALPHASHAPEREMOVAL)	删去二维地图中的家具。
创建 Voronoi 图 (CREATEVORIGRAPH)	由二维栅格地图生成维诺图。
创建多边形 (CREATEPOLY)	在使用 edge skipping 生成 Topology 边的同时创建多边形。
移除外围部分 (REMOVEOUTSIDE)	删除维诺图在地图边界外的部分。
Dead End Removal (REMOVEDEADENDS)	删去长度短于阈值的 Dead End Edge 及其尽头顶点。
合并多边形 (MERGEPOLYS)	在移除 Dead End Edge 后将其上附加的多边形与邻居多边形合。
检测 $\alpha$ Shape (PERFORMALPHASHAPE)	从栅格地图中检测 $\alpha$ shape 以检测开阔区域。
保留最大子图 (KEEPBIGGESTGROUP)	仅保留最大 (边总长最大) 的连通子图。
去除射线 (REMOVERAYS)	删去图中指向无穷远的边 (射线)。
边切割 (CUTEDGES)	切割跨越区域的边。
合并区域 (MERGEAREAS)	将同一 $\alpha$ shape 区域内的多边形合并为一个区域。

表 3.1 用于生成区域图的函数及其描述，函数的使用展示于算法 1 中。

**Table 3.1 Functions for Area Graph generation, used in Algorithm 1, and their descriptions.**

顶点与末尾的半边对应的 face 中的 site 按顺时针方向相连，再将这些 VD 半边关联的 site 按顺时针顺序连接起来 (即 site 的连接顺序与其关联的维诺短边的连接顺序刚好相反)，最后将最后一个 site 与首部的 junction 相连，即可创建一个附加于新建的  $G^M$  半边的半多边形。该过程可以表达为

$$\begin{aligned}
 hpoly(h_{ij}) &= (v_i^{VD}, v_{k_1}^{VD}, v_{k_2}^{VD}, \dots, v_{k_m}^{VD}, v_j^{VD}, s_{k_m j}^{VD}, s_{k_{m-1} k_m}^{VD}, s_{k_{m-2} k_{m-1}}^{VD}, \dots, s_{k_1 k_2}^{VD}, s_{i k_1}^{VD}) \\
 &= (h_{ij}, s_j^{VD}, s_{k_m}^{VD}, s_{k_{m-1}}^{VD}, \dots, s_{k_1}^{VD}, s_i^{VD})
 \end{aligned}
 \tag{3.12}$$

其中， $s_{k_x k_{x+1}}^{VD}$  表示维诺半边  $h_{k_x k_{x+1}}^{VD}$  关联的 face 中的 site。

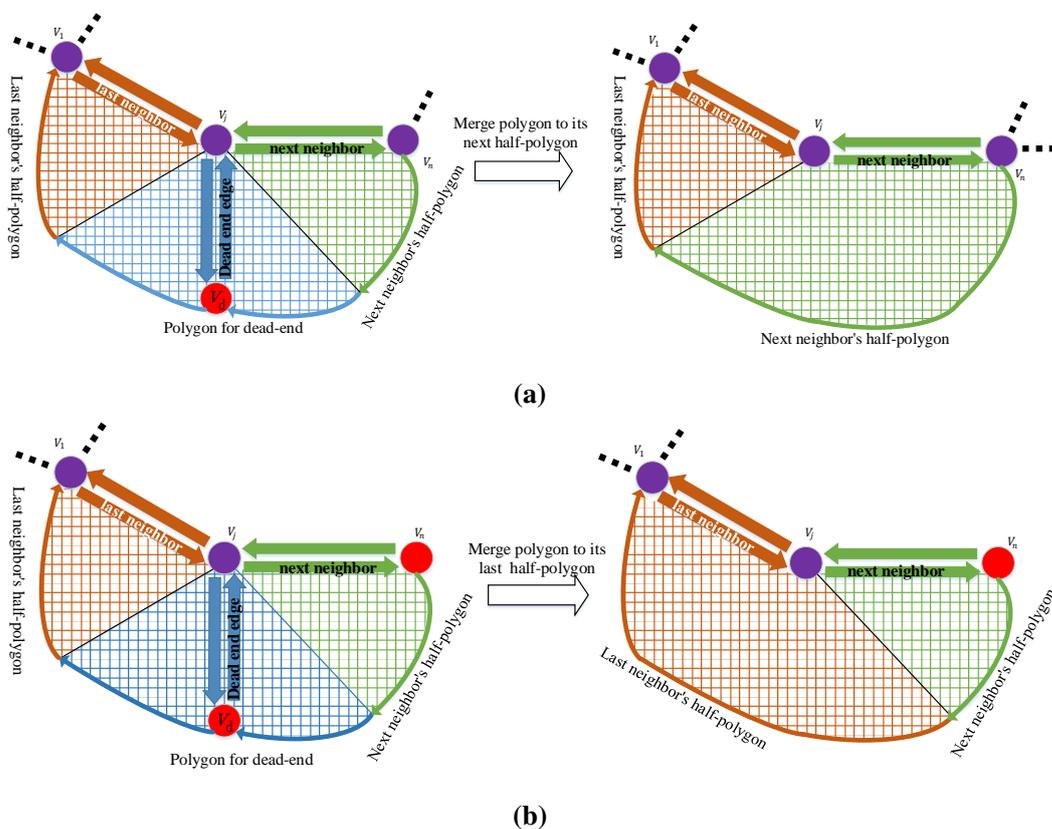


图 3.3 将附加于尽边的多边形合并到其相邻半多边形中的两种情况（这里只显示  $e_{dj}$  的相邻半多边形而不是整个相邻的多边形）。其中，红点代表 Dead End Vertex，与 Dead End Vertex 相连的边就是尽边，紫色点代表 Junction 顶点。(a) 当其相邻边不是尽边，将尽边多边形  $poly(e_{dj})$  与其后置半多边形  $hpoly(h_{jn})$  合并。(b) 将尽边多边形  $poly(e_{dj})$  合并到它的前置半多边形  $hpoly(h_{lj})$  中，因为  $h_{dj}$  的后置邻居是一条尽边。

Figure 3.3 The two cases of merging polygon of the dead-end edge into its neighbor's half-polygon (here only shows half polygons of  $e_{dj}$ 's neighbors instead of the whole polygon). Red points represent dead end vertex, the edge connected to a dead end vertex is a dead end edge. Purple points represent junctions.

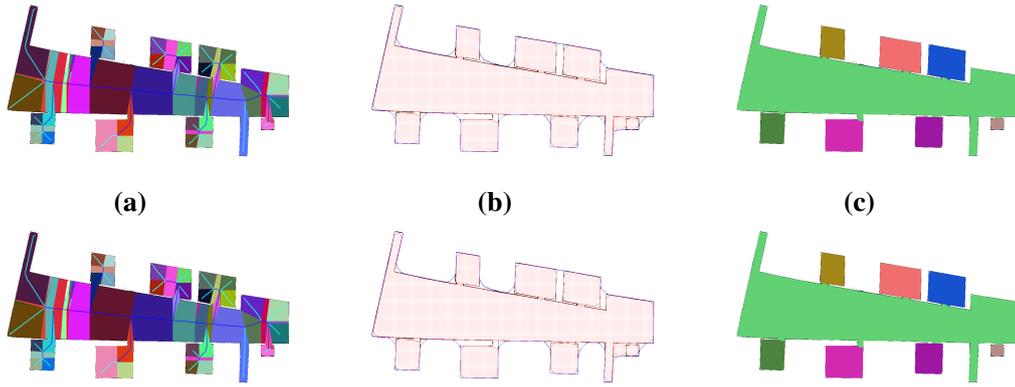


图 3.4 (a) Topology Graph 的边附加了多边形，显示为彩色的区域。图中的 Dead End Edge 显示为浅蓝色线条，连接 junction 顶点的边显示为深蓝色，这些边形成地图的骨架。图上附加的多边形是合并之前的区域。(b) 使用 Alpha Shape 检测以从网格地图中检测地图边界（显示为蓝线）和开放空间（房间）（显示为红色斑点图案的多边形）。(c) 相同 Alpha Shape 内的区域已进行了合并。

(a) The Topology Graph attached polygons for the edges, shown as colored areas. Dead end edges are shown as light-blue lines, edges connecting junctions are shown as dark blue skeleton. The polygons attached on the Topology Graph are areas have not be merged. (b) Alpha shape detection to get the boundary (blue line) and open space (room) (red patterns) from the grid map. (c) Areas within the same alpha shapes have been merged.

相应地，在建立一条半边  $h_{ij}$  的孪生半边  $h_{ji}$  时，孪生半边关联的半多边形  $hpoly(h_{ji})$  也会被创建（成为  $h_{ij}$  关联的半多边形  $hpoly(h_{ij})$  的孪生半多边形）。方向相反的一对孪生的半多边形被视为一个多边形，存储于  $P^M$ ，即  $G^M$  的多边形集合中。一个多边形关联一条  $G^T$  的边，也即  $G^M$  的边。此结构可描述为式3.8:

$$\begin{aligned}
 p_{ij}^M &= \{hpoly(h_{ij}^M), hpoly(h_{ji}^M)\} \\
 &= poly(e_{ij}^M) \in P^M.
 \end{aligned}$$

### 3.2.1.2 Dead End Removal

对连接的尽边 (Dead End Edge) 短于阈值的尽头顶点 (Dead End Vertex) 进行去除是为了对图进行简化，避免最终的骨架中出现过多杂枝。当  $G^M$  中一条尽边被移除时，附加到该边的多边形不能被移除，否则最终生成的区域将有缺损，因此算法将该多边形合并到它的相邻多边形中（函数 MERGEPOLYS）为了便于理解，作者在图 3.3 中展示了这个过程：如果一个尽边  $e_{dj} = \{h_{dj}, h_{jd}\}$  需要被删除，它

的多边形  $p_{dj} = poly(e_{dj})$  将被合并进入它的两个相邻的半多边形之一。这里将  $h_{dj}$  按顺时针顺序相连的下一条半边  $h_{jn}$  标记为  $h_{dj}$  (或  $e_{dj}$ ) 的后置半边, 按顺时针顺序在  $h_{jd}$  前一条相连的半边  $h_{lj}$  则称为  $h_{jd}$  (或  $e_{dj}$ ) 的前置半边。而前置半多边形和后置半多边形也按相应的规则定义。在算法中, 我们优先合并尽边的多边形  $poly(e_{dj})$  到它的后置半多边形  $hpoly(h_{jn})$ 。如果  $h_{jn}$  也是一个连接 Dead End 的半边, 那么  $poly(e_{dj})$  将被合并到它的前置半多边形  $hpoly(h_{lj})$  中。该策略有助于避免尽边多边形形成的区域不平衡。图 3.3 显示了合并死角多边形的两种情况。算法2展示了该多边形合并过程的伪代码。

---

**算法 2** 移除尽边并合并其多边形。

**算法 2** Remove a dead end edge and merge its polygon.

---

```

1: function REMOVEDEADENDS( $e_{dj}$ )
2:    $h_{dj}, h_{jd} \leftarrow half\_edge(e_{dj})$ 
3:    $hpoly(h_{dj}) \leftarrow half\_polygon(h_{dj})$ 
4:    $hpoly(h_{jd}) \leftarrow half\_polygon(h_{jd})$ 
5:    $h_{lj} \leftarrow last\_halfedge(h_{jd})$ 
6:    $hpoly(h_{lj}) \leftarrow half\_polygon(h_{lj})$ 
7:    $h_{jn} \leftarrow next\_halfedge(h_{dj})$ 
8:    $hpoly(h_{jn}) \leftarrow half\_polygon(h_{jn})$ 
9:   if IsDeadend( $h_{jn}$ ) then
10:      $hpoly(h_{lj}).polypoints \leftarrow h_{lj}.points + hpoly(h_{dj}).sites +$ 
        $hpoly(h_{jd}).sites + hpoly(h_{lj}).sites$ 
11:   else
12:      $hpoly(h_{jn}).polypoints \leftarrow hpoly(h_{jn}).points + hpoly(h_{dj}).sites +$ 
        $hpoly(h_{jd}).sites$ 
13:   end if
14: end function

```

---

在经过上面的边修剪和多边形生成及合并阶段之后, 多边形被添加并关联到图中的边, 存储于图  $G^M$  的多边形集合  $P^M$  中, 如图3.4的第一个子图展示了一个图  $G^M$  的可视化示例, 其每条边上都有关联的多边形。

### 3.2.2 区域的生成

为了达到将地图按房间、走廊等分割为有意义的区域的目标，我们的区域生成算法主要包括三个步骤：1) 检测地图中的开放区域，2) 分割跨越房间边界的多边形，3) 合并属于同一房间的多边形。最后将合并而成的新多边形的度量信息仍然存于图  $G^M$  中，作为区域，以作为最后的 Area Graph 节点。

#### 3.2.2.1 区域检测

维诺图在开放区域或房间内的生成会变得不稳定。这意味着房间墙壁上的微小变化将对维诺图的拓扑结构产生重大影响，这会严重影响我们的地图分割效果。理想情况下，我们不希望将房间分割成不同的区域，而是将每个房间分割为一个大面积，成为区域图中的一个节点。因此，我们采用房间检测算法来合并房间中的过分割区域。为此，我们使用 CGAL 的 2D Alpha Shapes 库，用于检测房间。

Alpha Shape 通常用于检测点集之中的开放空间。Alpha Shape 的大小取决于参数  $\alpha$ ，它是圆盘的平方半径。我们将给定参数的圆盘放在空间中（点集中的每两个点可以画两个给定半径的圆盘），若这些圆盘中不包含任何点则视为检测到了开放空间，然后通过用线段连接建立空圆盘的点来创建一个 alpha shape 多边形。

$\alpha$ -shape 检测到的边界内的开放空间被认为是房间。可以被检测到的房间的最小面积取决于参数  $\alpha$ ，它是可以放入检测到的房间的最大空圆盘的半径的平方，也就是说  $\alpha$  是决定要检测的最小区域的大小的。 $\alpha$  越大，检测到的最小的  $\alpha$ -shape 越大（检测到的最小房间的面积越大），则检测到的房间越少。

$\alpha$  的值是根据一个参数计算得出的：宽度  $W$ ，也可以看作是圆盘的直径。其被称为宽度是因为它主要是由建筑结构（比如门和走廊）的宽度决定的。要将房间检测为一个单独的区域，最重要的一点是要保证圆盘不能通过门。因此，圆盘半径不能小于门宽度的一半  $W_d$ 。而如果要将整个走廊分割为一个区域，则圆盘直径应小于最窄走廊的宽度  $W_c$ 。因此，当  $W_d < W_c$  时， $W$  合理的选择就是满足  $W_d < W < W_c$ 。章节3.3 部分通过实验分析了选择  $\alpha$  值的策略。

首先，我们将宽度  $W$  的单位从米转换为像素，然后从  $W$  到  $\alpha$  的转换可以通

过以下方式计算:

$$\begin{aligned} W_{pixel} &= \frac{W}{\text{Resolution}}, \\ \alpha = R^2 &= \left(\frac{W_{pixel}}{2}\right)^2. \end{aligned} \quad (3.13)$$

在图 3.4 的第二幅子图中, 我们展示了从地图中提取的 alpha shape ( $\alpha$  shapes)

---

**算法 3** 切割跨越区域的 alpha shape。

**算法 3** Cut edges across the alpha shapes.

---

**function** CUTEDGES( $G^M, \alpha$ Shapes)

**Require:**  $G^M = (V^M, E^M, P^M)$ ,  $\alpha$  shape list  $S_\alpha$

**Ensure:**  $G'^M = (V'^M, E'^M, P'^M)$

```

2:    $E'^M \leftarrow E^M$ 
   while True do
4:     for  $e \in E^M$  do
       for  $s_\alpha \in S_\alpha$  do
6:         if  $e$  and  $s_\alpha$  intersect at point  $p$  then
            $toCutMap[e] \leftarrow p$ 
8:         end if
       end for
10:    end for
       if  $toCutMap.isEmpty()$  then
12:        break
       end if
14:    for  $e \in toCutMap$  do
        $p \leftarrow toCutMap[e]$ 
16:        $e_1, e_2, poly(e_1), poly(e_2) \leftarrow \text{CUTEDGEANDPOLY}(e, p)$ 
        $E'^M \leftarrow e_1, e_2, P'^M \leftarrow poly(e_1), poly(e_2)$ 
18:        $E'^M.remove(e), P'^M.remove(poly(e))$ 
       end for
20:    end while
end function

```

---

的示例，图中将检测到的 alpha shape 多边形显示为红色的多边形图案。

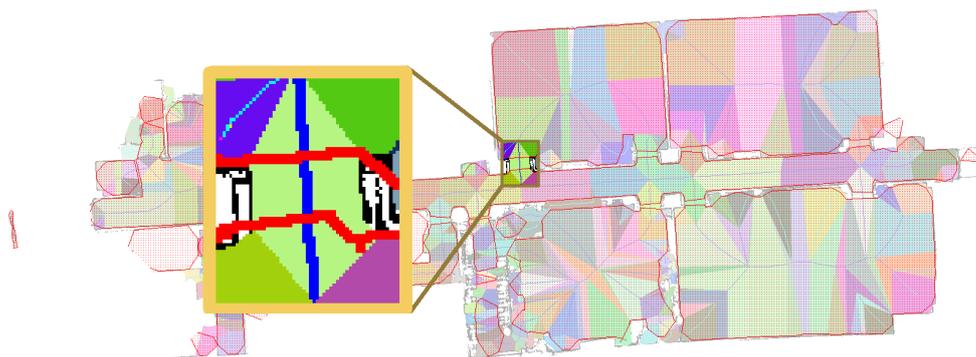


图 3.5 上图放大了一条边及其多边形与两个 alpha shape 的边界相交的例子，图中红线是用于检测区域的 alpha shape 的边界，蓝色的是图  $G^M$  中一条横跨多个区域的边，此边将被进行两次切割生成三条分属于三个区域的新边及三个相应的新多边形。

**Figure 3.5** The above image zooms in on an example where an edge and its polygon intersect the boundaries of two alpha shapes. The red curve in the figure is the boundary of the alpha shape used to detect the area, and the blue curve is an edge of  $G^M$ , which spans multiple alpha shapes. This edge will be cut twice to generate three new edges belonging to the three regions and three corresponding new polygons.

### 3.2.2.2 切割跨区域的多边形

在通过  $\alpha$  shapes 检测来寻找房间后，需要将  $G^M$  图中位于同一个房间内的多边形合并为一个区域来表示一个完整的房间。除此之外，由于  $G^M$  图中有的边跨越房间的边界（例如，穿过门），需要将这些边及其多边形在区域连接处断开。

如果一条边跨越了  $\alpha$  shape 的边界，即它的一个端点在  $\alpha$  shape 的内部，另一个端点在  $\alpha$  shape 的外部，并且该边不是长度短于阈值的尽边，则这条边将在通过点 (*passage point*) 处被切为两段，相应地，它的多边形将被切割为两个多边形。其中，通过点是边和  $\alpha$  shape 边界的交点。

跨区域的边及其多边形的切割算法在函数 `CUTEDGES` 中实现，具体步骤以伪代码的形式显示在算法 3 中。为了帮助读者理解该切割算法，此处我们对照算法 3 对其步骤进行详细的解释。在图  $G^M$  中，一条边有可能与多个 alpha shape 的边界发生相交，即一条边属于多个区域，则该边需要被多次分割，我们通过遍历所有的边进行检查并记录与区域边界交点（算法 3 第 4-10 行）。以图 3.5 为例，当两个房间的门刚好面对面， $G^M$  图中的一条边  $e_{ij}^M$  的一端  $v_i^M$  在房间 A 内，另一端  $v_j^M$  在房间 B 内，该边穿过房间 A、B 的门。当这两个房间被通过 alpha shape

检测到，则该边与检测这两个房间的 alpha shape 的边界共产生两次相交，分别交于点  $p_a$  和点  $p_b$ 。则算法将以  $p_a$  所在的位置为分界对边  $e$  分进行切割，并以这交点建立新的顶点  $v_a \in V^M$ ，使原来的边被切割为两条新边  $e_{ia}^M, e_{aj}^M$ ，其附属多边形也相应的被切为两个新的多边形（算法 3 第 14-19 行）。随后，进入下一次 while 循环以寻找是否还有边与 alpha shape 相交，并再次对这些边进行切割。

### 3.2.2.3 合并同一区域的多边形

首先我们对所有 alpha shape 进行编号，随后通过检查每个多边形的边属于哪个 alpha shape 来为其多边形分配 *roomID*，属于同一个房间的所有多边形都被分配了相同的 *roomID*。然后将具有相同 *roomID* 的多边形合并为一个多边形，代表一个区域，该过程实现于函数 MERGEAREAS。图 3.4 的最后一幅图显示了完成多边形合并后的区域。可以看出，图中的区域皆是符合房间或者走廊等语义结构的。当然，我们可以根据不同的需求通过改变检测的参数来使生成的区域代表不同的语义，例如一个区域可以是整条走廊，其沿途允许有分叉口，也可以是一段没有分叉口的走廊，则一条走廊会被分为多个区域。

### 3.2.3 创建区域图

通过以上步骤，我们已经将地图分割为不同的区域，由此可以创建区域图  $G^A$ ： $G^M$  图中的区域被作为区域图  $G^A$  的顶点  $v^A$ ，且拓扑地图  $G^A$  中使用区域之间的通道作为区域图  $G^A$  中的边  $e^A$ ，连接相邻的区域节点。从建筑结构的意义来说，通道即两个区域的相连之处，比如门，穿过通道则可以跨越不同的区域。在图  $G^A$  中，通道的度量信息记录了一个坐标，即其在  $G^M$  图中对应的  $v^M$  的位置，还有一条线段，即两个相邻区域的共同边界。

## 3.3 参数 $\alpha$ 的确定

本章将进行实验来分析  $\alpha$  参数值对地图分割结果的影响，并讨论该如何确定用于分割的参数  $\alpha$ 。为此，我们使用 Matthew 的相关系数 (Matthew's Correlation Coefficient, MCC) (Mielle 等, 2017) 对比人工给定的参照结果作为真值 (Ground Truth) 进行分割结果的评估。MCC 的计算如下：

$$MCC = \frac{tp * tn - fp * fn}{\sqrt{(tp + fp)(tp + fn)(tn + fp)(tn + fn)}} \quad (3.14)$$

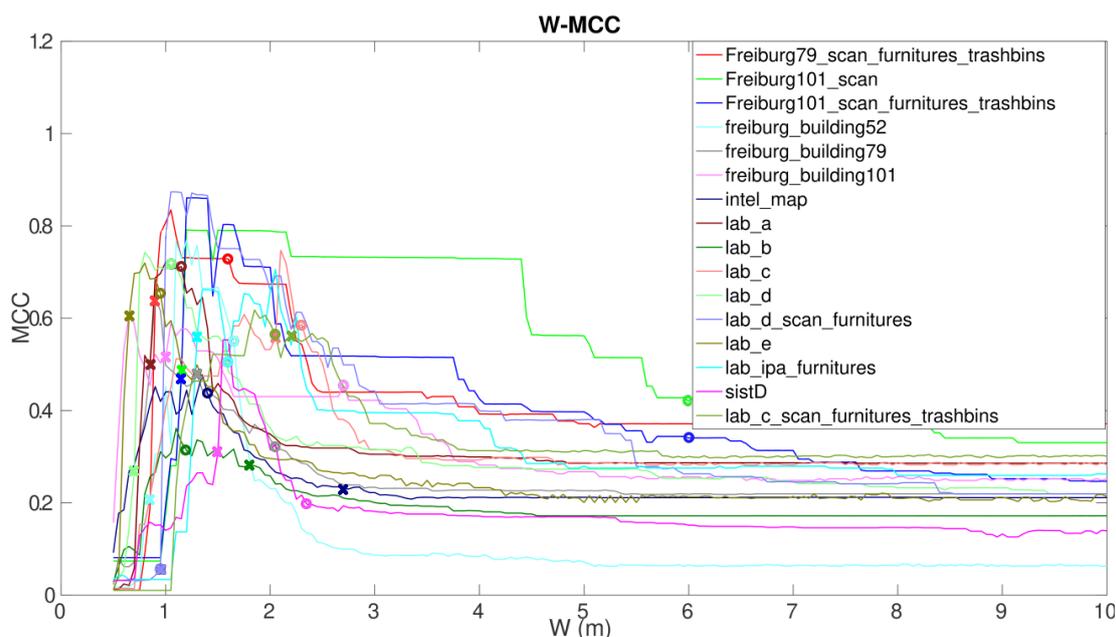


图 3.6 根据不同参数  $W$  进行分割的结果使用 Mielle 等 (2017) 的标准 MCC 进行分割质量评估。图中横轴是参数  $W$ ，它决定了我们算法中的参数  $\alpha$ ，纵轴是由该参数产生的分割结果的 MCC 分数。对于每条曲线，在  $W = W_d$ （最宽的门宽）和  $W = W_c$ （最窄的走廊宽度）处的 MCC 分数分别用 “x” 和 “o” 标记。

Figure 3.6 Use MCC score from Mielle 等 (2017) to evaluate segmentation quality according to  $W$ , which decides the  $\alpha$  in our algorithm. The MCC score at  $W = W_d$ , the widest door width, and  $W = W_c$ , the narrowest corridor width, are marked with “x” and “o”, respectively, for each curve.

其中， $tp, tn, fp, fn$  分别是指对分割结果的某个区域中，每个像素点与真值相比是真阳性（正确地分割为属于该区域）、真阴性（正确地分割为不属于该区域）、假阳性（错误地分割为属于该区域）和假阴性（错误地分割为不属于该区域）。因此，MCC 可以用来量化地评价算法分割结果与真值的吻合度。MCC 分数的范围是从 -1 到 1，最好的分割质量会获得  $MCC = 1$  的分数。因为本文的算法是为真实环境的地图分割而设计的，所以实验运行在来自 Bormann 等 (2016) 的 Bormann 数据集的子集上，该子集去除了 Bormann 数据集中的人造地图（人造地图即非从真实环境扫描获得的地图），仅包括非人造地图和包含家具的地图。另外，实验数据包括我们校园的建筑物的扫描地图（地图 *sistD*）。

考虑到门不会窄于  $0.5m$ ，而室内走廊几乎不会宽超过  $10m$ ，本章的实验测试宽度设置为从  $W = 0.5m$  到  $W = 10m$ ，本文的分割算法根据这一系列的  $W$  算出来的  $\alpha$  运行产生分割结果，本实验使用 MCC 分数对这些地图分割结果进行评

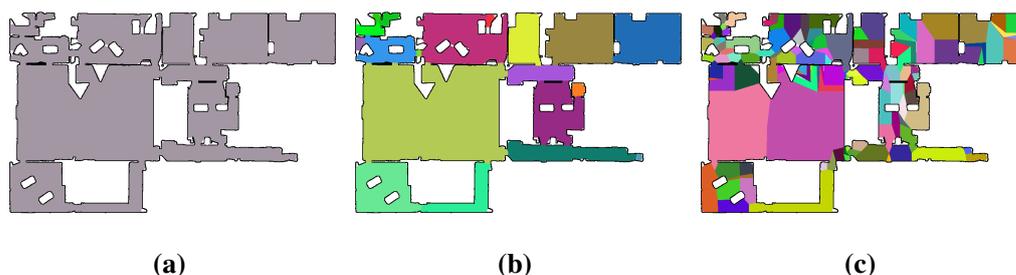


图 3.7 为 Alpha Shape 检测算法设置三个不同的参数  $W$  值产生的对地图  $lab\_d$  的分割结果。

(a) 设置  $W = 0.5m$  导致地图欠分割, 结果的 MCC 得分分别为 0.03; (b)  $W = 0.8m$  时分割结果良好, 结果的 MCC 得分分别为 0.74; (c)  $W = 8.55m$  时 alpha shape 没有检测到任何房间, 导致地图被过分割, 结果的 MCC 得分分别为 0.23。

**Figure 3.7 The segmentation results with different  $W$ s for the alpha shape room detection of map  $lab\_d$ . (a)  $W = 0.5m$  leads to under-segmentation, whose MCC score is 0.03; (b)  $W = 0.8m$  gives good results, whose MCC score is 0.74; (c) When  $W = 8.55m$ , alpha shape doesn't find any rooms, which leads to over-segmentation, whose MCC score is 0.23.**

估, 这些分割结果的 MCC 分数与对应参数  $W$  的曲线如图 3.6 所示。

图 3.6 表明, 对于较小的宽度  $W$ , 由于 alpha shape 可以穿过门, 一个 alpha shape 将吞没整个空间, 将地图的所有部分都检测并合并为一个区域 (见图 3.7a): 此时欠分割的情况发生, 即将应该分割为不同区域的部分合并为同一个区域。

而在另一个极端, 设置较大的  $W$  值会导致 alpha shape 检测到很少甚至没有开放空间, 从而最终拓扑图中的区域是纯基于 Topology Graph 的边生成的区域, 这些区域会使房间被过度分割 (见图 3.7c), 即将本该属于一个的区域分成了多个区域。

观察图 3.6 可发现, 当  $W_d < W_c$  时, 宽度  $W$  的最佳设置为比地图中最宽的门稍大; 而对于  $W_d > W_c$  的情况, 曲线的峰值出现在略小于  $W_c$  的  $W$  处。计算在  $W_d \leq W_c$  的情况下的最佳设置与门宽的差  $W_{best} - W_d$  的平均值, 为  $0.1m$ ; 以及计算了在  $W_d > W_c$  的情况下的  $W_{best} - W_c$  的平均值, 为  $-0.1m$ 。因此, 建议宽度  $W$  的设置策略是: 对于  $W_d \leq W_c$  的地图, 使用  $W = W_d + 0.1m$ , 而在  $W_d > W_c$  的情况下选择  $W = W_c - 0.1m$ 。

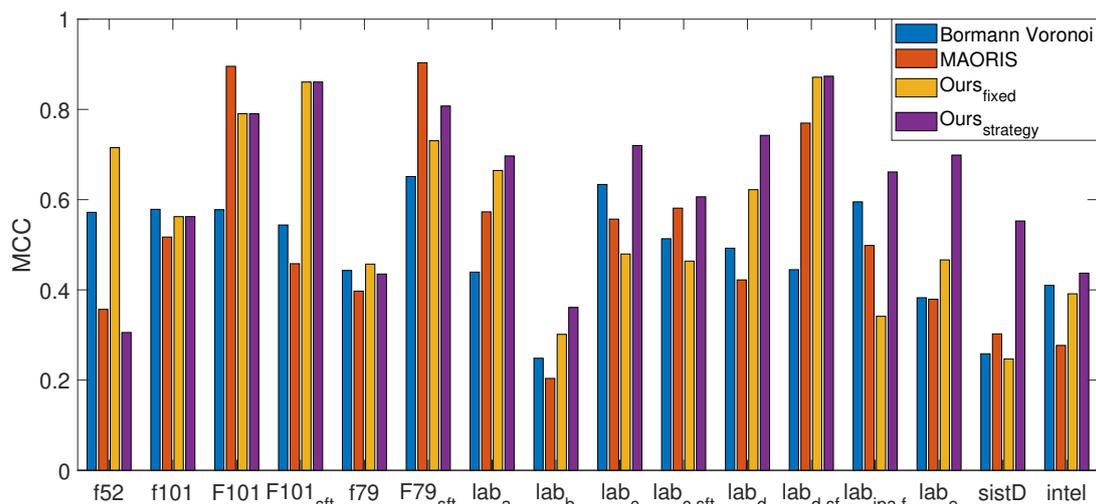


图 3.8 三种地图分割方法的结果对比。分数高的则效果好。

Figure 3.8 Comparison of the segmentation results from three methods. Higher is better.

### 3.4 与地图分割相关工作的对比实验

为了生成以区域为节点的拓扑地图 Area Graph，作者提出的 Area Graph 生成算法首先对地图进行了分割。为了展示本算法的地图分割质量，在本章，使用本文的 Area Graph 区域生成算法与 MAORIS 方法 (Mielle 等, 2017) 和由 Bormann 等 (2016) 实现的传统基于 Voronoi 的分割进行对比实验。章节 1.2.1.1 简要介绍了这两种方法。

#### 3.4.1 在普通的室内地图上的对比实验

本实验将我们的区域生成算法视为一种地图分割算法，与广为使用的由 Bormann 等人实现的基于维诺图的分割算法和 MAORIS 算法进行地图分割质量的比较。仍然是使用 MCC 分数、对比给定的地图分割真值作为参照结果来对地图分割的质量进行评估。在实验中其他两种分割算法使用他们各自论文中描述的最佳参数。由于它们的参数是固定的，为了公平，本文算法使用  $W = 1.25m$  来进行对比实验，其中  $1.25m$  是测试数据集上最佳结果的平均  $W$ 。但在对比实验中也会展示了根据上一章（章节 3.3）中描述的策略设置的  $W$  参数的算法分割结果，以显示该策略的优势。

本章的对比实验在 Bormann 数据集 (Bormann 等, 2016) 子集和我们收集的地图 *sistD* 上分别运行 MAORIS<sup>1</sup> 分割算法、Bormann 实现的基于 Voronoi 的分割

<sup>1</sup>MAORIS 的代码已开源在 <https://github.com/MalcolmMielle/maoris>

算法和本文提出的 Area Graph 生成算法来产生各自的地图分割结果。这些测试地图共有 16 张，不仅包括简单的地图，还包括家具地图和不完整的地图。由于本文在预处理步骤中去除了每张输入地图中的噪点和家具、杂物，为了公平，本实验中使用的所有三种方法都是以已移除噪点和家具杂物的地图作为输入。对于用于评估的参照结果，实验使用 Bormann 提供的地图分割真值（该数据集中有提供真值），对于不属于 Bormann 数据集的地图，作者让与该项目无关的学生对地图进行分割的结果作为真值。

本文的算法仅将输入的地图图片中的黑色像素点作为生成维诺图的输入基点集。在去噪阶段，预处理算法将非黑色的像素绘制为白色，包括灰色的背景。作者在之前的会议论文中使用去除了灰色背景的地图作为输入运行了上述的三种分割算法进行对比实验。但是当后来重新进行此实验时，作者发现 MAORIS 算法和 Bormann 的 Voronoi 分割算法在白色背景的地图上运行需要更多的时间，并且与灰色背景的地图相比，这两个算法在白色背景的地图上的表现更差。这是因为这两种算法都没有检测地图边界，因此也对背景进行了分割。为此，本实验在运行两种算法进行比较时恢复了地图的灰色背景。本实验中用到所有地图数据集、以及我们提出的 Area Graph 的生成算法实现都已经在网上开源<sup>2</sup>。

人为定义环境的分割真值有时非常主观，因此 MCC 分数并不反映唯一的正确答案。出于这个原因，除了展示各分割结果的 MCC 分数，我们还在图 3.9 中展示了一些由这三种方法产生的地图分割结果，并在后文中对结果进行了分析。MCC 分数的比较结果在图 3.8 中绘制为柱状图，并在表 3.2 中进行了总结。在  $\frac{10}{16}$  的数据上，本文的方法在设置固定参数  $W = 1.25m$  的情况下表现优于 Bormann 的方法和 MAORIS 方法。当使用章节 3.3 中描述的策略设置的  $W$  参数运行算法时，本文的方法在  $\frac{13}{16}$  的数据上获得了更高的 MCC 分数。

数据集中，地图  $f52$ 、 $F79_{sft}$ 、 $lab_c$ 、 $lab_{ipaf}$  和  $lab_{csft}$  是完整的地图，只包含很少的家具，没有大的开放空间，这在真实的建筑物环境中是几乎没有的。虽然这些噪音很小及杂物很少的地图可以从高质量的地图构建方法 He 等 (2019) 或使用本文的家具移除方法来获得，但在实际情况下，地图中通常包含复杂的家具，一旦在地图构建的过程中扫描到，就很难通过自动移除的预处理算法进行移除。本文的方法在地图  $lab_{ipaf}$  和  $f52$  的分割结果中得分较低主要是由于这两个地图

<sup>2</sup><https://github.com/STAR-Center/areaGraph>

的分割参照结果是较为主观的。在固定  $W$  参数的情况下，由于建筑中的门非常宽，本方法在地图  $lab_c$ 、 $lab_{ipaf}$  和  $lab_{csft}$  上的得分较低。但是，用上一章提出的策略决定参数  $W$  会改善结果。

地图  $F101$ 、 $F101_{sft}$ 、 $lab_d$ 、 $lab_{dsf}$  和  $sistD$  也是没有复杂家具的完整地图，但它们包含大的开放空间。一旦大空间中包含从墙壁突出的障碍物，基于 Voronoi Graph 的 Bormann 的算法会检测到额外的关键点，而 MARIO 算法会检测到突然的像素距离值变化，两者会导致开放空间区域被过度分割。由于使用 alpha shape 检测和合并房间，的方本文法获得了更高的分数，在使用固定的  $W$  参数和根据策略设置参数的情况下，本文的方法在这些地图上的平均分分别提高了至少 19.10% 和 34.15%。

如地图  $f79$ 、 $f101$ 、 $intel$  和  $lab_b$ ，这些区域当中有家具和噪声的地图，包含大量嘈杂障碍物。尤其是地图  $lab_b$ ，其中的扫描噪点和杂物非常多，图2.4展示了机器人扫描的原始  $lab_b$  地图、去除噪点处理后的地图和去除家具等杂物之后的地图。可以看出，尽管预处理有很大帮助，但地图中仍会存在难以去除的障碍物，使分割方法难以对地图中的区域进行完整的分割，表 3.2（第四列）在该类型地图的分数显示，用于对比的三种方法在此类地图上都没有提供好的分割结果。本文的算法的分割结果得分稍微高一些。

地图  $lab_a$ 、 $lab_b$  和  $lab_e$  是不完整的环境扫描图，这容易同时导致地图被欠分割和过度分割。如表 3.2 的最后一列所示，由于我们的算法可以成功检测到部分房间，并且检测地图的边界，算法可以将不完整的房间检测为区域并将它们与未知区域（背景）区分开来。不论在选择固定参数还是按策略选择参数的情况下运行算法，本文的算法的分割结果表现明显优于其他方法。

基于 Voronoi 图的分割算法 Bormann 等 (2016) 的 MCC 的平均分数为 0.49，MAORIS 方法 Mielle 等 (2017) 的 MCC 的平均分数为 0.52，我们的方法运行于固定  $W$  和按策略选择  $W$  的两种情况下的 MCC 平均分分别为 0.56 和 0.63。因此，实验显示本文的方法比基于 Voronoi Graph 的 Bormann 的方法、MAORIS 能更好地分割地图，特别是在家具、大开放空间和不完整的地图上。

此外，实验测量了三种算法的平均运行时间，如表 3.3 所示，本文的方法是最快的。所有三种方法都以最少的时间分割地图  $f101$ ，此地图很简单，只有六个房间，没有噪音或家具。MAORIS 和本方法对地图  $lab_e$  的分割耗费最多时间。

表 3.2 测试地图分为四种：(1) 完整的地图，家具少，很少有大的开放空间；(2) 完整的地图，家具少，且包含大的开放空间；(3) 区域中有很多家具的地图；(4) 不完整的地图。然后，对不同方法我们计算其对于不同类型地图的分割结果的 MCC 平均分数。

**Table 3.2 The maps are divided into four types: (1) complete maps with few furniture and without big open space; (2) complete maps with few furniture and including big open space; (3) maps with lots of furniture in area middle; (4) incomplete maps. Then we compute the MCC average scores of segmentation results by different methods in different types of maps.**

MCC average	Complete, few furniture		Furniture in middle	Incomplete
	Small space	Big space		
Bormann	0.593	0.463	0.420	0.363
Voronoi				
MAORIS	0.592	0.570	0.406	0.385
Ours (fixed W)	0.546	0.678	0.428	0.478
Ours (strategy)	<b>0.620</b>	<b>0.764</b>	<b>0.506</b>	<b>0.587</b>

而 Bormann 的算法用于分割地图 *intel* 的时间最多。这两张地图都不完整且嘈杂，建筑结构复杂，房间很多。可见，房间越多、环境越复杂，则地图分割算法耗时越长。其中，房间合并步骤在所有三种方法中都是非常耗时的。

### 3.4.2 迷宫地图对比实验

除了常规的建筑室内地图外，作者还在迷宫地图上进行地图分割实验，以突出本算法可以根据实际的拓扑结构需要进行地图分割的能力。对于走迷宫的应用，其拓扑结构需求倾向于将没有分支的路径划分为一个区域，即在交叉路口，使得提取的拓扑地图只在代表交叉口处进行选择，这样的拓扑地图更符合基于拓扑结构的路径规划的需求。在迷宫应用中，不同分支路径中的部分不应合并为一个区域，否则则定义为欠分割；同理，属于同一分支的路径不应该被分割，否则则定义为过度分割。

由于本文的方法是基于 Voronoi 图结构进行分割的，因此在交叉路口处图  $G^M$  的边会产生分叉，使算法可以很方便地在交叉口根据分支边划分区域，则交叉口连接的不同方向可作为不同的区域，而没有交叉路口的长走廊仅划分为

表 3.3 三种地图分割方法的运行时长比较。除了计算它们的平均运行时间外，该表还显示了这些算法分割耗时最短或最长的地图。

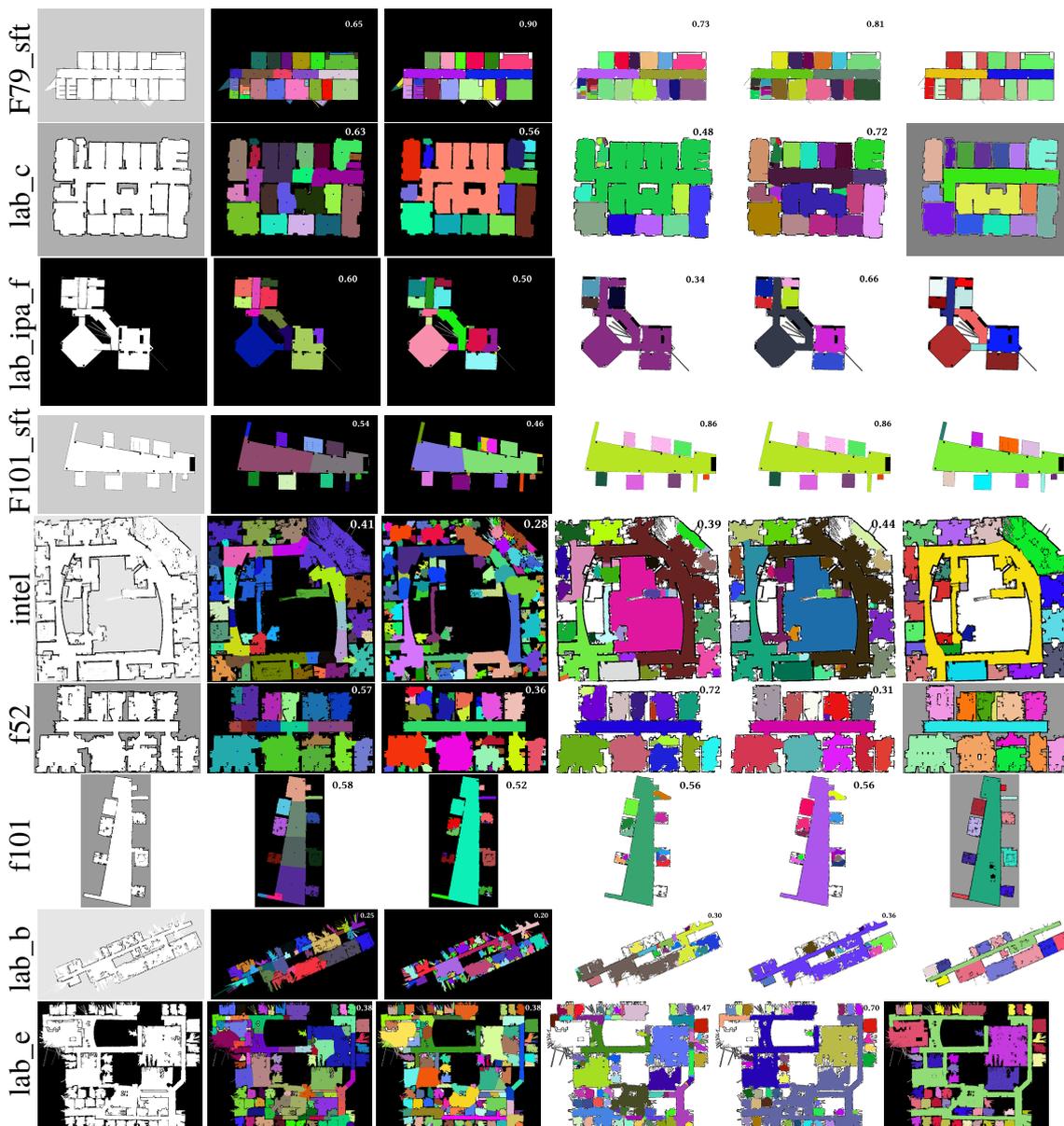
Table 3.3 Runtime comparison of the three segmentation methods. Besides computing their average runtime, the table shows these algorithms segment which maps with least or most time.

Runtime (s)	Average	Min	Map	Max	Map
Bormann's	6.29	0.73	f101	22.32	intel
MAORIS	17.50	2.91	f101	55.78	lab_e
Ours (fixed W)	<b>1.31</b>	<b>0.25</b>	f101	<b>3.84</b>	lab_e
Ours (strategy)	1.38	<b>0.25</b>	f101	4.22	lab_e

一个区域表示。本实验仍是使用本文的区域生成算法与基于 Voronoi Graph 的 Bormann 的方法和 MAORIS 的方法进行对比，图3.10展示了两个迷宫环境的地图及对其使用不同算法的分割结果。

在人造迷宫地图中，我们将  $\alpha$  shape 检测算法的参数值  $\alpha$  设置得很大，大于走廊宽度，则相当于房间检测算法在区域生成过程中不被使用。可以看出，本文的算法很好地构建了适用于迷宫环境的拓扑表示，区域皆为从尽头到交叉口或交叉口之间的路径。而其他两种算法无法生成有意义的区域。在这张地图上，这三种方法的结果与真值相比的 MCC 分数分别为 0.43、0.06 和 0.74。最后一个分数不是 1 是因为人工给定的真值使用了不同的策略来划分区域的连接处，因此造成了一些差异，这种划分方式与我们的划分方式都是合理的。

第二个迷宫环境来自 RoboCup Rescue 比赛现场。该地图在沿墙壁处有一些噪点，导致了过度分割。由于路口分割是相当主观的，本文的方法在路口有一些过度分割。对于其他两种方法，MAORIS 的方法发生欠分割，Bormann 的过度分割。在这张图上，三种方法的 MCC 得分分别为 0.37、0.32 和 0.44。可见，对此地图，本文的结果也不完美，但仍然是最好的。这些结果表明，本文的方法比其他方法更好地提取了环境的底层拓扑结构，因为它可以直接基于维诺图创建拓扑表示。



(a) 输入 (b) Bormann (c) MAORIS (d) 固定参数 (e) 可变参数 (f) 真值

图 3.9 比较三种不同的地图分割方法的分割结果。上图将地图中分割好的不同区域涂为不同的颜色以便区分。此外，每个结果与参照结果（真值）的对比获得的 MCC 分数展示在每幅子图的右上角。

Figure 3.9 Comparison of the segmentation results from three different methods. We paint different areas of the map in different colors to easlyer distinguish them. Additionally the MCC score of the comparison with the ground truth segmentation is depicted in the upper right corner of each result.

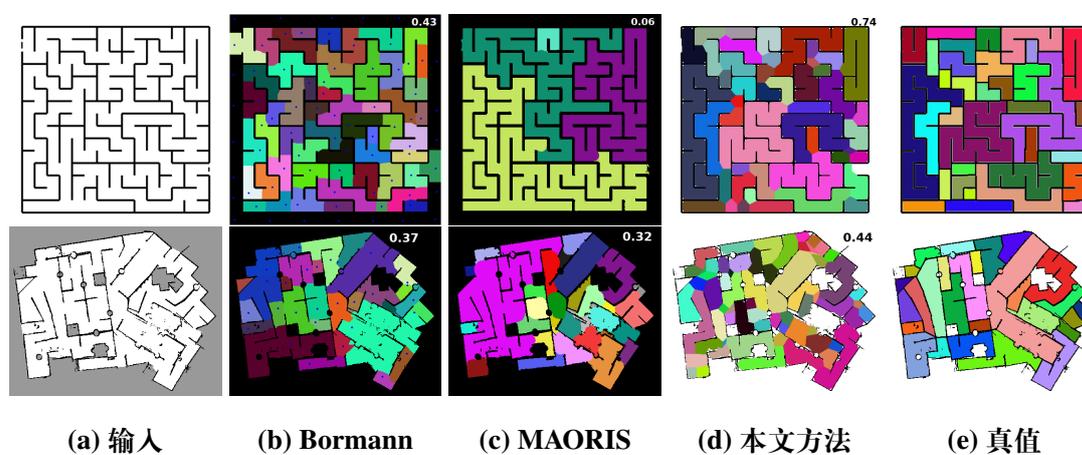


图 3.10 比较不同方法对迷宫地图的分割，遵循图3.9的风格进行结果展示。我们的算法使用了一个很大的参数  $W$  值，有效地禁用了 alpha shape 房间检测算法。

Figure 3.10 Comparison of the segmentation of the different methods on mazes, following the style of Fig. 3.9. Our algorithm uses a big value for  $W$ , effectively disabling the alpha shape room detection.

### 3.5 开源代码

作者对提出的区域图 Area Graph 的构建算法进行了代码开源。该代码及数据集公布在了网站 <https://github.com/STAR-Center/areaGraph>。在该代码库的首页展示的"README.md"文件中,作者描述了如何编译及使用该代码。该代码是在 Linux 系统上开发及运行的,读者在使用之前需先安装 g++, cmake, Qt4, CGAL。用户首先输入地图图片并设置参数,运行"example\_segmentation"来对地图进行分割。

该代码库中,作者还附上了会议论文 Hou 等 (2019b) 中地图分割评估实验所使用的数据集。其中主要是来自 Bormann 等 (2016) 文章的数据集,该数据集包含不同建筑结构的地图,利于我们测试对不同建筑的地图分割效果。由于 Bormann 在文中进行了对地图分割算法的测试,他在数据集中标明了地图分割的参照结果,本章的实验使用了这些参照结果作为真值对分割算法的表现进行了量化的评估。此外,作者贡献了其所在的学院楼二楼 D 区的三维点云压制成的二维地图,该地图是通过高精度的 FARO 三维激光扫描仪扫描环境获得的,地图所涉及的环境面积约 1700 平方米,分辨率为 0.05 米。

### 3.6 本章小结

本章是对区域图 Area Graph 的拓扑结构表达及其构建算法的阐述和介绍。区域图的数学表达及其构建过程中涉及的元素和符号比较多,为了帮助读者理解,本章的第一小节列举了维诺图、Topology Graph 和区域图的数学符号与表达,并简要地介绍了这些表达的形成。随后,作者详细地介绍了区域图的构建过程,该过程包括: 1) 从 Topology Graph 的边创建多边形; 2) 对 Alpha Shape 检测算法检测到的房间及走廊等,进行多边形切割及合并以生成完整区域; 3) 以区域为节点构建拓扑图。随后通过实验讨论参数  $\alpha$  的设置。最后的对比实验分为两部分,前面一部分是比较不同方法对常规建筑的室内地图的分割结果,另一部分比较不同方法在迷宫地图上的分割结果。实验展示了本文提出的地图分割方法(区域生成方法)对地图的分割更符合人类给出的参照结果,即更符合人类直觉上的“有意义的区域”。尤其是在迷宫应用中,本文的方法比其它对比方法有更显著的出色表现。



## 第 4 章 基于区域图的路径规划方法

将栅格地图分割成适当的区域是机器人技术中许多应用的一项重要任务。原因是由此提取的拓扑表示大大降低了算法的搜索空间、缩短了计算时间，路径规划是拓扑地图的基本应用之一。路径规划几乎是所有移动机器人应用都会要求的部分，是机器人学中的一个主要的研究内容。在空间中寻找起点位置和终点位置之间的一系列点序列或边序列形成的路径称为路径规划。路径规划本质上是对空间形成的拓扑图进行搜索。在点云或栅格地图表示中进行路径规划，算法的搜索空间是点云中的点或栅格地图中的单元格。本文提出的 Area Graph 拓扑表示，是对空间进行一个区域的提取，形成更高层次的拓扑表示，拓扑图的节点不再是栅格地图中的单元格，而是表示房间或走廊分支的区域。使用路径规划算法在我们提取的拓扑表示 Area Graph 上进行路径规划，则算法的搜索空间大小将从地图单元格数量变为地图的区域数。地图的单元格数量通常由分辨率决定，例如一个 5 米 × 4 米 = 20 平方米的房间，以 0.05 的分辨率（地图中每个单元格表示 5cm × 5cm 大小的实际环境）构建地图，则单元格数量可达到 8000 个，而对其提取区域则可识别为一个区域。因此使用以区域为节点的拓扑表示进行路径规划，可使搜索空间大小有数量级的下降，因此可大大提高算法计算速度。为了展示提取区域为节点的拓扑地图在这种应用上的优势，我们在本章展示基于我们提出的 Area Graph 的路径规划应用。

### 4.1 Passage Graph 的创建

#### 4.1.1 算法原理

对于在拓扑地图上做全局路径规划，首先要通过  $A^*$  搜索算法 Hart 等 (1968) 找到从起点节点到终点节点的路径，这里的“路径”是指一条从起点到终点的节点连接通路，即可在拓扑图层次从起始节点转移到终点节点的边序列。然而，为了使机器人在真实环境中的不同区域之间进行移动，还需要构建度量图层面的路径，即获得在栅格地图中从起始点到终点之间连通的点序列作为路径。但是，我们不能在线上规划时再在栅格地图层次搜索单元格序列作为路径，否则搜索空间的大小仍然是单元格数量。为此，本文从 Area Graph 提取路径来构建通道

图 (Passage Graph), 通过在离线时预先提取穿越区域的路径, 使在线上规划时不再需要进行单元格数量级的搜索, 而只专注于拓扑图层次的搜索。

Area Graph 中连接相邻区域的连接处定义为通道, 对于 Area Graph 中任何给定的区域, 已知该区域的所有通道都可通过 Topology Graph 生成的骨架连接。由于通道是区域的出入口, 一条通道总是由至少两个区域共享, 因此, 机器人可以通过通道来实现区域间的转移。那么, “到达某通道所在的位置” 即等价于 “到达或离开该通道所连接的区域”。也就是说, 如果创建连接通道之间的机器人可行路径, 那么就相当于允许机器人以 “进入区域 A-离开区域 A-进入区域 B-离开区域 B-进入区域 C-……” 这种串联区域的方式不断进行区域之间的转移, 即创建了适用于机器人在真实环境中穿梭于区域间的路径。基于此原理, 本文创建通道图 Passage Graph。Passage Graph 以 Area Graph 中连接区域的通道 (passage) 为节点, 而 Passage Graph 中连接这些通道节点的边存储的度量信息, 是在栅格地图中连接这两个通道位置的路径。因此, Passage Graph 可以看作是 Area Graph 的对偶图, 因为 Area Graph 中的边的度量信息 (通道) 是 Passage Graph 中的节点, 而 Area Graph 节点中的度量信息 (区域中的骨架) 是 Passage Graph 中的边。在接下来即将描述的构建 Passage Graph 的算法中, 主要的一个目标是在线下构建连接通道的路径, 也就是从 Area Graph 提取连接通道的路径。

对于通道图的边的度量信息, 在线下创建通道之间的路径并预存在 Passage Graph 中, 计算并记录每个区域中从一个通道节点到另一个通道节点沿着基于维诺图的骨架形成的路径的距离。在线上需要进行全局路径规划时, 可使用  $A^*$  搜索算法搜索起始与终点区域之间的节点路径, 从而获得了起点和终点之前的大部分机器人可行路径, 但是, 从起点到离线路径以及从离线路径到终点的路径还是缺失的, 因此要将起点与终点位置连接到通道图的离线路径中才可完成完整的全局路径规划。在收到进行路径规划请求时, 算法将新建两个新的 (虚拟) 通道添加到通道图中: 一个在起点位置, 一个在终点位置, 这两个虚拟通道节点属于它们所在的区域。随后算法生成新节点到其所在区域的离线路径的连接: 用直线连接新节点与离线路径上最近的一个点, 并计算从新节点到区域中所有通道节点沿着骨架路径的长度。这样, 在通道图上使用  $A^*$  来串联通道节点 (包含起点和终点这两个虚拟通道节点) 则可获得起点和终点之间的完整路径。

### 4.1.2 算法描述

Passage Graph 记为  $G^P = (E^P, V^P, A^P)$ ，该图强烈依赖于先前的拓扑图表示。这里  $A_p$  是 Area Graph 区域单元的集合，每个  $v \in V^P$  是一个通道节点， $e \in E^P$  是边，每条边连接两个通道节点，每条边的度量信息都将记录其在该区域连接的一个通道到另一个通道的路径。此外，每条边都会在  $G^P$  中获得一个区域 ID，以标记它属于哪个区域的。

要为  $G^A$  构建通道图  $G^P$ ，对于房间检测后已经进行过合并的区域，例如图4.1b的示例，如果区域中包含有两个以上的通道，我们则需要在每对通道之间找到一条路径作为边的度量信息进行存储。为此，我们使用  $A^*$  算法在该区域的 Topology Graph 上进行搜索：两个通道节点之间可以通过一系列 Topology Graph 节点使用 Topology Graph 的边进行连通。该步骤可以表示为

$$\begin{aligned} e_{ij}^P &\in E^P \\ &= (v_i^P, v_j^P) \\ &= \{e_{ik_1}^T, e_{k_1k_2}^T, e_{k_2k_3}^T, \dots, e_{k_nj}^T\} \end{aligned} \quad (4.1)$$

其中， $v_i^P, v_j^P$  是区域的两个通道节点。注意，它们不一定是 Topology Graph 的节点，因为在之前进行区域合并的步骤中，它们可能是因为跨越区域边界而被切割生成的通道，即 Topology Graph 的边与区域边界的交点，这种情况下生成的通道节点是 Topology 边途经的一个点。在另一种情况中，一个通道顶点不是通过切割生成的，而是本来就属于 Topology Graph 的 junction 顶点，则该通道节点就是 Topology Graph 的节点。但不管哪一种情况，Passage Graph 的通道节点之间的路径都可以由 Topology Graph 的一系列边构成。

对所有区域中的每对通道运行基于  $A^*$  的通道间寻径过程，然后保存这些通道间的路径。这是在离线预计算步骤中完成的。这样我们可以大大降低在线规划算法的计算量。图4.1c对生成的通道图进行了可视化，可见作为走廊的区域由于连接多个房间因此该区域会具有多个通道。则通道图中属于该区域的通道节点会在此区域中生成多条边。而只有一个通道的区域自然没有生成任何通道图的边，这类区域常常是仅与走廊相连的、仅有一个门的房间。图4.1c是典型的“房间-走廊”建筑结构，该地图中的大部分边缘都在代表走廊的大中心区域内。

在线上规划的步骤中，用户给定两个点分别作为起点和目标点。算法首先检

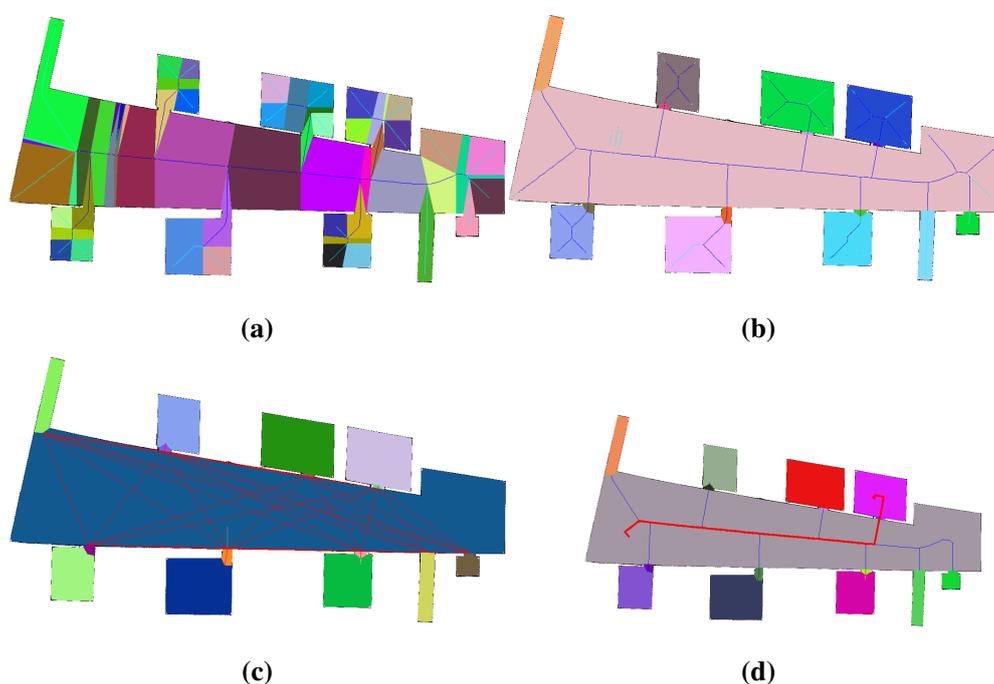


图 4.1 以图3.4的地图为例，展示生成 Passage Graph 的过程的示意图。(a) 根据 Topology Graph 的边生成的多边形。(b) 由 Alpha Shape 检测到的同一房间内的区域已被合并。(c) 通道图的边显示为红线。从图中可以看到深蓝色的大中心区域的所有通道之间是如何相互连接的，通道图的每条边都关联一条从 Topology Graph 的边获得的路径并记录路径的长度。(d) 图中的红色加粗线条展示了基于拓扑表示的路径规划算法搜索到的从起始点到终点的一条路径，显然它是通过将起始点和终点连接到线下保存的拓扑骨架路径上形成的。

Figure 4.1 Illustrations for the steps of the Passage Graph generation on the example map from Figure 3.4. (a) The polygons generated from the Topology Graph edges. (b) Areas within the same room, which detected by Alpha Shape, have been merged. (c) The edges of the Passage Graph are visualized as red lines. One can see how all of the passages of the big blue center area are connected to each other. Each of the edges is attributed with a path, obtained from the Topology Graph edge, and the length of the path. (d) The red bold line in the figure shows a path from the start point to the end point searched by the topological map-based path planning algorithm. The path is obviously formed by connecting the start point and the end point to the offline-saved topological skeleton path.

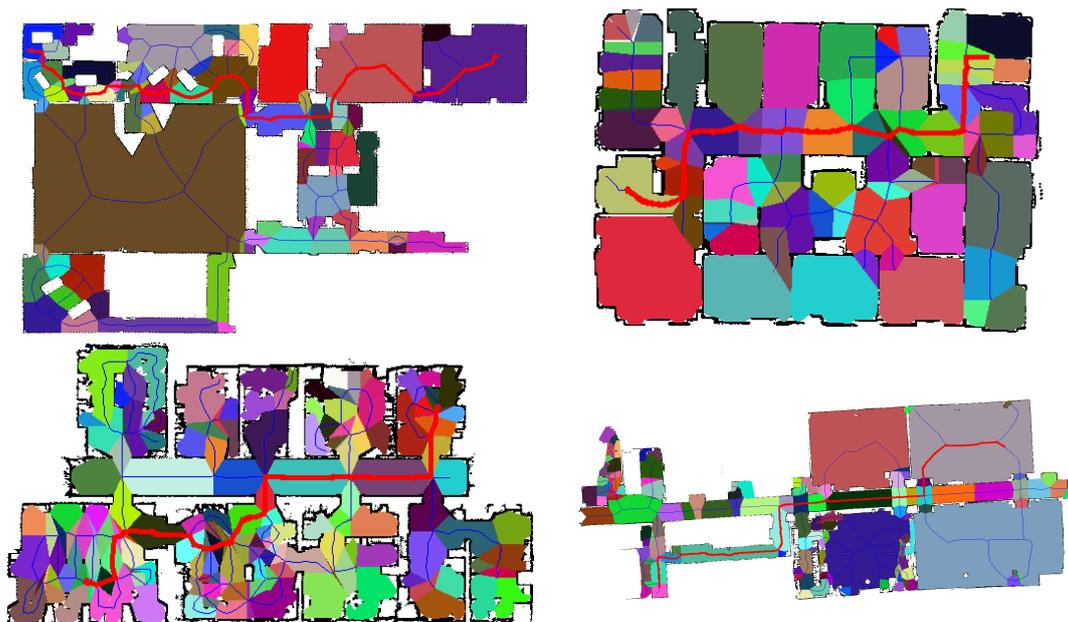


图 4.2 基于 Area Graph 拓扑表示构建的 Passage Graph 进行全局路径规划。随机给定起点与终点，由我们的算法获得的路径显示为加粗的红色线条。

Figure 4.2 Perform global path planning based on the Passage Graph, which is constructed from the Area Graph. Given a random pair of start and end point, the path searched by our algorithm is shown as a bold red curve.

查每个点所在的区域。在起点和目标点所在的区域，将分别添加一个新的虚拟通道点  $v_s^p, v_e^p \in V^p$ 。新建的虚拟通道节点将与该区域的所有通道节点之间都在 Passage Graph 中添加新的边，以此来与该区域的所有通道形成连接关系。紧接着，算法为这些新边创建路径。为此，将新的虚拟通道节点与该区域最近的骨架路径上的与该节点最近的点通过直线段连接起来。一旦新的虚拟通道连接到骨架，就可以使用与线下的路径搜索算法相同的  $A^*$  实现来计算新节点到该区域其他节点的路径。

最后，我们可以在 Passage Graph 上应用  $A^*$  算法通过搜索串联起点到终点的通道节点序列来找到从起点到目标的机器人可行路径。一些全局路径搜索结果示例如图 4.2 所示。

#### 4.2 效率展示实验

为了比较在网格地图上的规划和在通道图上的规划效率，本章进行了计算时间长短的比较实验，将传统  $A^*$  算法在基于网格地图的的规划时间与在通道图

的规划时间进行比较，并且比较算法在大地图中规划的路径长度。

本实验使用的大地图是从本校教学楼扫描获得的 3D 点云压缩得到的 2D 扫描平面图。我们在本实验中使用的地图经过一些人工编辑：清除了一些噪点和障碍物，并且移除了关闭的门，并在玻璃隔板（因为激光会穿越玻璃，激光雷达无法将玻璃扫描为障碍物）和未扫描到的地方添加墙壁。这张地图是一张单单元格数量为  $2000 \times 1500$  的图像，并设置分辨率为每个网格单元代表环境中的  $0.1m \times 0.1m$ 。

对于本次实验使用的地图，由于建筑物内部面积较大（由上述数据可算得该建筑物的外框尺寸为  $200m \times 150m$ ），房间、开阔空间等区域较多，从网格地图中提取 Passage Graph 所需的初始化时间为 11.8 秒。在 Passage Graph 提取之后，我们给定多对起点与终点，使用相同的 Passage Graph 对不同的起点和目标点进行路径搜索。作者使用本文实现的基于网格图的  $A^*$  规划器替换机器人操作系统 ROS 的导航库 move\_base 的 global\_planner Lu (2022)，使该算法可在真实机器人系统中被机器人用于规划和导航应用。为了公平起见，实验只记录两种算法的线上路径搜索时间，不包括其他处理时间。

网格地图在路径搜索中也可看作度量图 (Metric Graph)，以单元格为节点进行搜索。基于网格地图的  $A^*$  算法要搜索的节点维度取决于像素数，在这张图中超过  $3 * 10^6$  个。而基于 Passage Graph 的  $A^*$  算法的搜索空间取决于区域的数量，在这张地图中为 245。换句话说，基于拓扑表示的路径规划将计算复杂度降低了 4 个数量级。

实验分别设定多对起始点与目标点，运行基于网格地图的  $A^*$  算法和基于 Passage Graph 的  $A^*$  算法进行路径搜索，将搜索的计算时间与获得的路径长度列于表 4.1 中。如表所示，当起点和目标在同一个房间时，我们的算法在时间上不能表现出巨大的优势，而且本文算法的路径距离大约是  $A^*$  的 1.2 倍。原因是，在这种情况下，路径非常短且简单。网格图上的  $A^*$  无需绕过许多障碍即可找到路径，而本文的算法首先必须到达最近的 Topology Graph 边上的点。

令人惊奇的是，当起点和终点在不同房间时，基于网格地图的  $A^*$  运算时间随着路径长度的增加而显著增加，而我们的基于拓扑图的规划的时间仅根据区域结构的复杂性的变化有轻微的增长。

关于路径长度，我们可以看到本算法得到的路径总是比网格图上  $A^*$  得到的

表 4.1 不同路径距离下基于网格地图的  $A^*$  与基于通道图的  $A^*$  的规划时间比较。Table 4.1 Comparison of planning time for  $A^*$  on grid map with  $A^*$  on Passage Graph under different path distance.

Path Distance (m)		Planning Time (ms)		Remark
Grid	Passage	Grid	Passage	Path in Passage Graph
17.53	21.53	1.706	0.212	In same middle room
56.81	67.44	5.497	0.456	In same big room
25.87	33.84	11.834	0.263	Cross 4 rooms
48.65	49.87	14.079	0.494	Cross 7 rooms
130.07	137.00	48.979	1.292	Cross 11 rooms
102.17	112.16	88.726	0.784	Cross 17 rooms
210.48	235.75	127.758	1.772	Cross 23 rooms

路径长。当路径长度较短时，我们的算法得到的路径最高可达网格图上  $A^*$  得到的路径长度的 1.3 倍（在第三组结果中）。幸运的是，这个比率随着路径长度的增加而降低。在最后两组数据中，可以看到这个比率下降到 1.1 倍左右。但是，这有它的好处。基于网格地图的  $A^*$  算法倾向于产生贴着墙壁的路径，这会导致机器人不得不在避障算法上做更多的努力，由此局部路径规划器为了避障产生的实际机器人轨迹也会与全局路径规划产生的轨迹相去较远，这为跟踪全局路径增加了困难。而基于 Passage Graph 的路径是基于维诺图的修剪版本产生的，在维诺图的修剪过程中已移除了靠近障碍物的边，由此产生的路径则会远离障碍物，几乎都在区域的中间，这样的路径虽然长度上不占优势，但是对避障算法更加友好，由避障算法获得的机器人的实际行进路径也能更好地跟踪全局路径。并且我们发现本算法选择的路径与人类在真实环境中行走选择的路径更相似。为了进一步缩减路径长度，在未来的工作中，作者将考虑在离线预计算步骤中对区域中通道之间的路径进行进一步优化，这将获得兼顾路径长度和避障友好的路径规划结果。

表4.1中最后一组起始点与终点之间两种算法找到的路径如图 4.3 所示。

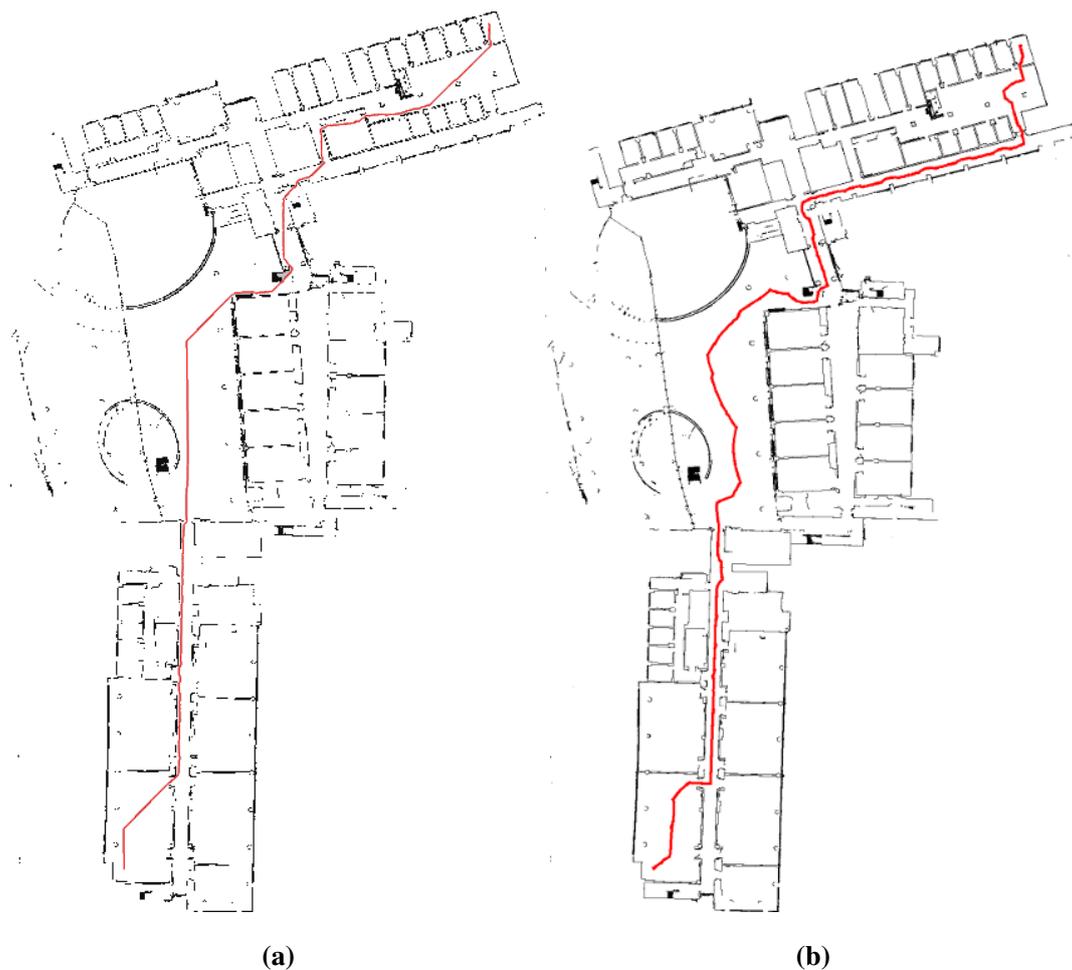


图 4.3 比较在栅格地图和通道图上的  $A^*$  路径规划算法。(a) 在栅格地图上的  $A^*$  算法搜索结果。(b) 在 Passage Graph 上的  $A^*$  搜索结果。

Figure 4.3 Compare  $A^*$  path planning algorithm on grid map and Passage Graph. (a) Result of  $A^*$  algorithm on grid map. (b) Result of  $A^*$  algorithm on Passage Graph.

### 4.3 本章小结

本章介绍的基于区域图的路径规划是对区域图的一个应用实例。作者首先通过构建区域图的对偶图 Passage Graph 来预存区域中的路径。随后在线上使用  $A^*$  搜索算法，通过结合对起终点区域进行基于单元格的路径规划，和对中间区域使用离线预存路径进行路径搜索，使基于拓扑地图的全局路径规划算法的计算时间比基于栅格地图的路径规划时间缩短了两个数量级。

## 第 5 章 基于区域图的地图匹配

### 5.1 算法简介

地图构建中的一项重要任务是匹配两个地图，这些地图可能来自单个或多个机器人、来自机器人地图或设计平面图 CAD 绘图。匹配多个地图的需求可以是来自地图匹配、地图融合（地图合并）(Huang 等, 2005)、机器人定位 (Badino 等, 2012) 或地图评估 (Schwertfeger 等, 2015)。本文提出了两种地图匹配算法——启发式聚类法 (*Hypotheses Clustering*) 和生长邻居法 (*Neighbor Growing*)，可以根据从二维地图生成的拓扑表示 Area Graph 有效地匹配两个 2D 栅格地图，并在接下来的多个实验中展示这两种匹配方法的性能。图 1.2 展示了区域图的生成和通过其进行地图匹配的流程。图 5.1 显示了使用两种建议方法匹配两幅地图的示例。这些地图匹配的方法已在会议论文 Hou 等 (2019a) 和期刊论文 Hou 等 (2022) 中发表。

#### 5.1.1 算法流程

给定两张地图，其中有对同一环境的部分场景重叠，对这两张地图进行匹配的算法工作流程如下：

1. 将给定的 2D 地图分割成区域，本文生成前文提到区域图作为地图分割的解决方案。
2. 从地图分割的区域中提取特征。用于匹配的特征包括区域的面积大小、通道间距离和凸包最长距离等。
3. 通过将特征距离乘以它们的权重来计算每对区域之间的匹配成本，并将区域匹配成本记录在加权邻接矩阵  $M_{cost}$  中。
4. 本文提出了两种方法可以匹配两个给定的地图：启发式聚类法和生长邻居法。
5. 启发式聚类法 (*Hypotheses clustering*):
  - (a) 对于匹配成本低于特定阈值的区域对，可被视为潜在的匹配区域，我们使用 Kuhn-Munkres (KM) 算法 (Kuhn, 1955; Munkres, 1957) 对这两个区域的直接邻居进行二分匹配，并将匹配成本低于阈值的邻居对的匹配成本乘以  $M_{cost}$  中原来的成本值（即原来两个区域的匹配成本）作为新成本。

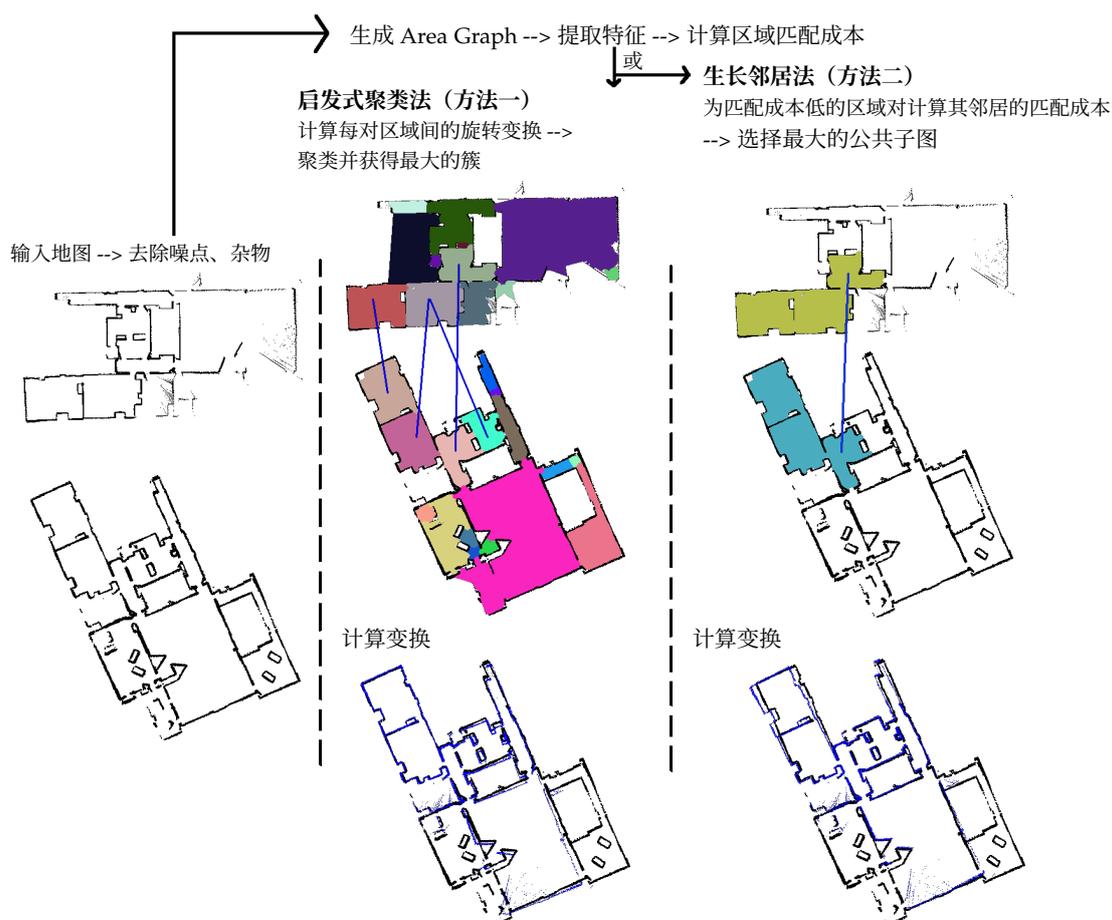


图 5.1 输入是由两个机器人分别对同一环境的扫描地图，两个地图之间有  $43.73^\circ$  的相对旋转。第二行是由两个栅格地图生成的区域图。对于我们提出的启发式聚类方法，蓝线显示由我们的算法自动进行的区域匹配。该方法由该匹配结果估计得旋转变换为  $43.99^\circ$ 。对于我们提出的生长邻居法，蓝线显示最大匹配子图中的区域对，由此估计出的旋转变换为  $44.74^\circ$ 。对于有损坏的地图，我们不计算地图间的变换。可以看见使用两者中的任何一个方法，两个粒子图都可以被很好地合并为一个完整的地图。

Figure 5.1 Two maps of an environment partially scanned by two robots, respectively, with  $43.73^\circ$  rotation, are the inputs. In the second column, we can see the Area Graphs generated for the two maps. For our Hypotheses Clustering method, the blue lines show the areas that were automatically matched by our algorithm. The method estimated rotation of  $43.99^\circ$  for the matching. For our Neighbor Growing method, the blue lines show the area pairs in the biggest matched subgraph, from which the method estimated rotation of  $44.74^\circ$  for the matching. For the broken-map matching, we don't calculate the map transformations. We can see that the two partial maps can be well merged into a complete map, with either of the two methods.

(b) 查找相互匹配: 为每个区域在另一张地图中找到其  $k$  个最佳的对应区域。如果两个区域是彼此的  $k$  个最佳匹配之一, 则认为它们彼此匹配并将它们保存在列表 *best\_matches* 中。

(c) 对于匹配成功的区域对, 估计它们之间的旋转并将角度保存为潜在的地图间旋转变换, 或成为一个旋转假设 (Rotation Hypotheses)。

(d) 对旋转假设进行聚类 (相近的值聚集为一簇), 以投票选出两个地图之间正确旋转所在的范围。包含最多旋转假设的一个簇称为最佳簇 (*best cluster*)。

(e) 在最佳簇中遍历样本, 估计匹配对之间的旋转, 并计算将旋转应用于两个地图时的重叠大小以计算地图间的最佳变换。

#### 6. 生长邻居法 (Neighbor growing):

(a) 查找相互匹配的区域对并保存在列表 *best\_matches* 中 (与上述 5(b) 相同的步骤)。

(b) 以 *best\_matches* 列表中的区域对作为根节点使用深度优先搜索, 通过匹配成本低于阈值的邻居匹配来增长匹配子图。

下面我们详细介绍这些模块。我们的两种匹配方法的匹配流程示例如图 1.2 所示。

## 5.2 算法的具体实现

### 5.2.1 区域特征的提取与成本计算

我们的地图匹配方法是基于区域匹配的。因此, 首先, 我们用面积图生成算法分割一对给定的地图。对于面积图中的每个区域, 我们计算其面积大小 (*area size*)、区域周长 (*perimeter*)、通道距离 (*passage distance*)、通道数 (*passage number*)、外围凸多边形面积大小 (*convex hull area size*)、外围凸包的周长 (*convex hull perimeter*)、外围凸包的最长线段 (*convex hull longest distance*)、面积周长比 (*area circumference ratio*) 和 *Hu* 矩 (Huang 等, 2010) 作为候选特征, 分别计算两个区域的对应特征的差距作为特征距离。迭代最近点 (*Iterative Closest Point, ICP*) 误差也作为最特征距离并列的两个区域间的匹配成本。特征差距越小即匹配成本越小, 则说明两个区域更可能是匹配的, 可以根据区域的匹配成本判断区域的匹配程度。在本节中, 将介绍这些特征及其距离的计算。它们在区域匹配中的影响将在章节 5.3.2 中进行分析, 其中一些特征将被选择用于最终的区域匹配。这些特征的详细信息

将在以下部分中进行描述。

### 5.2.1.1 区域面积与周长

地图中的区域是由有界多边形进行几何表示的。它的面积大小和周长是最基本的且具有意义的几何特性。要计算这两个几何特征，我们的算法需要知道两张地图的尺度比例（分辨率）。值得注意的是，由于用于面积大小的单位是  $m^2$ ，而其他要素（例如周长）的单位是  $m$ ，区域匹配总成本是所有特征的特性和，为了保持特性和的线性，区域的面积大小的特征差距  $c^a$  更合理的计算方法应当是对面积进行开方再做差，归一化时亦是除以面积的开方。而周长的特征差距只需用两个区域的周长值直接做差，再用较大的周长值进行归一化。因此，区域大小和周长的特征差距计算为：

$$c_{ij}^a = \frac{|\sqrt{a_i} - \sqrt{a'_j}|}{\sqrt{\max(a_i, a'_j)}}, \quad c_{ij}^p = \frac{|p_i - p'_j|}{\max(p_i, p'_j)}. \quad (5.1)$$

$c_{ij}^a$  是第一张地图中第  $i$  个区域的面积大小与第二张地图中第  $j$  个区域的面积大小的特征差距； $c_{ij}^p$  是第一张地图中第  $i$  个区域的周长与第二张地图中第  $j$  个区域的周长之间的特征差距。其中  $a_i$  和  $a'_j$  分别是第一张地图中第  $i$  个区域和第二张地图中第  $j$  个区域的面积大小； $p_i$  和  $p'_j$  分别是第一张地图中第  $i$  个区域和第二张地图中第  $j$  个区域的周长。

### 5.2.1.2 凸多边形的面积与周长

为了计算区域的外围凸多边形（凸包），本算法使用了开源库 CGAL 的 2D Convex Hulls 类 (Hert 等, 2022)。然后计算每个凸包的区域大小和周长作为匹配区域的特征。使用凸包的几何特征而不是区域多边形的主要优点是，通过忽略未合并到房间中的缺失区域，过度分割的房间仍然有可能与其在另一张地图中正确分割的房间相匹配。凸包面积和周长的距离的计算方法与 5.2.1.1 部分中的  $c_{ij}^a$  和  $c_{ij}^p$  相同。

### 5.2.1.3 外围凸包的面积周长比

区域的形状可能是区域匹配的有用信息。面积值的平方根与周长的比值可以作为一个反映凸包与圆的接近程度的特征。圆的面积值为  $\pi R^2$ ，周长为  $2\pi R$ 。定义面积周长比可以用来测量多边形的有多接近圆形，利用它的面积和周长

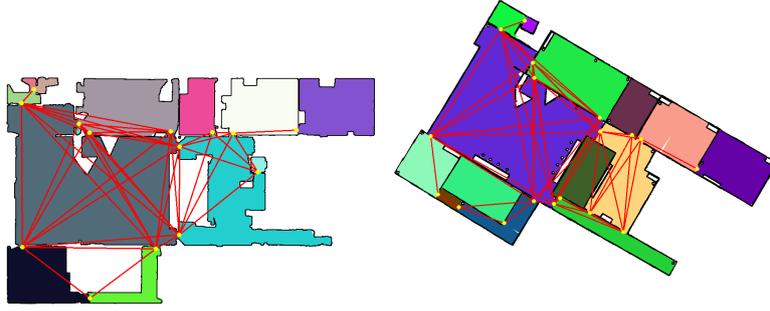


图 5.2 机器人生成的地图（左）与相应的平面图（右）相匹配。地图分为不同的区域，用不同的颜色表示。通道点以黄色圆点突出显示，我们将属于同一区域的通道用红线连接起来，通道的距离特征是根据红线的长度计算的。

Figure 5.2 A robot-generated map (left) to be matched with the corresponding floor plan (right). The maps are divided into different areas, which are represented in different colors. The passage points are highlighted as yellow points, and we connect the passages belonging to the same area with red lines. The passage distance features are computed from the lengths of the red lines.

可进行计算：

$$r^{ap} = \frac{\sqrt{Area/\pi}}{\frac{1}{2}Perimeter/\pi} = \frac{\sqrt{R^2}}{R} = 1. \quad (5.2)$$

对于一个圆形，其面积周长比为

$$\frac{\sqrt{Area/\pi}}{\frac{1}{2}Perimeter/\pi} = \frac{\sqrt{R^2}}{R} = 1. \quad (5.3)$$

因此，对于对应于区域  $x$  的凸多边形，其面积周长比  $r_x^{ap}$  越接近 1，该凸多边形越圆。我们将此特征差距计算为

$$c_{ij}^r = \frac{|r_i^{ap} - r_j^{ap}|}{\max(r_i^{ap}, r_j^{ap})}. \quad (5.4)$$

#### 5.2.1.4 通道特征与区域最长线段

我们注意到通道通常是门和走廊的交界处，是固定的。自然，通道之间的距离也是固定的。因此，本文提出了将区域中的通道数和通道之间的距离作为特征来匹配区域的想法。在图 5.2 中，同一区域的所有通道都用红线连接，并计算每个区域中所有通道之间的距离。区域  $i$  中的通道距离与区域  $j$  中的通道距离之间的最小差异计算为匹配成本  $c_p d^{ij}$ ：

$$c_{ij}^{pd} = \frac{\min_{n,m} |pd_{i_n} - pd'_{j_m}|}{\max(pd_{i_n}, pd'_{j_m})}. \quad (5.5)$$

在上面的式子中，一对通道之间的距离被记为  $pd_{i_n}$ ，其中  $i$  是区域 ID，这是区域  $i$  中的第  $n$  个通道距离。例如， $N$  个区域  $i$  中的通道距离可描述为  $\{pd_{i_1}, \dots, pd_{i_N}\}$ 。

通道数是指一个区域中通道的数量（即区域图中的边）。通道数的特征差距（两个区域中通道数的差）计算为

$$c_{ij}^{pn} = |pn_i - pn'_j|, \quad (5.6)$$

其中  $pn_i$  是区域  $i$  的通道数。该特征的稳定性取决于分割质量和地图的完整性。如果区域被过度分割或区域不完整，则通道数可能不是环境中的真实通道数。

有些区域只包含一个通道。这种情况下则无法获得这些区域的通道距离作为特征。我们观察到两个对应区域的最长线段长度会非常接近，后面的实验显示，以此为特征甚至优于通道距离的准确性。但是，由于该特征会受到地图扫描完整性的影响，因此算法仍将通道距离作为匹配特征，仅当该区域没有多个通道以计算通道距离特征时才计算其最长距离。因为区域多边形中最长的线段与其凸包中的相同，并且在后者中找到最长的线更快，所以我们从该区域的凸包中计算最长的线段。两个区域之间的最长线段距离计算为

$$c_{ij}^l = \frac{|l_i - l'_j|}{\max(l_i, l'_j)}, \quad (5.7)$$

其中  $l_i$  是区域  $i$  的最长线段的长度。

除了用于区域匹配成本的计算，通道距离和凸包最长线段将是接下来进行区域对齐计算的重要工具。

#### 5.2.1.5 Iterative Closest Point (ICP) 的误差值

迭代最近点 (Iterative Closest Point, ICP) 是一种广泛使用的点集配准算法，可以用于对齐点云并计算点云之间的变换。ICP 交替执行两个步骤：匹配点云和求解变换。它能够对齐具有相似结构的两个点集。作为变换计算的副产品（例如 [Horn \(1987\)](#)），ICP 还返回匹配点集对齐的剩余误差。算法使用这个 ICP 误差来估计两个区域的形状匹配程度。

由于每个区域都有自己的区域多边形，可以通过轮廓来描述。因此，通过用一组点来对轮廓进行离散化，就可以利用 ICP 来评估相似度。本算法实现使用 20cm 的步长对区域多边形上的点进行采样（点太密集会使计算量过大）。本方法使用开源库 Point Cloud Library (PCL) ([Rusu 等, 2011b](#)) 来实现 ICP 计算。对于地

图，每个被占用的点都被加入点集。当 ICP 收敛时，欧几里得适应度分数  $c_{ij}^{ICP}$  被用作匹配距离，它是点集  $P$  和  $Q$  之间的平方距离的平均值。给定对应的点集  $C$ ，该特征差距计算为

$$c_{ij}^{ICP} = \frac{1}{n} \sum_{c \in C} (Rp_{c_p} + t - q_{c_q})^2 \quad (5.8)$$

其中  $p \in P, q \in Q$ .

ICP 对姿态的初始猜测很敏感，可能会陷入局部最小值。为了缓解这个问题，可使用 12 个不同的偏航角作为地图（点集）间的初始旋转猜测来运行 ICP 算法：从 30 度到 360 度，步长为 30 度。这样，两个点集的真实旋转与最佳初始猜测之间的最大误差仅为 15 度。算法使用 12 次 ICP 运行中的最小误差作为特征差距。

### 5.2.1.6 Hu 矩

Hu 矩 (Hu, 1962; Huang 等, 2010) 是一种广为使用的形状比较方法，以其对图像平移、缩放和旋转的不变性而闻名。Hu 矩是使用形状的中心矩计算的。本算法使用两个 OpenCV 函数：*moments* 用于计算中心矩，然后函数 *HuMoments* 用于计算区域的 Hu 矩，作为匹配特征。形状的 Hu 矩是一组七个值。对于两个相似的形状，它们的七个 Hu 矩值将非常接近。两个区域之间的 Hu 矩距离计算为

$$c_{ij}^{Hu} = \sum_{n=1}^7 |h_{in} - h_{jn}| \quad (5.9)$$

其中  $h_{in}$  是区域  $i$  的第  $n$  个 Hu 矩值。如果两个区域轮廓接近，则该特征差距  $c_{ij}^{Hu}$  会很小。

## 5.2.2 区域间的匹配成本

在前面的章节中已经描述了区域特征的细节。在进入实际匹配算法之前，作者简要描述所有区域之间的匹配成本。

将地图  $A$  中的每个区域与地图  $B$  中的所有  $m$  个区域进行比较。对于每一次比较，例如，当对地图  $A$  中的区域  $i$  与地图  $B$  中的区域  $j$  进行比较，首先计算所选特征的特征差距并将它们保存在行向量中。然后通过将这对区域的特征差距向量和权重向量  $W_f$  相乘作为所有特征成本的加权和来计算该区域对之间的总匹配成本。地图  $A$  的区域  $i$  与另一张地图  $B$  中所有  $m$  个区域之间的匹配成本记

录在向量  $C_i$  中:

$$C_i = \begin{bmatrix} c_{11} & c_{12} & c_{13} & \cdots & c_{1F} \\ c_{21} & c_{22} & c_{23} & \cdots & c_{2F} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ c_{m1} & c_{m2} & c_{m3} & \cdots & c_{mF} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_F \end{bmatrix} = \begin{bmatrix} C_{i1} \\ C_{i2} \\ \vdots \\ C_{im} \end{bmatrix} \quad (5.10)$$

其中  $c_{jf}$  是区域  $i$  和另一张地图  $B$  中的区域  $j$  之间的第  $f$  个特征距离 (每对区域要比较的  $F$  个特征)。  $C_{ij}$  是第一张地图中的第  $i$  个区域和第二张地图中的第  $j$  个区域之间的匹配成本。加权邻接矩阵  $M_{cost}$  保存了两幅给定地图中所有区域对之间的匹配成本:

$$M_{cost} = \begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1m} \\ C_{21} & C_{22} & \cdots & C_{2m} \\ \vdots & \vdots & \cdots & \vdots \\ C_{n1} & C_{n2} & \cdots & C_{nm} \end{bmatrix} \quad (5.11)$$

在章节5.3.2 和章节5.3.3中, 作者将讨论特征的选择和权重向量的设定。下一节将介绍启发式聚类法, 而5.2.4节将描述生长邻居法。

### 5.2.3 方法一: 启发式聚类法

#### 5.2.3.1 过滤差匹配

为了找到潜在的匹配良好的区域对, 仅选择特征成本低于阈值的区域对记录在 *matches\_list* 中, 以便在下一步中计算它们的邻居的相似度。

#### 5.2.3.2 匹配邻居区域

加入邻居匹配信息是基于拓扑结构的地图匹配的一个优点。为了使用邻居信息, 本文的匹配算法不仅检查单对区域的相似性, 而且还考虑了它们的邻居的匹配成本。

Kuhn-Munkres (KM) 算法 (Kuhn, 1955; Munkres, 1957) 是一种可以以最小成本匹配二分图的算法。区域  $A_i$  和  $B_j$  的邻居是两个不相交的集合  $AN$  和  $BN$ , 恰好形成一个二分图。对 *matches\_list* 中的一对区域  $(A_i, B_j)$  使用 KM 算法计算其邻居的最佳匹配。然后将  $(A_i, B_j)$  的匹配成本与其所有邻居的匹配成本相乘作为新的匹配成本。根据直觉可理解, 当成功匹配的邻居对越多, 或是邻居对的匹配

**算法 4** 寻找最佳的邻居匹配排列**算法 4** Find the Best Neighbor Matching Permutation

---

**function** BESTNEIGHBORS( $A_i, B_j, M_{cost}$ )
**Require:** Area  $i$  in map A as  $A_i$ , area  $j$  in map B as  $B_j, M_{cost}$ **Ensure:**  $MchAN, MchBN, M'_{cost}(i, j)$ 

```

2:    $AN = \{an_1, \dots, an_{n_a}\} \leftarrow A_i.neighbors()$ 
       $BN = \{bn_1, \dots, bn_{n_b}\} \leftarrow B_j.neighbors()$ 
4:    $short\_size \leftarrow \min(AN.size(), BN.size())$ 
       $MchAN, MchBN \leftarrow KM(AN, BN)$ 
6:    $M'_{cost}(i, j) \leftarrow M_{cost}(i, j)$ 
      for  $i = 0$  to  $short\_size$  do
8:        $M'_{cost}(i, j)* = M_{cost}(MchAN[i], MchBN[i])$ 
      end for
10: end function

```

---

成本越小，则该区域新的总匹配成本越小。该方法在算法 4 中进行了描述，其中  $MchAN$  和  $MchBN$  分别是区域  $A_i$  和  $B_j$  的一对一匹配的邻居。然后我们得到一个新的成本矩阵  $M'_{cost}$ ，它记录了区域对之间的新匹配成本，其计算包含了邻居的匹配成本。

## 5.2.3.3 寻找相互匹配的区域

对于地图 A 中的每个区域，我们记录在地图 B 中与其最相似的  $k$  个（例如  $k=3$ ）区域，即与其匹配成本最低的  $k$  个区域；反之，对地图 B 中的每个区域亦然。区域  $i$  和  $j$  被认为是相互匹配的区域当且仅当区域  $i$  是区域  $j$  的  $k$  个最近匹配之一并且区域  $j$  也是区域  $i$  的  $k$  个最近匹配之一。将所有相互匹配的区域对记录在匹配对列表  $best\_matches$  中，用于生成地图间的旋转假设。

## 5.2.3.4 匹配的区域对之间的旋转变换

给定一对匹配区域  $(A_i, B_j)$ ， $A_i$  和  $B_j$  之间的变换是根据它们中匹配的通道间线段或区域中的最长线段来估计获得的。这个过程如图 5.3 所示。旋转变换通过两个线段之间的角度差来计算，记为  $\alpha_{ij}^r$ ，其中线段  $p_n^i$  和  $p_m^j$  的中点分别作为两个区域的旋转中心，地图 A 中的区域  $A_i$  将通过旋转中心与区域  $B_j$  对齐，并

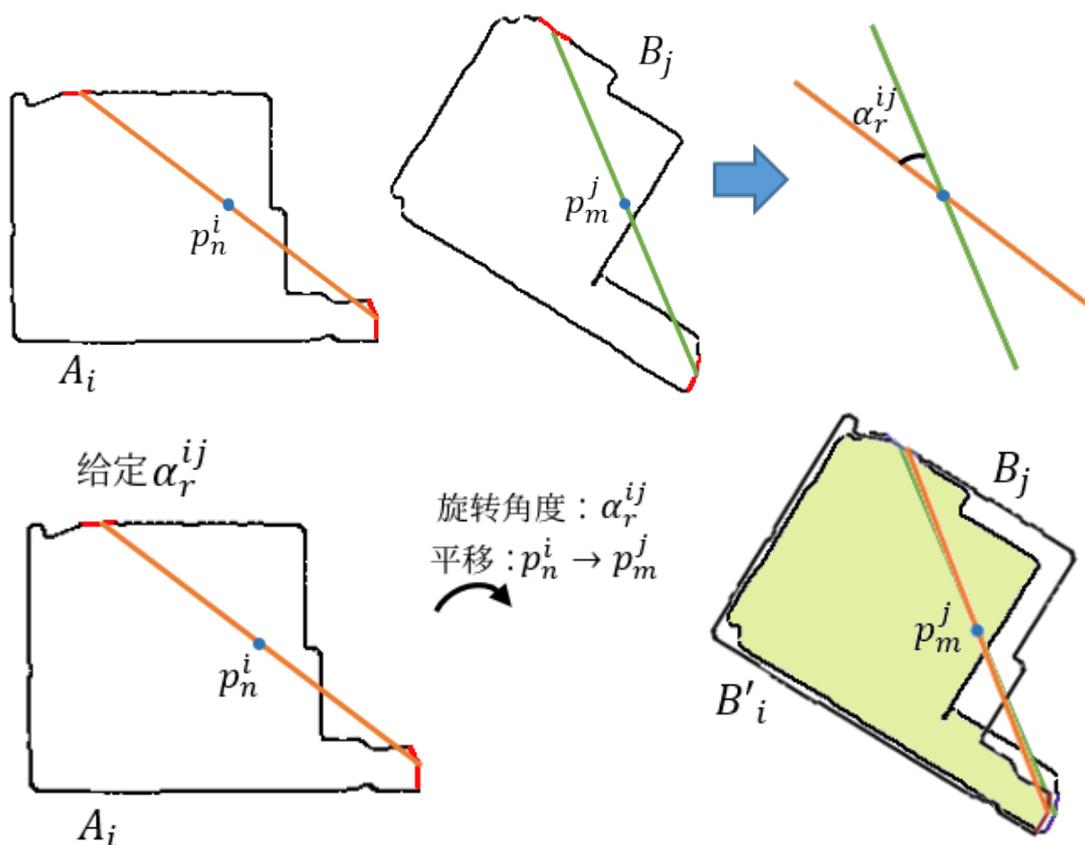


图 5.3 (a) 计算两个区域之间的变换。(b) 将具有给定角度的区域  $A_i$  及其匹配段的中点转换为匹配区域  $B_j$ 。段落以红色突出显示。

Figure 5.3 (a) Compute the transformation between two areas. (b) Transform the area  $A_i$  with the given angle and its matched segment's midpoints to its matched area  $B_j$ . The passages are highlighted in red.

旋转角度  $\alpha_r^{ij}$  来变换到地图  $B$  中成为区域  $B'_i$ 。这里线段  $p_n^i$  表示区域  $i$  的通道之间的连接线段中与区域  $j$  中的通道连接线段  $p_m^j$  相互匹配成功的一条 ID 为  $n$  的线段；或者，当区域  $i$  仅有一个通道， $p_n^i$  是区域  $i$  中的最长线段。然后我们计算从区域  $A_i$  变换过来的区域  $B'_i$  和  $B_j$  之间的重叠，并将其作为判断两个区域是否匹配正确的标准。然后将正确匹配区域之间的旋转角度记录在列表中作为可能正确的旋转变换，称为旋转假设。

### 5.2.3.5 对旋转假设进行聚类

这一节对根据不同匹配区域对估计出的旋转假设进行聚类，以确定两个地图之间的旋转。聚类算法基于上一步中确定的两个地图的两个匹配区域之间的旋转，通过以下方式将相近的结果聚集在一起。

给定匹配区域的列表，聚类过程首先将列表中的第一个旋转设置为新聚类的中心。给定列表中的另一个旋转，如果它到所有簇中心的距离均大于设定的角度阈值（例如， $3^\circ$ ），我们创建一个以它为中心的新簇，否则，将它添加到中心值与它最接近的簇中，并重新计算该簇的中心。聚类完成后，具有最多旋转假设的聚类，记为  $Cluster_{max}$ ，将被认为是包含正确地图旋转变换的聚类。

### 5.2.3.6 搜索最佳的旋转估计

给定旋转假设聚类获得的最佳簇  $Cluster_{max}$ ，遍历它的样本：对于每个样本  $(\alpha_{ij}, (p_n^i, p_m^j))$ ，将最佳聚类中的所有属于地图 A 的区域变换到地图 B 上，使其与其匹配的地图 B 的对应区域重叠。然后计算它们之间的重叠百分比，并计算最佳簇中的所有匹配对的重叠百分比的平均值。其中变换由角度  $\alpha_{ij}$  作为旋转，平移由  $p_n^i$  和  $p_m^j$  的中点之间的平移给出。对于一对匹配区域  $(A_i, B_j)$ ，其面积值为  $a_i$  和  $a_j$ ，它们的重叠面积百分比  $OP$  由下式计算

$$OP = \frac{Overlap\_Area}{\min(a_i, a_j)}. \quad (5.12)$$

然后选择获得最高重叠平均值的样本作为地图之间的最终变换估计。

### 5.2.4 方法二：生长邻居法

在建图过程中，由于传感器漂移或定位错误等原因，我们可能会获得损坏的地图。在这种情况下，从损坏的地图的不同部分到设计平面图的变换矩阵可能是不一致的。作者提出通过使用邻居生长匹配法来匹配地图的不同部分（作为子图匹配）来处理这种情况。其思路是，通过区域的特征匹配算法寻找成功互相匹配的区域来完成子图匹配。该算法中的“子图”是指区域图 Area Graph 的公共子图，子图中的每个节点代表地图中的一个区域。子图生成方法是通过“生长邻居节点”：若两幅地图中的两个节点成功匹配上了，则将它们各自加入子图，接下来尝试匹配它们的邻居，若其邻居节点成功相互匹配，则将其邻居节点也加入子图。如此一步步通过邻居将互相匹配且相互连接的区域加入子图中。最终分别为地图 A 和地图 B 生成区域图子图，这两个子图中的节点是一一相互匹配的关系，则这个子图可视为由两幅地图创建的两个区域图的公共子图。

算法 5 生长邻居法

算法 5 Neighbor Growing

---

```

function NEIGHBORGROWING(best_matches,  $M_{cost}$ )
2:   while best_matches is not empty do
       pairinit  $\leftarrow$  best_matches.min()
4:     best_matches.remove(pairinit)
       close_list = {}, open_list = {}
6:     if close_list.find(pairinit) then
           continue
8:     end if
       open_list.insert(pairinit)
10:    while open_list is not empty do
           ( $A_i, B_j$ )  $\leftarrow$  open_list.min()
12:      open_list.remove(( $A_i, B_j$ ))
           close_list.insert(( $A_i, B_j$ ))
14:       $MchAN, MchBN \leftarrow$  BESTNEIGHBORS( $A_i, B_j, M_{cost}$ )
           for  $i = [0, MchAN.size()$ ] do
16:             if (NOT close_list.find(( $MchAN[i], MchBN[i]$ ))) AND (
                $M_{cost}(MchAN[i], MchBN[i]) < \text{Threshold}$ ) then
                   open_list.insert(( $MchAN[i], MchBN[i]$ ))
18:             subgraph[pairinit]  $\leftarrow$  ( $MchAN[i], MchBN[i]$ )
                   end if
20:             end for
           end while
22:     end while
end function

```

---

5.2.4.1 初始设置

在对子图进行生长之前，首先选择最有可能是正确匹配的区域对作为初始节点，以此来生长它们的邻居。在本算法中，使用章节5.2.3.3描述的算法来完成寻找潜在的成功匹配区域对的步骤。即，算法通过将相互之间的特征匹配成本最

低的区域对选择为潜在的匹配对，并将它们保存在 *best\_matches* 作为子图的初始增长节点。

#### 5.2.4.2 通过匹配良好的邻居生长子图

从匹配成本最低的一对初始匹配节点（以地图中一个区域为一个节点）开始，使用章节5.2.3.3描述的 KM 算法对其邻居进行匹配，随后其匹配成本低于阈值的邻居对  $(AN_i, BN_j)$  被放入 *open\_list*。*open\_list* 是保存接下来要访问的区域对的列表，*close\_list* 是保存已访问的列表。当 *open\_list* 不为空，该算法会选择 *open\_list* 中成本最低的对来生长其邻居匹配。当 *open\_list* 为空时，选择未访问的最佳潜在匹配区域对作为下一个初始节点对以开始下一次子图生长。使用这种不断生长子图的算法，可以从每对初始节点获得一个公共子图。然后我们舍弃只包含一个节点的子图。算法 5 描述了生长子图的过程，其中函数 **BESTNEIGHBORS** 在算法 4 中描述，*Threshold* 为自定义的阈值，用来过滤掉错误的匹配项。

#### 5.2.4.3 从公共子图计算地图变换

由于从损坏的地图到正常地图可能没有一致的变换，在这种情况下算法不使用聚类算法估计地图间的变换。相反，算法为每对匹配的区域计算它们之间的变换并计算它们的重叠面积以检查它们是否是正确的匹配。但不计算全局变换。

对于图 5.1 中的示例，两个给定地图之间存在一致的变换。所以我们可以从它们的最大公共子图中计算地图之间的全局变换。但这对于损坏的、无一致变换的地图匹配没有意义。

### 5.3 地图匹配实验

本章节进行了三个实验。第一个实验是特征选择实验，在章节 5.3.2 中将进行介绍。该实验比较了不同特征对区域匹配的影响。章节5.3.3的实验探索 ICP 误差的影响。随后，设置了两个实验将本文的方法与其他最先进的算法进行比较。一个是在章节 5.3.4.1 中描述的，该实验将启发式聚类法与图像配准方法和最先进的地图匹配方法进行了比较。在 5.3.4.2 节中介绍的第二个比较实验将生长邻居法与非刚体点配准方法进行了比较，并展示了生长邻居法在匹配损坏的地图的应用上的良好表现。

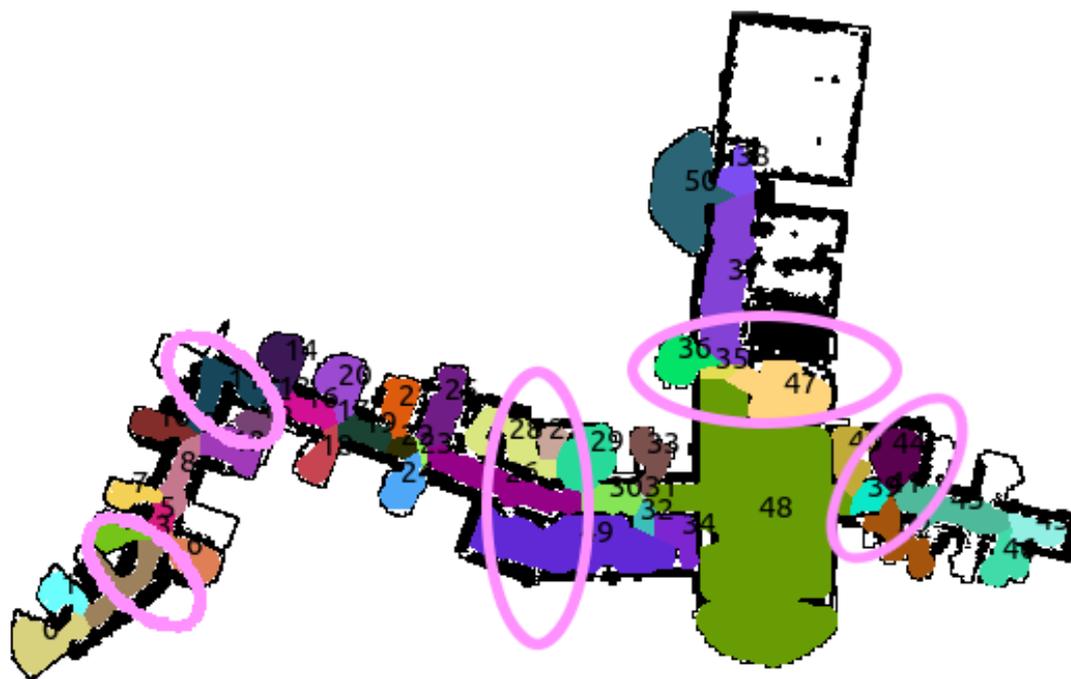


图 5.4 对一个主要包含走廊的环境的损坏地图的分割。地图拐弯处附近有一个区域重叠（用圆圈圈出）。

Figure 5.4 A segmentation of a broken map of an environment mainly containing corridors. There is a area overlap near the bend corner (circle out).

### 5.3.1 数据集与评价标准

本章的匹配实验中使用了三个数据集。来自 [Bormann 等 \(2016\)](#) 的数据集被用作完整的地图匹配和不同模态的地图匹配的例子。该数据集中的每一对地图都包含对同一环境的一幅设计平面图或人造地图和一幅机器人扫描地图，本实验中，其中一个地图会被旋转随机角度，以便各算法可以估计不同的旋转情况。对于部分重叠的地图匹配实验，则使用来自 [Carpin \(2008\)](#) 的数据集。其中包括一组 Andrew Howard 的 Fort AP Hill 数据集，该数据集有一组包含四幅对相同环境的不同部分进行构建且有一定重叠的地图 ([Howard 等, 2006](#); [Howard, 2020](#))，以及一组由两个移动机器人探索同一建筑物的不同部分收集的地图。对损坏地图的匹配实验所用的数据集则来自 [Birk \(2010\)](#)。

正如在 5.2.4 节中提到的，由于损坏的地图中包含对环境不一致的映射变换，而假设聚类方法只能获得单个变换，它不适合损坏地图的匹配。因此，在接下来的实验中，将在 Bormann 的 ([Bormann 等, 2016](#)) 和 Carpin 的 ([Carpin, 2008](#)) 数据集上使用启发式聚类法进行地图匹配以估计地图间的全局变换，而在 Birk 的数

数据集上使用生长邻居法寻找两幅拓扑地图间的公共子图来匹配两幅地图的对应部分。这两个方法的匹配结果都将与对应的真值 (ground truth) 进行比较。

对于未损坏的地图，真值包括一个变换矩阵和一幅地图 A 变换到与地图 B 正确重叠的可视化图像。对于损坏的地图数据，与正常地图相比，真值中标注了每个损坏地图中变换不一致的部分（块）的数量，例如，具有三处传感器漂移导致的弯折处的损坏地图包含四个不一致的块。实验将根据被方法正确匹配的块的百分比来评估对损坏地图的各个匹配方法。

### 5.3.2 特征选择实验

正如我们在 5.2.1 节中介绍的，可能影响区域配的特征包括区域面积、周长、凸包面积、凸包周长、通道距离、最长线段、ICP 误差、凸包的面积平方根和周长的比以及该区域的通道数等。随后，章节 5.2.1 部分介绍了特征距离的计算来衡量区域匹配的成本。本实验中使用的数据集中有四种类型的环境。每种类型在使用两对地图。每对地图数据可能属于一种以上类型的环境类型。本实验中，对上述类型的地图对，算法使用上述的每项特征距离作为匹配成本，逐项尝试进行两幅地图中的所有区域匹配，并计算区域匹配的正确率（正确对应匹配的区域对占有所有可以互相匹配的区域对的百分比）。表 5.1 中列出了所有特征距离在不同环境类型的地图中的区域匹配正确率表现。

#### 5.3.2.1 根据环境类型分析

第一种类型为主要包含走廊结构的环境。在这种情况下，alpha shape 房间检测算法对区域生成的贡献很小，因此区域边界大多遵循底层 Voronoi 图的拓扑结构：以分叉路口作为主要的区域分割处（区域相连处）。图 5.4 中显示了一个这类环境的示例。从表 5.1 的第二列和第三列可以看出，在这种环境下考虑邻居信息进行区域匹配会降低正确性。这是因为此时区域分割呈碎片化，导致很难在两个给定的地图中保持区域分割的一致性。

第二种类型是损坏的地图。地图的弯折处可能是由扫描传感器的漂移或定位的漂移引起的，导致相邻区域有小部分重叠。图 5.4 显示了弯折处引起的重叠的示例情况，该弯折处在图中用粉红色圆圈突出显示。弯折导致相邻区域的变换不一致，另外，重叠导致区域不完整。在这种情况下，邻居匹配的分数很低，变得不可靠。表格的第四列和第五列可以支持这种观点：添加邻居信息并不能提高

Features distance	Corridor	Corridor (Neighbors)	Broken	Broken (Neighbors)	Small Overlap	Small Overlap (Neighbors)	Complete	Complete (Neighbors)
Area	40.71%	18.57%	51.13%	50.95%	33.33%	8.33%	36.90%	58.93%
Perimeter	28.93%	14.29%	37.26%	36.31%	29.17%	8.33%	13.69%	22.62%
Convex Hull Area	49.11%	21.43%	53.27%	50.30%	47.92%	16.67%	40.48%	66.67%
Convex Hull Perimeter	45.00%	18.57%	59.58%	48.87%	41.67%	16.67%	40.48%	61.90%
Passage Distance	14.64%	32.14%	20.89%	26.49%	20.83%	20.83%	15.48%	10.71%
Longest Line	27.50%	11.43%	54.58%	41.13%	29.17%	16.67%	26.79%	42.26%
ICP Error	33.16%	32.09%	39.57%	44.27%	41.67%	54.17%	19.64%	3.57%
Passage Num	2.86%	12.86%	19.52%	41.85%	8.33%	8.33%	38.10%	49.40%
$\frac{\sqrt{Area/\pi}}{\frac{1}{2}Perimeter/\pi}$	26.79%	37.86%	33.04%	39.76%	37.50%	20.83%	9.52%	9.52%
Hu-Moments	26.62%	9.09%	47.16%	40.34%	35.71%	25.00%	23.86%	27.43%

表 5.1 匹配算法分别使用每项特征距离逐一运行，计算它们的区域匹配正确率以找出每项特征的影响以决定最后在我们的区域匹配算法中使用哪些特征。此外，我们比较了考虑和不考虑邻居的匹配成本两种情况下的表现。

Table 5.1 The matching algorithm is run with each feature separately. Their matching correctness is computed to find the effect of each feature, to decide which features to use in our area matching algorithm. Additionally we compare the performance of the matching costs with and without considering neighbors.

区域匹配的正确性。

第三种地图是地图间重叠区域较少的地图，例如只有 3 或 4 个区域是重叠的。对于这种情况，区域匹配的正确率也非常低。并且，使用邻居信息可能会导致其正确率大幅下降。

表现最好的数据类型是完整的地图案例。这种类型的数据我们使用不同模态的地图作为输入，即包括设计平面图和机器人地图。图 5.2 中显示了一个示例。地图的不同模态可能会影响区域分割的一致性，但通过将网格地图抽象为拓扑表示，本文的方法效果很好。在这种情况下，邻居信息改善了结果，因为两幅地图的拓扑结构都是稳定而且是良好的结构表示。表 5.1 的最后两列支持此结论。

### 5.3.2.2 根据特征距离分析

接下来我们通过特征距离分析表 5.1。可以观察到，面积、凸包面积和周长对区域匹配具有相似的影响，是获得最高区域匹配正确率的特征距离。在所有类型的地图中，在不考虑邻居匹配的情况下，它们的匹配至少获得 30% 的正确率。使用通道距离、最长线段、Hu 矩、ICP 误差和通道数用于匹配成本计算的表现表明它们在区域相似度测量中也很有效。通道距离的平均正确率为 20.25%，而最长线段的正确率为 31.19%。使用特征 Hu 矩匹配各类地图的区域，正确率均在 20% 以上。因此，算法将在匹配算法中使用这些特征距离。此外，由于算法使用通道距离和最长线来计算区域的旋转中心，所以这两个特征距离也将用于接下来的区域相似度计算。

ICP 误差在一些地图中获得了很高的分数，在 39.57% 以上，例如损坏的地图和小重叠的地图。完整地图使用 ICP 误差进行匹配的正确率很低，因为这里使用了不同模态的地图（例如机器人地图和布局图）作为输入，则其中两个地图的形状不完全匹配。而不同模态的地图匹配在机器人应用中是尤为重要的。ICP 误差的计算比较耗时，作者会在下一节（章节 5.3.3）中详细讨论在区域匹配中使用 ICP 误差是否可以大幅提高地图匹配的准确率，从而决定是否牺牲计算时间以将其添加为特征距离测量。

周长，尤其是区域周长，仅在损坏的地图匹配中拥有 30% 以上的正确率，这是因为损坏地图中只包含一种模态的输入，并且损坏图中对应的区域几乎相同。而在其他环境类型中，这两个特征无法高精度地测量区域相似度，获得的正确率低于 30%。

鉴于这些数字，我们决定选择这些特征用于接下来的匹配实验：面积、凸包面积、Hu 矩、通道距离、最长线和通道数。

### 5.3.3 ICP 误差对区域匹配成本的影响

权重向量  $W_f$  中的元素为各特征距离用于计算匹配成本的权重因子，依次对应：面积大小、凸包面积、ICP 误差、通道距离、最长线段、通道数和 Hu 矩这些特征距离的权重。

正如在章节 5.3.2 中所示，ICP 误差在区域相似性测量中表现良好。然而，ICP 算法的计算需要大量的时间。本实验将通过比较在有和没有 ICP 误差作为特征距离的两种情况下，匹配算法的匹配正确率和计算时间来决定是否在区域匹配中使用它。

本实验中，两种方法（启发式聚类法和生长邻居法）都使用两个权重向量设置来运行。要在匹配成本计算中使用 ICP 误差，权重向量设置为  $W_f^1 = [0.1 \ 0.1 \ 0.01 \ 0.1 \ 0.1 \ 0.01 \ 0.1]$ ，其中 ICP 误差的权重为 0.01。不使用 ICP 误差，权重向量则设置为  $W_f^0 = [0.1 \ 0.1 \ 0.0 \ 0.1 \ 0.1 \ 0.01 \ 0.1]$ ，其中 ICP 误差的权重为 0。向量元素如此设置是为了让所有特征距离和权重的乘积处在同一个数量级。

表 5.2 显示了在不同数据集上使用两种权重设置的两种方法的平均正确率。可以观察到把 ICP 误差加入匹配计算会降低我们的算法正确率，包括对启发式聚类法和生长邻居法。而且，在匹配算法中包含 ICP 误差比不包含要花费更多的时间。因此，在接下来的比较实验中，算法不会使用 ICP 误差加入区域匹配的成本计算。

### 5.3.4 与最先进的地图匹配算法的比较实验

#### 5.3.4.1 启发式聚类法

为了展示使用启发式聚类法进行地图匹配与普通图像配准方法相比的优势，两种流行的图像配准方法加入了这个比较实验。第一个方法是基于尺度不变特征变换 (Scale-invariant feature transform, SIFT) 特征描述子、并使用随机抽样一致 (RANDOM SAMPLE CONSENSUS, RANSAC) 算法的图像匹配 (Fischler 等, 1981)。使用 OpenCV 进行实现，并报告包括特征提取和匹配在内的算法运行时间。第二个方法是点集与点集匹配的迭代就近点法 ICP (Iterative Closest Point)，使用点云库 PCL (Point Cloud Library) 的实现 (Rusu 等, 2011b)。与章节 5.2.1.5 中描述的

Dataset	Bormann (HC)	Carpin (HC)	Birk (NG)	Time (s)
$W_f^0$	100.00%	88.89%	77.92%	2.41
$W_f^1$	83.33%	88.89%	74.79%	24.68

表 5.2 对于 Bormann、Carpin 和 Birk 的地图数据，匹配算法启发式聚类法 (HC) 和生长邻居法 (NG) 分别在有和没有 ICP 误差作为特征距离的情况下运行。我们看到 ICP 误差并没有改善匹配。由于 ICP 计算非常耗时，我们决定在区域匹配中不使用 ICP 误差。

**Table 5.2** The matching algorithm, Hypothesis Clustering (HC) and Neighbor Growing (NG), are run with and without the ICP error as feature distance, for the Bormann, Carpin and Birk maps. We see that ICP error does not improve the matching. Since the ICP computation is quite time-consuming, we decided not to use ICP error in the area matching.

的 ICP 误差计算类似，即，使用 12 个初始角度猜测运行 ICP。然后从这 12 次 ICP 运行中返回具有最低误差的最佳匹配作为最终匹配结果。对于这种 ICP 方法，2D 网格图中的每个占用单元格都是一个点。此外，本实验选择了最先进的匹配方法 Shahbandi 等 (2019) 与启发式聚类法进行比较，该方法也是依赖于区域分割的方法。

本实验在数据集 "Bormann" (Bormann 等, 2016) 和数据集 "Carpin" (Carpin, 2008) 上进行比较。数据集 "Bormann" 即章节 3.4.1 的分割实验中使用的数据集，其包含有一些针对同一环境的不同模态的地图，例如 *lab\_c* 和 *lab\_c\_scan*、*lab\_d* 和 *lab\_d\_scan\_furnitures*、*lab\_f* 和 *lab\_e*。Carpin 的数据集则提供仅有部分重叠的地图用于匹配，即来自 Fort Hill 数据集的四张地图和两对长廊地图。对于 Fort Hill 数据集中的 4 幅地图（标记为 0、1、2、3），我们分别进行 6 组实验：0 到 1、2、3；1 到 2、3；2 到 3 共 6 对地图的匹配。

实验中评估算法匹配结果的第一步是通过对匹配结果进行可视化，让人类判断地图匹配是否正确。这很容易没有偏见地完成，因为算法对两幅地图的匹配要么成功要么灾难性地失败。若是地图大体上对齐成功但有小的旋转和平移误差，亦认为是匹配成功。成功匹配的地图对的百分比显示在表 5.3 中。只有成功匹配的地图，才会被用于计算平均旋转误差  $err_R$ 。如果旋转矩阵  $R_{result}$  和 ground

Dataset	Bormann		Carpin		Time (s)
Methods	Correct	$err_R$	Correct	$err_R$	
SIFT	00.00%	-	77.78%	<b>0.0191</b>	<b>0.28</b>
ICP	50.00%	<b>0.0224</b>	33.33%	0.0351	13.38
Shahbandi	50.00%	0.0581	55.56%	0.0394	16.70
Ours (HC)	<b>100.00%</b>	0.0404	<b>88.89%</b>	0.0380	3.06

表 5.3 地图匹配结果比较。我们的启发式聚类法与基于 SIFT 特征的图像匹配法、ICP 点云配准法和 Shahbandi 等 (2019) 的工作进行了比较。我们的方法获得了最高的匹配正确性，并且比最先进的方法 (Shahbandi 等, 2019) 更快。

Table 5.3 Map Matching Comparison. Our Hypothesis Clustering compares with SIFT feature-based image matching, ICP registration and the work of Shahbandi 等 (2019). Our method obtained the highest matching correctness and is faster than the state-of-the-art method ((Shahbandi 等, 2019)).

truth 旋转矩阵  $R_{GT}$  是相同的，则  $R_{result}R_{GT}^T$  应该是恒等于单位阵的。因此，它到单位矩阵的距离则被计算作为旋转误差：

$$err_R = \frac{1}{correct_{num}} \sum_{correct\_matches} \|R_{result}R_{GT}^T - I\|_2, \quad (5.13)$$

其中  $correct_{num}$  是正确匹配的地图对的数量。

各方法的正确率和运行时间见表 5.3。在这些方法中，可以观察到，启发式聚类法在两个数据集中都产生了最高的正确率。为了更好地理解  $err_R$  的值，此处计算在旋转误差值为  $1^\circ$ 、 $1.5^\circ$  和  $2^\circ$  的时候， $R_{result}$  和  $R_{GT}$  之间的差值  $err_R$  的值作为参考：分别为 0.0247、0.0370 和 0.0493。将表 5.3 中的值对照这些值，可以看到，启发式聚类法的旋转误差低于  $2^\circ$ ，从而得出结论：启发式聚类法达到了地图匹配的准确度要求。

图 5.5 显示了各种方法的部分匹配结果。对于不同模态的地图，启发式聚类法 (HC) 比其他方法获得了至少高出 50.00% 的匹配正确性。这是因为启发式聚类法关注环境结构和一些固定的环境因素，例如门和房间。可以观察到，Shahbandi 的方法在具有强自相似性的地图上表现更好，因为它耗尽了搜索空间来寻找最佳解决方案，但作者认为这使得他们的算法不适合匹配大型环境的地图。启发式

聚类法在高自相似度图上仍然表现良好，运行时间比 Shahbandi 的方法少得多，是因为启发式聚类法只在潜在的正确解中进行搜索。

对于包含长走廊的地图 (*LongRun* 和 *LongCorridor*)，除了启发式聚类法，没有一个方法能正确匹配它们。在这些地图中，角点特征无处不在，这使基于特征的配准算法对特征的提取相当混乱。此外，两对走廊地图相互之间仅有小部分的重叠，也属于小部分重叠的地图。为了使两幅地图的不完整走廊间能有更多重叠区域匹配，算法在其分割中不应用房间检测。

对于仅有部分重叠的地图进行匹配，可应用于离线的多机器人地图合并（地图融合）。由于不完整的区域重叠削弱了邻居之间的连接关系，这会影响启发式聚类法的匹配正确性。然而，对于 Fort Hill 数据集的地图，地图中的区域特征丰富且差异很大，这有助于匹配，尤其是基于 SIFT 的匹配。由于地图间只有部分重叠，(*LongRun*、*LongCorridor* 和 *Fort Hill* 数据集)，所有方法在这些数据集上的失败率都很高。但是值得注意的是，启发式聚类法在匹配部分重叠的地图时表现最好，如图 5.5 所示，由此展示了本算法在多机器人地图融合应用上的优越性。但是，由于启发式聚类法在很大程度上依赖于两个地图分割的一致性。在这个数据集中，有一个启发式聚类法未能正确匹配的地图对，即 Fort Hill 数据集的地图 2 和 3，如图 5.5 的最后一行所示。这是因为两张图的重叠部分区域不完整，导致对应部分的分割不一致。

ICP 算法在部分重叠的地图上表现不佳，因为最近点关联在这种情况下不能很好地匹配地图。ICP 不适用于具有不同模态的地图和表现出一些旋转对称性的地图。

总体而言，本文的启发式聚类法 (HC) 在两个数据集中都获得了最高的正确率并达到了良好的准确性。鉴于其在不同模态的地图上表现出的巨大优势，启发式聚类法比基于 SIFT 的匹配慢是可以接受的。此外，启发式聚类法和 ICP 点云配准方法和最先进的办法 (Shahbandi 等, 2019) 相比，具有更高的正确率和更短的计算时间。

#### 5.3.4.2 生长邻居法

生长邻居法 (NG) 是用于匹配损坏的地图，作为一个补充方法。该方法充分利用了图的邻居信息。在该实验中，作为参照结果的真值给出损坏地图的块数。生长邻居法可以生长多个子图，实验中作者将子图中成功匹配的区域所属的块



图 5.5 实验结果展示了不同算法的匹配结果。结果表明，与其他算法相比，我们的算法具有更好的表现。

Figure 5.5 Experiment result to show the matching results of different algorithms. The result shows that our algorithm has a better performance compared to the others.

的数量计算为该算法正确匹配的块数。本实验将邻居生长方法与 Ma 等 (2015) 的工作 *RPM-L2E* 进行比较，后者是一种通过点对点齐的非刚体点配准方法。实验计算其点配准结果看上去大致可接受的块作为正确匹配的块进行计数。

Methods	No Rotation	Rotated Maps	Total
RPM-L2E	68.54%	2.5%	35.52%
Ours (NG)	77.92%	75.63%	76.77%

表 5.4 我们的邻居生长 (NG) 方法与 Ma 等人的工作 (RPM-L2E) 比较对损坏地图的匹配能力，通过两个地图之间正确匹配的块的百分比来评估。

Table 5.4 Comparison of our Neighbor Growing (NG) method with Ma et.al.'s work (RPM-L2E) on matching broken maps, evaluated by the percentage of parts that are correctly matched between the two maps.

实验使用来自 Birk (2010) 的损坏地图数据集，其中包含来自两个环境的地图。对于每个环境，都包含有一个正常的完整地图和几个从它弯曲而来的损坏的完整地图。作者将所有损坏的地图进行旋转作为额外的样本。在图 5.4 的例子中，弯折处用粉红色圆圈突出显示，其中一些门被遮挡。因此，作者删除了破损地图中的门，因为有些门被弯折处的墙壁挡住了，会导致本文的地图分割算法对一部分区域无法分割。接下来，将运行这些方法来匹配所有损坏的地图与其对应的正常地图。

图 5.6 显示了非刚体点配准方法 RPM-L23 和邻居生长方法 (NG) 的匹配结果。表 5.4 显示了两种方法的平均块匹配正确率。表格显示，邻居生长方法 (NG) 发现了更多的弯折块。在没有旋转和有旋转的情况下，邻居生长方法的块匹配正确率比 RPM-L23 分别高 9.38% 和 73.13%。可推断 RPM-L2E 方法受到旋转的影响比较严重。从图 5.6 可以看出，当对应点之间没有旋转时，RPM-L23 方法表现良好。但是，当地图发生弯曲时，某些部分相对正常地图有旋转，非刚体点配准方法无法配准被弯曲的点。而在这种情况下，邻居生长方法仍然可以将一些弯曲区域与正常地图中的对应部分进行匹配。有时，某些弯曲块太小而无法单独匹配，例如 "Robot\_cast" 中“无旋转，3 块”这个样本里的左上部分。然而，它们可以通过生长为其他块中的区域的邻居来匹配。因此，本实验表明，基于区域图的生长邻居法具有通过邻居生长子图来匹配损坏地图中不同部分的能力。

## 5.4 本章小结

本章介绍了两种基于二维室内拓扑地图的地图匹配算法，启发式聚类法和生长邻居法。这两种方法其实可以看作是一种方法的两种变体，因为两种方法的核心都是根据从区域提取的特征进行地图间的区域匹配。在这两种方法中，首先都是先通过地图分割生成以区域为节点的拓扑地图，随后对每个区域都进行特征提取，然后把两幅地图的所有区域通过特征尝试互相匹配，若两个区域的匹配成本低于设定阈值，则被认为是可能正确的匹配。当输入地图之间有一致的全局变换，可使用启发式聚类法进行地图匹配。启发式聚类法为所有可能正确匹配的区域对计算两个区域之间的（平移与旋转）变换，然后将该变换应用到两个地图之间，通过地图重合情况筛除掉错误的变换，最后通过聚类法选出最正确的地图间变换。当输入地图之间无一致的全局变换，比如当地图损坏时，可使用生长邻居法来匹配地图的不同部分。生长邻居法的原理是通过图的连接关系，把可能正确的邻居区域节点加入公共子图。公共子图是两幅拓扑地图成功匹配的子图。这是一种利用图结构的匹配方法。最后，本章通过为这两种匹配方法分别设计的对比实验，展示了这两种方法的匹配正确率表现良好。

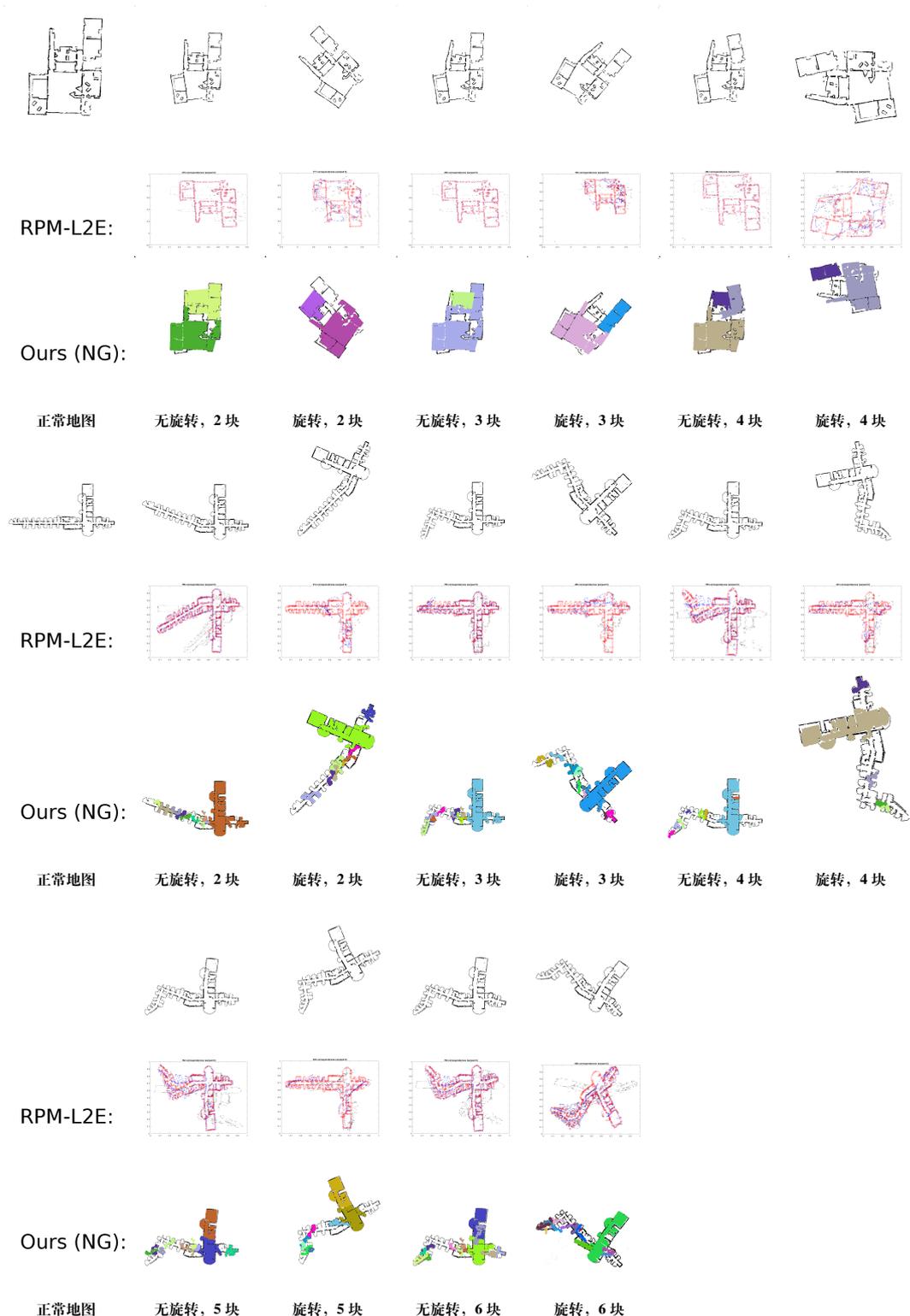


图 5.6 我们的生长邻居法与非刚体点配准方法 (PRM-L2E) 的比较。我们的结果中相同颜色的区域表示这些区域属于同一个子图。

Figure 5.6 Comparison of our neighbor growing method with the non-rigid transformation method (PRM-L2E). Areas in the same color in our results mean that these areas belong to the same subgraph.



## 第6章 对三维环境的多层次拓扑-度量图表示

### 6.1 多层次拓扑-度量图结构总览

章节4介绍了区域图 Area Graph，基于区域图我们进一步提出了多层次的拓扑-度量图表示。该拓扑表示是基于三维度量图构建的，该实现以三维点云地图为输入，通过将三维地图划分为区域作为拓扑节点。该方法的优点体现在输入和输出两方面。在输入方面，该方法使用以天花板和地板为分隔的多楼层建筑结构的点云作为输入。在输出方面，该方法可（1）生成包含不同维度的度量信息的拓扑表示，这些表示适用于不同的机器人应用；（2）根据楼层-区域-空间块的划分来构建多层级的拓扑表示，该表示提取多层建筑的拓扑结构。该算法通过从三维体素占用地图提取空闲的空间块，通过添加连接空间块的通道、合并空间块生成区域作为拓扑节点，随后使用 Area Graph 生成算法来产生更好的区域分割结果来构建更好的拓扑地图表示。最高层的全局拓扑表示通过连接多个楼层来构建。由于本拓扑表示构建过程中包含的对三维空间的区域划分算法可看作作者提出了一种创新的地图分割算法，本章的最后使用一系列对比实验展示本算法在地图分割上的准确性和对地图噪声的稳健性优势。

#### 6.1.1 多维度的度量信息

本章的三维拓扑-度量图的提出是受到 Blochliger 等 (2018) 的启发，不同于它们使用障碍物来表示地图，我们使用可遍历的未占用空间来表示地图。该拓扑-度量图支持不同维度的表示（示例图见图6.1）。值得注意的是，所有维度的地图都是基于三维空间的，此处的“维度”是指图的顶点和边包含的信息的维度。（1）0D 表示是一个纯粹的拓扑图，其表示空闲三维空间及其之间的连接形成的拓扑关系。（2）1D 表示中的每一个节点是空间中记录了三维坐标系的一个点。（3）2D 表示中的每个节点记录的是二维区域，边为区域的连接处边界段，用于实现机器人在区域间的转移。（4）3D 表示中，其节点记录的是三维空间块，并以空间块之间的二维（交界）平面作为拓扑图的边，则机器人可以通过连接不同空间块的这些平面穿越不同的三维空间。

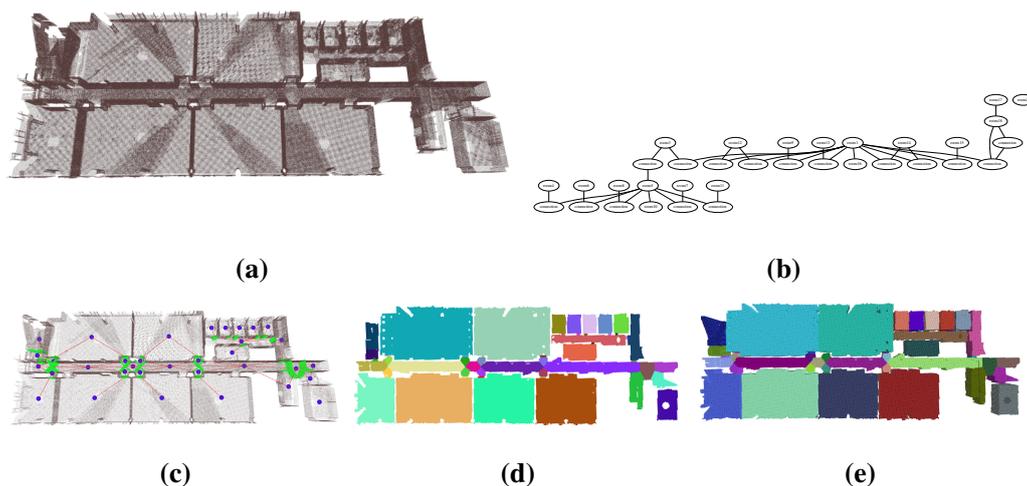


图 6.1 输入三维点云 (a)，本算法产生不同维度的输出：(b) 0D 拓扑地图；(c) 使用记录了坐标的三维点作为节点的 1D 地图；(d) 以区域为节点的 2D 地图；(e) 记录空间块信息的 3D 地图。

Figure 6.1 Input a 3D point cloud (a), then different results with different dimensional metric information are provided by our method: (b) 0D topological map; (c) 1D map whose vertices are points with coordinates; (d) 2D map with regions as vertices; and (e) 3D map with volumetric information.

### 6.1.2 多层次的拓扑结构

本文提出的多层次拓扑-度量表示的度量信息是多维度的（如上一节6.1.1所述），而其拓扑结构是多层次的：高级别的拓扑节点包含低级别的拓扑节点或低级别的拓扑图，即，高级别拓扑节点是多个低级别拓扑节点的父母。(1) 在最低级别的拓扑表示里，节点记录的是空间块 (volume)，代表一块空闲空间。空间块是作为用于合并区域的过渡节点，是一段连续空闲的三维空间，并不代表完整房间或走廊。(2) 在次低级别里，其节点为区域 (region)，本算法的目标是把各个完整的房间、走廊、门所界定的空间分别划分为单独的区域。其中，门所处的小区域，被标记为连接处 (connection)，多数区域之间有门连接。由于根据房间和走廊划分的区域可能非常大，我们将这一层的区域称为大区域 (big region)。(3) 算法对大区域进行进一步划分，作为第二层拓扑表示，作为子区域层。(4) 本表示方法支持以多层建筑的点云为输入，最高级的拓扑图以每一层 (storey) 作为一个拓扑节点。因此，本表示的多层次结构共四层：楼层-大区域-子区域-空间块。

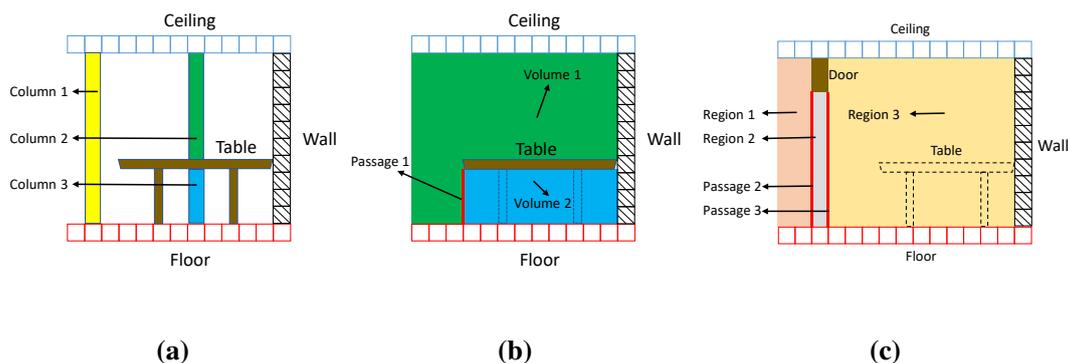


图 6.2 (a) 室内空间的纵列 (*column*) 示意图 (图中仅显示了三条纵列), 实际上空间中布满纵列。(b) 具有相似顶高的纵列组成一个空间块 (*volume*)。空间块之间的接触面作为通道 (*passage*)。(c) 图中展示了三个区域, 其中区域 2 是门所处的区域。区域 3 代表的是房间, 由图 (b) 的 *volume 1* 和 *volume 2* 合并而成。三个区域之间有两个通道。

**Figure 6.2 (a) columns in an indoor environment (only three example columns are shown). Actually, the free space is filled with columns. (b) Columns with similar top height form a volume. The contact surface between volumes is passage. (c) There are three regions: Region 2 is a door; Region 3 is a room, formed from volume 1 and volume 2. There are two passages between the three regions.**

## 6.2 构建算法

本方法以三维点云作为输入, 给定一栋包含多个楼层的建筑完整点云, 算法将生成多层次的拓扑表示, 不同层次的节点所代表的空间不同, 分别代表楼层、区域和空间块。为了构建楼层层次的拓扑表示, 算法通过识别天花板和地面来区分楼层, 图中每个节点代表一个楼层。其中, 每个楼层则包含多个区域 (例如, 房间), 而每个房间由空间块 (*volume*) 组成。为了构建这种多层次的表示方式, 对于每一层楼的点云, 算法使用如下方法来提取拓扑表示, 图6.2展示了示意图。(1) 首先, 遍历空间生成最基本单元——纵列 (*column*): 空间中竖直方向上连续的非占用单元组成一纵列, 记录为  $(x, y, z_1, z_2)$ 。(2) 将顶高相似的纵列组合为同一 *volume*。当两个 *volume* 之间共享一个通道 (*passage*), 即一个接触面, 则会被合并入同一个区域。不同的区域之间通过检测门来划分, 中间没有门的 *volume* 则合并为一个区域。(3) 使用 Area Graph 生成算法进行二维地图分割, 以对大区域进行进一步分割。

### 6.2.1 点云预处理及楼层检测

对于输入的建筑点云，首先要使  $Z$  轴正方向为世界中的垂直向上的方向，可以通过主成分分析 (Principal Component Analysis, PCA) 算法来找到建筑的垂直方向。随后，使用开源的点云算法库 (Point Cloud Library, PCL) (Rusu 等, 2011a) 的 *voxel filter* 来将点云转换为三维体素占用地图 (三维网格地图)，密度限制为  $5\text{cm}$  的分辨率。点云之中常会包含噪点，噪点的一个主要来源是激光传感器在玻璃上的反射。我们使用 PCL 库的 *Euclidean Cluster Extraction* 算法来去除噪点：我们将相互距离小于阈值  $20\text{cm}$  的点认为一个聚集簇，若簇中的点数少于  $100$ ，这些点会被认为是噪点从而被删去。

受 Armeni 等 (2016) 的启发，我们使用峰值检测算法来检测楼层。由于激光不能穿越被视为障碍的地板和天花板，点云中相邻楼层的地板和天花板之间是空的，而地板和天花板表面的点最密集。我们沿着垂直方向 ( $Z$  轴) 统计不同高度的点的数量，形成直方图。容易想见，包含点最多的峰值处是地板和天花板所在的高度，据此我们可以检测出天花板和地板。基于实际情况中，只有小部分建筑地板是崎岖的或倾斜的，此处我们假设地板是水平的。

我们使用一系列窗口滤波器，结合 Otsu (1979) 的方法来定位天花板和地板并且不需要人工设定阈值。该滤波器可以表示为：

$$g_c(s) = \frac{1}{c} \mathbb{I}[|s| \leq c] \quad (6.1)$$

其中， $\mathbb{I}[A]$  是指示器函数，当条件  $A$  为真时结果为  $1$ ，否则为  $0$ 。图 6.3 (左) 展示了一幅从点云生成的直方图。可以看出，在地板附近，有时会有多个峰值，这是由地板附近的噪点或由于地板不平坦引起的。不同的窗口大小会影响窗口滤波器的效果 (见图 6.3 (中))，因此本算法将通过以下策略检测地板和天花板：首先筛选出被最多窗口检测为峰值的点簇，随后以这些窗口中窗口大小最小的结果为准。因为较小的窗口分辨率较高。图 6.3 (右) 展示了通过该策略检测到的楼层地板和天花板。

### 6.2.2 纵列的生成

三维体素占用地图 (3D voxel occupancy map) 中，每个体素单元 (voxel) 可以有两种状态：要么被占用，要么空闲。而点云仅表示有障碍的位置，点之间的空隙不一定是空闲的，因此不便于通过点云来确定三维空间中的连续空闲空间。本

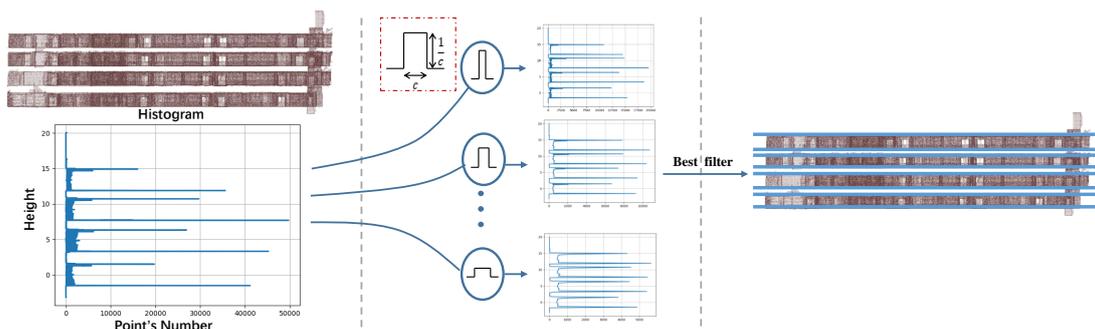


图 6.3 左：点云侧视图及沿着点云  $Z$  轴统计点数的直方图。中：从上到下窗口大小分别为  $c \in \{2, 4, \dots, 10\}$   $cm$  的窗口滤波结果。右：检测到的地板与天花板（标注为蓝色线条）。  
**Figure 6.3 Left: The side-view of a point cloud and the histogram to count point number along the  $z$ -axis. Middle: Window filter results, whose window sizes from top to the bottom are  $c \in \{2, 4, 6, 8, 10\}$   $cm$ . Right: The detection results of ceilings and floors (blue lines).**

算法为了检测成块的空闲空间，需要将输入的三维点云转化为三维体素占用地图，这一步使用开源工具 *sdf\_tools* (Hayne 等, 2016) 来实现。

我们引入纵列 (column) 这个概念，一条纵列是由相同  $x$  和  $y$  坐标上、沿着  $z$  方向空闲的一段连续体素单元组成，记为  $(x, y, z_1, z_2)$ 。所有纵列的底面单元格的边长（包括长和宽）都是相同的，等于网格的分辨率。纵列的高是该  $(x, y)$  坐标上连续空闲的空间长度决定的，用  $(z_1, z_2)$  这两个参数记录纵列的首尾高度坐标。图6.2a 通过三维空间的侧视图展示了三个纵列的例子。

通过上一节的方法，可以检测出分隔楼层的地面和天花板，算法将利用这个结果来避免在地面和天花板中间生成纵列。纵列是从顶部开始，通过遍历单元格是否空闲来生成，直至触碰到本楼层的地板则停止。注意到，同一楼层中的同一  $(x, y)$  上可能有多条纵列，因为三维空间中可能有障碍物会中断连续空闲的状态（例如空间中的桌子）。另外，如果纵列的顶部是无限高，该纵列也会被移除，因为它可能不属于室内的环境。

### 6.2.3 空间块与通道的生成

空间块 (volume) 是指一块连续空闲的有限三维空间，它通过组合连续相邻的纵列来生成，图6.2b展示了空间块的示意图，图6.4b展示了在真实建筑点云中生成的空间块。

本算法通过如下过程组合纵列来生成空间块：首先，随机地选取一条纵列作为种子，从它出发迭代地搜索相邻的纵列，当新纵列与上一相邻纵列有相似的顶

部高度 ( $z_2$  大小相差小于 10%)，将新纵列加入空间块。该算法可以很好地将同一顶部表面（例如，一块不太倾斜且比较平坦的天花板）下的空闲空间提取为一个空间块。由于室内总是会存在一些杂物（例如家具）成为障碍物，纵列合并时并不要求它们的底部高度相近。这样的设置使纵列容易组合成一个水平方向上几乎铺满房间的大空间块，这样的大空间块将作为生成的区域，算法的目标是一个房间仅用一个区域表示。

但是，空间块的底部不齐将导致它不能被用于地面机器人，为此算法建立一个空间块层级的拓扑图，图中每个拓扑节点所包含的空间块的底部是连续的。空间块之间的 *passage* 标记其两边的空间块高度是否是连续的。该拓扑图中的边表示的就是这种连接两个空间块的接触面，称为通道 (*passage*)。

本文将空间块层级 (*volume level*) 的拓扑图记为  $\mathcal{G} \triangleq (\mathcal{V}, \mathcal{E})$ ，其中  $\mathcal{V}$  是图中的节点集， $\mathcal{E}$  是图中的边集。算法通过如下步骤生成通道，并将其构建为拓扑图的边。在上一步生成空间块时，算法遍历所有的纵列，将每一个纵列都标注上所属的空间块的序号 (*ID*)。为了寻找属于通道的纵列，算法再次迭代地通过邻居关系遍历所有纵列，当发现新纵列与其邻居（上一条纵列）所属空间块的序号不同时，算法将两空间块相交处（共享）的体素单元格加入它们之间的 *passage* 中。我们将空间块  $i$  与空间块  $j$  之间的 *passage* 记为  $P_j^i$ 。通道  $P_j^i$  中的单元格会形成两个体素之间的接触面。当两个体素之间不止有一个通道时，比如某个房间与走廊之间有两个门，上述步骤生成的一个通道可能包含两个不连续的点集。为此，我们还要对每个通道使用聚类算法，使这类情况下产生的通道被划分为成两个通道，并分别创建为拓扑图中的两个边。最后，算法将每个空间块作为该层级拓扑图的一个节点，每个通道作为拓扑图中的一条边，图 6.4a 可视化了一幅由此方法构建的空间块层级的拓扑图。

#### 6.2.4 区域的构建

本多层次拓扑表示中，区域级 (*region level*) 的拓扑-度量图的构建目标是使其节点所代表的区域恰好是一块封闭的空间（例如一个完整的房间），仅使用门等作为通道来进出区域。算法将每个区域标记为“房间” (*room*) 或“连接处” (*connection*)，其中“房间”代表完整的房间或走廊，“连接处”则代表门。图 6.5b 展示了从三维占用地图生成的所有区域，不同的区域被涂以不同的颜色。其中门所代表的区域由于比较小被大区域遮挡而不太明显，我们在图 6.5a 将门所处的区域

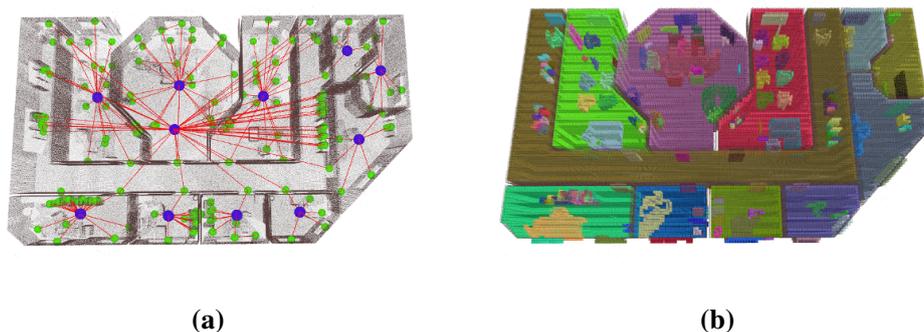


图 6.4 (a) 空间块级的拓扑图，其中体积大于  $20m^3$  的空间块的节点用蓝色表示。(b) 从三维体素占用地图生成的空间块。

Figure 6.4 (a) Volume-level topological graph, where the vertices whose volumes are larger than  $20m^3$  are visualized in blue. (b) Volumes generated from a 3D voxel occupancy map.

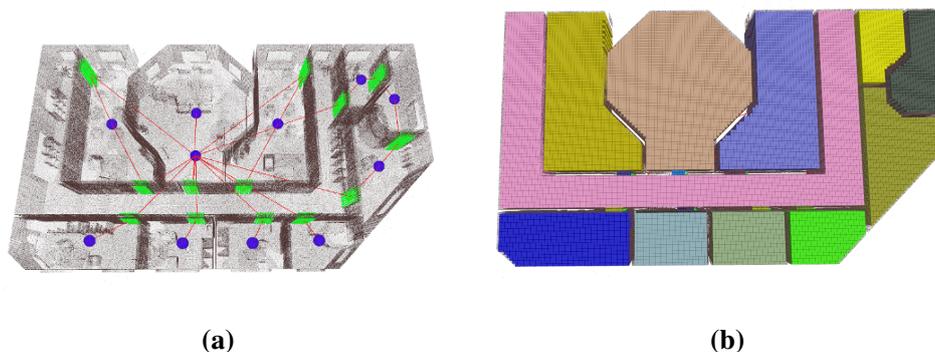


图 6.5 (a) 区域级的拓扑图。地图中的通道被标记为绿色。(b) 从三维体素占用地图生成的区域。

Figure 6.5 (a) Region-level topological graph. The passages are labeled in green in the map. (b) Regions generated from a 3D voxel occupancy map.

标记为绿色。

在室内环境下，房间通常是用门与其它区域连接的，而门所占用的区域常常比房间小很多。基于这个事实，本方法通过在空间块拓扑图中寻找“房间-连接处-房间”的结构来构建区域级拓扑图。首先设定一个体积阈值  $a_{th}$ ，随后选择超过阈值大小的空间块作为生成房间区域的种子。因为房间区域之间必须有连接处区域作为过渡，当两个房间种子之间有一条边相连（两个体积超过阈值的空间块间仅隔了一个 passage 接触面），则将它们合并，并删去其中一个种子，以使它们最终只生成一个房间。从种子空间块开始向外不断合并相邻空间块为一个区域，直到遇到另一个种子生成的空间块才停止。图6.6展示了该合并过程的示

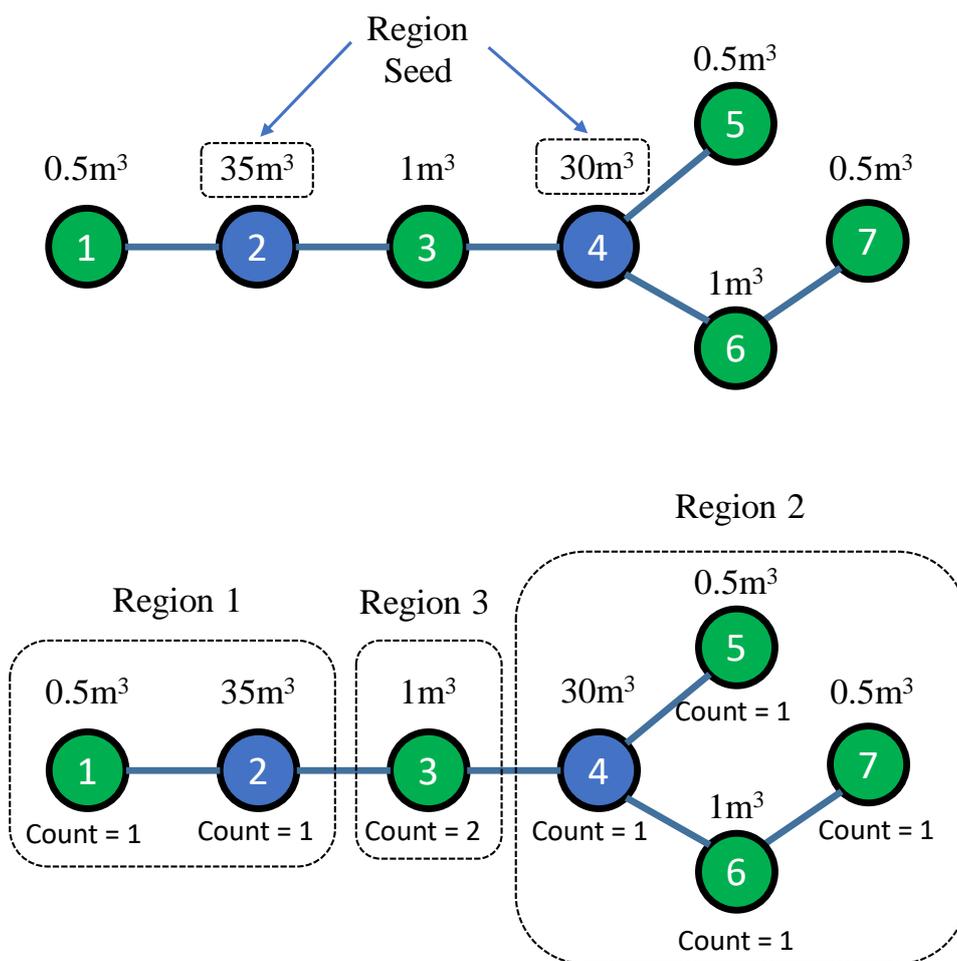


图 6.6 从空间块生成区域。

Figure 6.6 Generate regions from volumes.

意图。该过程会导致门所在的空间块同时加入门两边的区域。在完成空间块合并后，算法为每个空间块标上各自所属的区域序号。若某个空间块拥有超过一个区域序号，则将其从所属的区域移除，并创建为一个单独的区域，标记为连接处。图6.6的空间块 3 展示了这样一个示例：它会先被合并入由种子 2 和种子 4 所创建的区域，再被检测出来同时属于两个区域而被剔除出区域 2 和 4，单独成为区域 3。

上述过程中选择种子时会用到阈值参数  $a_{th}$ ，该参数的设置会影响种子的选择，进而影响区域的生成。通常来说，门的体积不会大于  $1m^3$ ，而门不能成为房间区域的种子。很显然，需要选择一个大于门并且小于最小的房间的体积作为阈值。我们测试不同的阈值设置对种子数量的影响，并将结果绘制于图6.7。图中

的蓝色折现（橙色点）是种子合并前的数量，当阈值  $a_{th}$  小于门的体积时种子最多，随着  $a_{th}$  变大，种子数量急剧下降且变动不大，当  $a_{th}$  继续增大到大于房间体积时，种子数量进一步减少。可见，在  $a_{th}$  大于门的体积的基础上，较小的阈值仍会使种子数大于真实房间数（红色水平线），这时就需要通过上述过程删除多余的种子。图中橙色线条（蓝色点）是经过种子移除过程后的种子数。可见，在较为宽泛的范围内设置阈值来生成种子，经过了种子移除后，种子数量都是稳定的（等于真实房间数）。

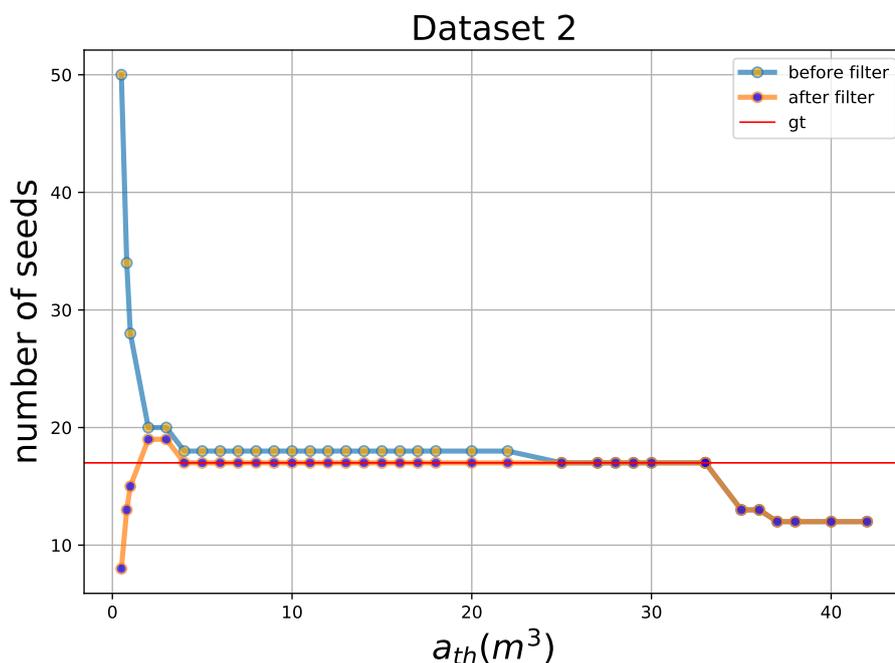


图 6.7 阈值  $a_{th}$  对种子数的影响，蓝色折现（橙色点）和橙色折现（蓝色点）分别为滤除多余种子前、后的种子数量，红色水平线为真实房间区域数。

Figure 6.7 The influence of threshold  $a_{th}$  to the seed number. The blue polyline (orange dots) and the orange polyline (blue dots) are the seed numbers before and after the filtering step, respectively. The red line is the ground truth room region number.

### 6.2.5 用 Area Graph 生成算法划分子区域

在上一节所介绍的拓扑图构建中，算法趋于将房间和走廊各自分割为独立的区域。但是，在某些情况下，用户希望这些区域被进一步划分：（1）由于传感器的问题，房间没有被分割为独立的区域；（2）建筑中的一条贯通的长走廊被划分为一个完整的区域，但是对于某些应用，用户希望将走廊划分成通向不同分叉口的子走廊区域。我们使用 Area Graph 生成算法来实现子区域划分。图6.8a中展

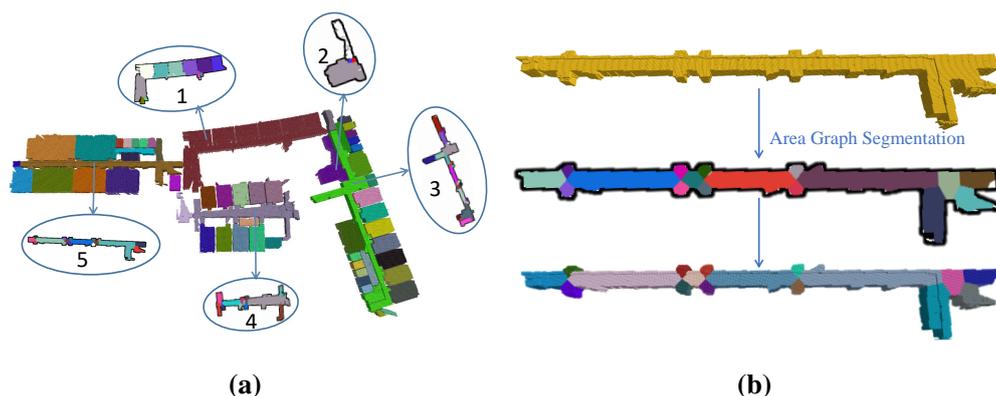


图 6.8 (a) 列举了几个大区域使用 Area Graph 生成算法划分出子区域的示例。(b) 将一个大区域的二维分割结果应用到三维地图中。

Figure 6.8 (a) Lists several example big areas that are divided into sub-regions using the Area Graph generation algorithm. (b) The 2D segmentation result of a big region is applied to the region in 3D map.

示了几个将大区域进一步划分为多个子区域的示例。其中，圆圈 1 指向的大区域包含了多个房间，这是因为房间之间使用与天花板同高的玻璃进行分隔，激光雷达在此产生的障碍点较为稀疏，因此算法将这几个房间构建为同一个区域，即产生了分割不足 (under segmentation) 的情况。当分隔房间的门与天花板等高时也容易出现这种情况。Area Graph 算法的分割结果展示在圆圈中，其正确地将几个房间分别划分为独立的区域。图6.8a中的例子 3、4、5 是连接多个房间的贯通型长走廊，大区域级的拓扑图将这类走廊划分为一个完整的区域。若将此区域分割结果应用在导航应用中，寻找房间之间的转移路径将耗费更多的时间（因为需要在走廊区域中进行像素级搜索寻找路径时搜索空间更大）。将每个分叉通道划分为一个子区域将有效提高在走廊中的路径搜索效率，而 Area Graph 构建过程中包含的地图分割算法正好适用于此种场景。

基于大区域级的拓扑图，本方法通过划分大区域为多个子区域，构建子区域层级 (sub-region level) 的拓扑表示。算法对所有超过  $20m^3$  的大区域映射成二维子地图，并使用 Area Graph 生成算法在其上对大区域进行二维分割。二维栅格地图的生成过程如下：首先初始化一张所有单元格都是占用状态的二维栅格地图，然后遍历区域中的每一个  $(x, y)$  单元格，若单元格上方有一条纵列则把该单元格状态改为空闲。由于从大区域映射的二维地图中不包含家具，因此 Area Graph 算法在由空旷的大区域生成的二维子地图上的分割准确率更高。如图6.8b所示，一

个大区域先产生二维地图并使用 Area Graph 进行分割，分割结果再被应用于三维地图，将大区域划分为多个子区域，作为子区域层级拓扑图的节点。子区域级拓扑图的边为 passage，其生成方法与章节6.2.3中介绍的方法相同，为两个区域的接触面。

## 6.3 实验评估

### 6.3.1 算法性能实验

本实验使用真实建筑点云作为实验数据集，表6.2展示了六个单层建筑点云，其中数据集 1 和 2 来自于上海科技大学的移动自主机器人实验室 (MARS Lab)，其它数据集来自苏黎世大学的可视化与多媒体实验室。

表6.2展示了输入点云及其三维和一维的输出——区域分割结果和提取的拓扑图。可见本方法在这些输入点云上的区域分割表现良好。尤其是，数据集 1 中存在由玻璃反射导致的噪点，经过 Area Graph 的再次分割，这一片的区域也被分割得很好。数据集 1 中也有少量的过分割，这是由于 Area Graph 在非结构化区域的分割导致的，但是不影响区域的可达性。数据集 3、4 中包含倾斜的天花板，但是在实验结果中，包含倾斜天花板的房间仍被分割为单独的区域，可见算法对倾斜的天花板也有一定的鲁棒性。算法的一维输出——以区域为节点提取的拓扑图将不同的区域连接起来，体现了区域之间的连接和可达性。在数据集 1 和 2 中有部分节点没有被边连入拓扑图，这是由于点云中有些门未被识别。数据集 1 和 2 中有这样的例子：由于门的上方是玻璃，使算法未成功识别门作为通道，因此没有产生相应的边。注意，数据集 2 不是数据集 1 的子集。

算法的输入是整栋建筑的点云，因此不是用于实时的应用，不要求实时的运算速度。从表6.2可看出，在所有数据集上运行我们的算法总共只花了 5 秒钟。影响运算时间的因素主要是建筑的规模和三维体素占用地图的分辨率。数据集 1 到 6 的运算时间与规模基本成正比。数据集 4 比数据集 5 规模小，但是运算时间略长，这是由于其使用了分辨率更高（单元体素更小）的体素地图。

### 6.3.2 对比实验

本算法包含的区域提取过程——首先对垂直方向的三维体素进行聚类成纵列，并通过合并纵列形成区域——可看作一种地图分割算法，本实验中我们把本算法的分割结果投影到二维平面来与其它二维分割方法作比较。表6.3展示了本

MCC	Area Graph	MAORIS	Ours (Projection)
Dataset 1	<b>0.606</b>	0.498	0.532
Dataset 2	0.589	0.615	<b>0.976</b>
Dataset 3	0.783	0.840	<b>0.993</b>
Dataset 4	0.747	0.279	<b>0.975</b>
Dataset 5	0.781	0.377	<b>0.997</b>
Dataset 6	0.930	0.646	<b>0.993</b>

表 6.1 不同分割方法在不同数据集上的 MCC 评估结果。

Table 6.1 The MCC evaluation for the different methods on different datasets.

方法与其它二维分割方法（Area Graph 生成算法 (Hou 等, 2019b) 和 MAORIS 算法 (Mielle 等, 2017)）对室内地图的二维分割结果。

本方法的输入是三维点云，与章节6.3.1所用的数据集相同，均是开源的。表6.3中的第一列是输入的二维网格地图，是从对应的三维点云映射而来的，已被滤除噪点（并未滤除三维点云中的噪点，仅在二维中滤除），作为对比方法 Area Graph 生成算法 (Hou 等, 2019b) 和 MAORIS 算法 (Mielle 等, 2017) 的输入。表格二、三列是对比方法的分割结果。表格第四列是本方法的结果（不包含使用 Area Graph 算法的子区域分割过程）。第五列是地图分割的二维参照结果，是如此生成的：在点云在门所在处进行划分，并投影到二维。

作者仍使用 Matthew 的相关系数 (Matthew’s Correlation, MCC) 作为分割结果的评估标准（同章节3.3和3.4），表6.1列出了三种方法在六个数据集上的 MCC 分数。从表中可见，本方法具有最好的分割表现——其在大多数数据集上 MCC 得分最高。本方法在数据集 1 上未获得高分，是因为该数据集中有一片区域的房间之间仅用玻璃分割，使本算法无法识别到门从而未成功将各房间分割为单独的区域。而另外两个方法对映射的二维地图进行分割，未受玻璃影响。除此片区域之外，本方法在数据集 1 的其它区域与在其它五个数据集上的分割结果都非常接近参照结果，在数据集 2-6 上都获得 97% 以上的 MCC 分数。

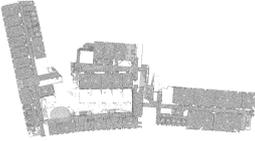
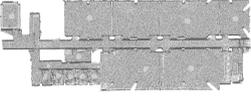
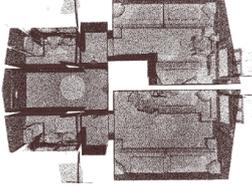
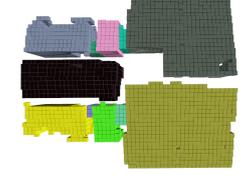
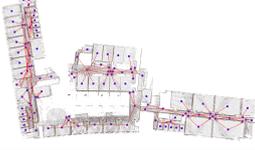
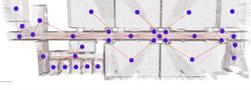
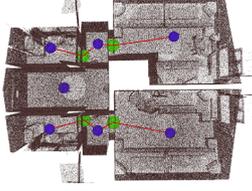
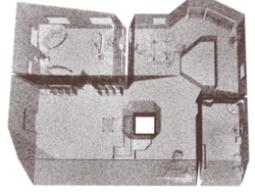
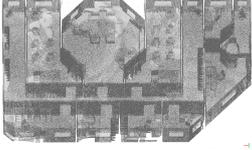
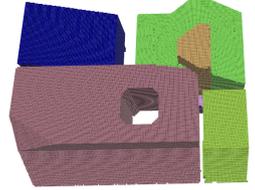
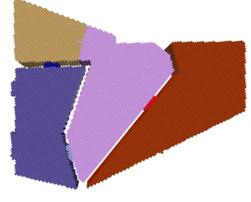
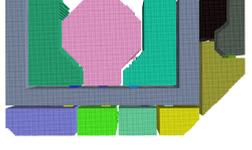
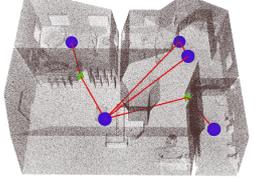
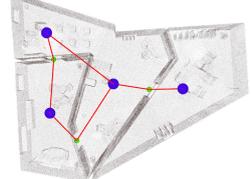
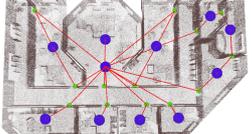
	Dataset 1	Dataset 2	Dataset 3
输入点数 长/宽/高 区域数	12,956,732 169.9m / 104.41m / 5.01m 70	2,929,680 73.57m / 28.97m / 9.33m 18	5,099,751 11.25m / 6.98m / 3.23m 7
分辨率 运算时间 (s) 体积阈值 $a_{th}(m^3)$	0.15 2.10 20	0.15 0.73 20	0.15 0.058 2
输入点云			
3D 地图分割			
区域级拓扑图			
	Dataset 4	Dataset 5	Dataset 6
输入点数 长/宽/高 区域数	5,096,515 16.24m / 13.90m / 7.02m 4	291,701 26.46m / 24.79m / 8.39m 3	1,199,532 24.15m / 16.03m / 3.37m 11
分辨率 运算时间 (s) 体积阈值 $a_{th}(m^3)$	0.10 0.48 20	0.15 0.27 20	0.15 0.31 20
输入点云			
3D 地图分割			
区域级拓扑图			

表 6.2 对不同数据集的评估实验。

Table 6.2 Evaluation experiments on different datasets.

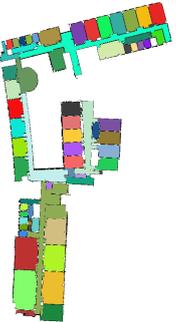
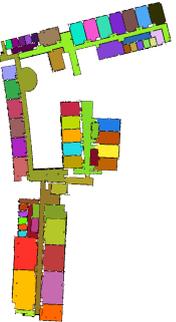
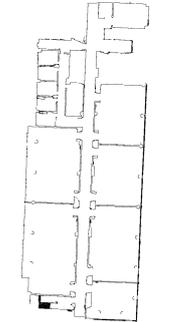
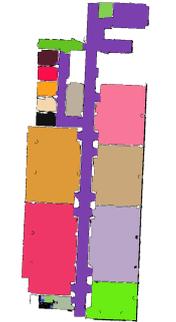
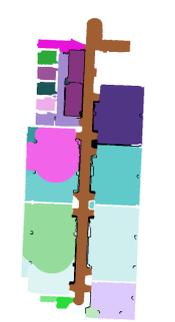
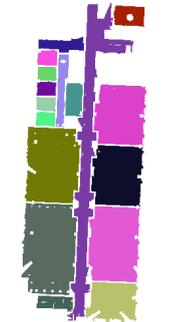
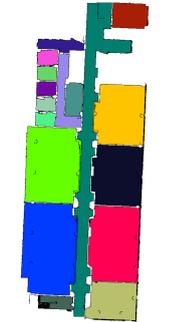
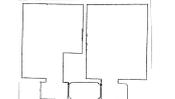
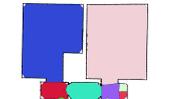
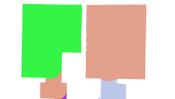
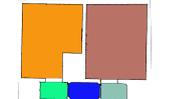
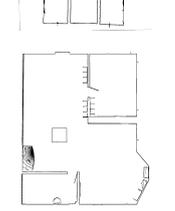
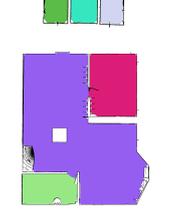
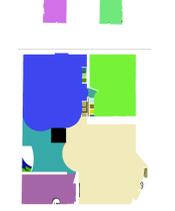
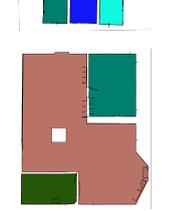
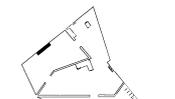
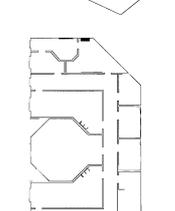
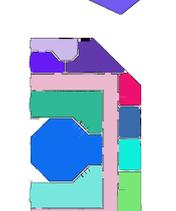
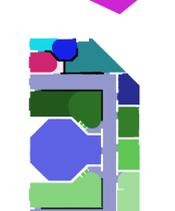
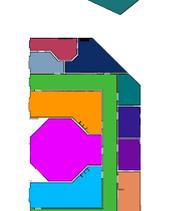
二维地图 (由点云压缩)	二维分割方法 Area Graph	二维分割方法 MAORIS	本方法 (二维结果)	参考结果
				
				
				
				
				
				

表 6.3 与两种最先进方法 (Area Graph generation algorithm & MAORIS) 的对比实验。

Table 6.3 Comparisons experiments with two state-of-the-art 2D segmentation methods (Area Graph generation algorithm & MAORIS).

## 第7章 总结与展望

本文对室内二维、三维地图提取拓扑表示的方法进行了研究，并将提出的二维拓扑地图应用于全局路径规划和地图匹配，为此提出了相应的算法。当机器人在任务中使用栅格地图时，若环境规模太大、或地图分辨率太高，都会导致算法计算量过大的问题。从地图中抽象出有意义的结构构建拓扑地图，则可以很好地解决此问题。考虑到以有意义的环境结构作为拓扑节点更有益于人机交互的应用，及生成拓扑图的自动化需求，作者首先提出了以房间、走廊等有意义的区域为节点的拓扑地图：以单层二维栅格地图为输入，提取的拓扑表示命名为区域图 (Area Graph)。并对基于区域图的路径规划、地图匹配这两个应用展开了研究。基于区域图，论文又进一步提出了对多楼层的建筑三维点云地图提取多层次的拓扑表示，以楼层-区域-子区域的多层拓扑结构表示多层建筑的室内环境。下面，分别对这四个方面进行总结，并展望接下来可能的研究方向。

(1) 第三章介绍了区域图的概念，以及从二维网格地图自动生成区域图并创建拓扑表示的算法。区域图的构建算法，主要是基于 Voronoi 图和使用 Alpha Shape 算法来把地图划分为有意义的区域（房间和走廊）。其主要贡献是提出了一种创新的拓扑表示，及提取拓扑结构的过程中提出了创新的地图分割方法。与其他分割方法的对比实验表明，区域图生成算法对复杂地图仍然有优于其他方法的分割结果。本方法用 Alpha Shape 检测算法有效地检测出了地图中的房间和走廊结构，使形成的区域更符合人类对区域划分的语义标准，有效地避免了对开阔区域的过分割。而通过考虑通道宽度（门宽）来设置房间检测参数  $\alpha$ ，有效地划分了不同区域，避免了分割不足的情况。本算法的地图分割结果很大程度取决于一个参数  $\alpha$ ，当门宽与走廊宽冲突时，或门宽宽于杂物间距时，仍然会产生过分割。后续的工作可以尝试使用多层次的区域检测算法，即设定多层  $\alpha$  参数，首先使用较大的  $\alpha$  检测出大片的开放区域，再使用较小的  $\alpha$  把其周围连通的小区域加入作为一个完整的区域。

(2) 第四章将区域图应用于路径规划。为此作者构建了区域图的对偶拓扑图 Passage Graph，预存穿梭于区域间的路径，大大加快了全局路径规划的计算。在未来的工作中，作者可以考虑将区域图应用于更多的算法，尤其是基于遍历和检

素的算法，充分发挥算法在拓扑地图上节点少、搜索空间小的优势。

(3) 第五章中，作者提出了基于区域特征匹配的两种地图匹配方法：启发式聚类法和生长邻居法，分别用于地图间有全局一致变换和无全局一致变换的地图匹配。在此章中，作者从区域中提取了各种可能用于测量区域相似性、且具有模态不变性的特征，并通过针对不同地图场景的实验充分探讨了各个特征对区域匹配正确率的影响，最终选择出了一些对区域相似性测量效果较好、对地图模态不敏感、计算耗时较短的特征用于区域匹配。在此之前并无通过区域特征进行拓扑地图的匹配的研究，因此，本文工作的一大贡献是提供了通过提取区域特征来用于拓扑地图匹配的新思路并取得了良好的效果。启发式聚类法的原理是通过区域匹配计算出可能的地图间变换，再通过对这些变换的聚类找出最能正确对齐两幅地图的变换。而生长邻居法作为一种补充方法用于损坏地图的匹配，其主要利用了拓扑地图的图结构，通过匹配区域的邻居生长出了两幅拓扑地图的公共子图。该方法结合了利用了区域的几何特征、结构特征和拓扑图结构，达到在无一致变换的地图间进行匹配的目的。本文的地图匹配方法由于是基于以区域为节点的拓扑地图的、基于区域匹配的，地图的匹配效果受限于区域分割效果，当两幅地图的对应区域分割不一致（例如，当地图中有过多且位置不同的杂物或区域不完整时容易发生）或地图间重合过少，就容易导致地图间对应区域过少从而匹配不成功的问题。未来的工作应进一步找到更多的基于环境固定结构的特征用于匹配，以缓解区域不一致或不完整导致的区域匹配不成功的问题。另外，各项特征在匹配中所占的权重参数，可通过优化或学习的方法调整成更加合理的设置。

(4) 第六章介绍了以多层的建筑点云为输入提取的多层次的拓扑表示，我们将其称为多层次拓扑-度量图。该方法提供多维度的输出，包括一个以区域为节点的拓扑图（0D，仅表示拓扑连接关系）、以空间中的点为节点的地图（1D）、二维区域分割（2D）、三维区域地图（3D，其节点是空间块）。形成的多层次拓扑-度量图包含四个层次，各层次（从高到低）的拓扑节点分别是：楼层、大区域、子区域、空间块。该拓扑图的构建的主要思想是将三维点云划分为各层级的区域，例如：通过检测天花板和地面划分楼层；通过检测门划分建筑中的房间和走廊；通过 Area Graph 生成算法将走廊按分叉划分；通过合并同顶高的空闲空间形成空间块。实验显示，该三维拓扑图构建过程中所涉及的区域分割方法，对

地图分割的准确性（与房间和走廊结构的吻合度）远优于对比实验中最近的二维分割算法。由于本方法的三维区域划分主要依赖于合并相似天花板高度下的空间，当天花板不平坦则可能产生区域过分割；当扫描点云不完整导致天花板不完整（存在洞坑）则其下方无法识别出空闲区域，会导致区域缺失；当天花板与门框高度相近，则无法识别门，导致房间与走廊被划分为同一个区域。基于这些问题，后续的工作应考虑增加点云填补算法，将上述不完整之处检测并填补，以提高算法分割区域的准确性和完整性。

总的来说，本文的主要贡献是提出创新的地图分割算法、以区域为节点构建拓扑表示、基于提出的拓扑表示提出创新的地图匹配算法。除了上面提及的后续工作要对现有工作进行的一些算法上的改进。在未来的工作中，作者将在以下方向开展工作：1) 根据地形（例如，马路和草坪）进行区域划分，构建园区内的室外拓扑地图，应用于地面机器人在大规模园区内的任务活动，且该室外拓扑地图可与建筑拓扑地图联合成为多层次的拓扑地图来表示完整园区。2) 拓展出线上的地图匹配算法，使机器人从任意区域出发，通过区域特征匹配确定机器人当前所在的区域，而非对比全局地图进行定位，使机器人在短时间内完成无初始猜测的全局定位。3) 将探索基于多层次拓扑-度量图的应用，尤其是可用于大规模的多层建筑室内应用，例如导航、定位、人机交互等。由于该拓扑图可用于三维空间，则它不仅可应用于地面机器人，亦可被用于空中机器人。

目前，拓扑地图的研究仍然未受到足够的关注和应用。自动生成以区域为节点的拓扑表示是一个创新的方向，仍有研究的空间，对拓扑地图的应用也仍需大量的研究。作者在此抛砖引玉，拓扑地图在机器人的应用中具有巨大的实用性，因此具有重要的研究意义，希望在未来的工作中，会有更多对拓扑地图构建和应用的研究工作。



## 参考文献

- 3DSOURCED. <https://www.3dsourced.com/rankings/best-3d-scanner/> [M]. On-line Resources, 2022.
- Alliez P, Giraudot S, Jamin C, et al. Point set processing [M/OL]. On-line Resources, 2022. [https://doc.cgal.org/latest/Point\\_set\\_processing\\_3/index.html](https://doc.cgal.org/latest/Point_set_processing_3/index.html).
- Armeni I, Sener O, Zamir A R, et al. 3d semantic parsing of large-scale indoor spaces [C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016: 1534-1543.
- Aurenhammer F, Klein R. Voronoi diagrams. [J]. Handbook of computational geometry, 2000, 5 (10): 201-290.
- Aurenhammer F, Klein R, Lee D T. Voronoi diagrams and delaunay triangulations [M]. World Scientific Publishing Company, 2013.
- Badino H, Huber D F, Kanade T, et al. Real-time topometric localization [M]//2019 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2012.
- Baker S, Matthews I. Lucas-kanade 20 years on: A unifying framework [J]. International journal of computer vision, 2004, 56(3): 221-255.
- Barnard S T, Fischler M A. Computational stereo [J]. ACM Computing Surveys (CSUR), 1982, 14 (4): 553-572.
- Beeson P, Jong N K, Kuipers B. Towards autonomous topological place detection using the extended voronoi graph [C]//IEEE International Conference on Robotics and Automation. 2005: 4373-4379.
- Birk A. A quantitative assessment of structural errors in grid maps [J]. Autonomous Robots, 2010, 28: 187-196.
- Birk A, Carpin S. Merging occupancy grid maps from multiple robots [J]. Proceedings of the IEEE, 2006, 94(7): 1384-1397.
- Birk A, Pathak K, Vaskevicius N, et al. Surface representations for 3d mapping [J]. KI-Künstliche Intelligenz, 2010, 24(3): 249-254.
- Blochiger F, Fehr M, Dymczyk M, et al. Topomap: Topological mapping and navigation based on visual slam maps [C]//2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018: 1-9.
- Bormann R, Jordan F, Li W, et al. Room segmentation: Survey, implementation, and analysis [C]// 2016 IEEE International Conference on Robotics and Automation (ICRA). 2016: 1019-1026.

- Brönnimann H, Fabri A, Giezeman G J, et al. 2D and 3D linear geometry kernel [M/OL]. On-line Resources, 2022. [https://doc.cgal.org/latest/Kernel\\_23/index.html](https://doc.cgal.org/latest/Kernel_23/index.html).
- Brunskill E, Kollar T, Roy N. Topological mapping using spectral clustering and classification [C]// Ieee/rsj International Conference on Intelligent Robots and Systems. 2007: 3491-3496.
- Buschka P, Saffiotti A. A virtual sensor for room detection [C]//Ieee/rsj International Conference on Intelligent Robots and Systems. 2002: 637-642 vol.1.
- Canny J. The complexity of robot motion planning [J]. 1988, 13: 27-32.
- Carpin S. Fast and accurate map merging for multi-robot systems [J]. Autonomous Robots, 2008, 25(3): 305-316.
- Carpin S, Birk A. Stochastic map merging in rescue environments [C]//Robot Soccer World Cup. Springer, 2004: 483-490.
- Carpin S, Birk A, Jucikas V. On map merging [J]. Robotics and autonomous systems, 2005, 53(1): 1-14.
- Chen H, Yang Z, Zhao X, et al. Advanced mapping robot and high-resolution dataset [J]. Robotics and Autonomous Systems, 2020: 103559.
- Da T K F. 2D alpha shapes [M/OL]. On-line Resources, 2022. [https://doc.cgal.org/latest/Alpha\\_shapes\\_2/index.html](https://doc.cgal.org/latest/Alpha_shapes_2/index.html).
- Den Berg J V, Overmars M H. Using workspace information as a guide to non-uniform sampling in probabilistic roadmap planners [J]. The International Journal of Robotics Research, 2005, 24 (12): 1055-1071.
- Diosi A, Taylor G R, Kleeman L. Interactive slam using laser and advanced sonar [J]. 2005: 1103-1108.
- Durrant-Whyte H, Bailey T. Simultaneous localization and mapping: part i [J/OL]. IEEE Robotics Automation Magazine, 2006, 13(2): 99-110. DOI: [10.1109/MRA.2006.1638022](https://doi.org/10.1109/MRA.2006.1638022).
- Edelsbrunner H, Kirkpatrick D G, Seidel R. On the shape of a set of points in the plane [J]. Information Theory IEEE Transactions on, 1983, 29(4): 551-559.
- Ekvall, Staffan, Kragic, et al. Object detection and mapping for service robot tasks [J]. Robotica, 2007, 25(2): 175-187.
- ElevationMap.net. <https://elevationmap.net/> [M]. On-line Resources, 2022.
- Elfes A. Using occupancy grids for mobile robot perception and navigation [J/OL]. Computer, 1989, 22(6): 46-57. DOI: [10.1109/2.30720](https://doi.org/10.1109/2.30720).
- Fabrizi E, Saffiotti A. Augmenting topology-based maps with geometric information [J]. Robotics & Autonomous Systems, 2002, 40(2-3): 91-97.
- Fankhauser P, Hutter M. A universal grid map library: Implementation and use case for rough terrain navigation [M]//Robot Operating System (ROS). Springer, 2016: 99-120.

- Fankhauser P, Bloesch M, Hutter M. Probabilistic terrain mapping for mobile robots with uncertain localization [J]. *IEEE Robotics and Automation Letters*, 2018, 3(4): 3019-3026.
- Fischler M A, Bolles R C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography [J]. *Communications of the ACM*, 1981, 24(6): 381-395.
- Foroughi F, Wang J, Nemati A, et al. Mapsegnet: A fully automated model based on the encoder-decoder architecture for indoor map segmentation [J/OL]. *IEEE Access*, 2021, 9: 101530-101542. DOI: [10.1109/ACCESS.2021.3097893](https://doi.org/10.1109/ACCESS.2021.3097893).
- Fraundorfer F, Engels C, Nistér D. Topological mapping, localization and navigation using image collections [C]//2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. Ieee, 2007: 3872-3877.
- Frese U. A discussion of simultaneous localization and mapping [J]. *Autonomous Robots*, 2006, 20(1): 25-42.
- Friedman S, Pasula H, Fox D. Voronoi random fields: Extracting the topological structure of indoor environments via place labeling [C]//IJCAI 2007, Proceedings of the International Joint Conference on Artificial Intelligence, Hyderabad, India, January. 2007: 2109-2114.
- Garulli A, Giannitrapani A, Rossi A, et al. Mobile robot slam for line-based environment representation [C]//Proceedings of the 44th IEEE Conference on Decision and Control. IEEE, 2005: 2041-2046.
- Ge S S, Zhang Q, Abraham A T, et al. Simultaneous path planning and topological mapping (sp2atm) for environment exploration and goal oriented navigation [J]. *Robotics and Autonomous Systems*, 2011, 59(3-4): 228-242.
- Georgiou C, Anderson S, Dodd T. Constructing informative bayesian map priors: A multi-objective optimisation approach applied to indoor occupancy grid mapping [J]. *The International Journal of Robotics Research*, 2017, 36(3): 274-291.
- Geraerts R J, Overmars M H. Enhancing corridor maps for real-time path planning in virtual environments [J]. *Computer Animation & Social Agents*, 2008.
- Geraerts R. Planning short paths with clearance using explicit corridors [J]. 2010: 1997-2004.
- Grisetti G, Stachniss C, Burgard W. Improved techniques for grid mapping with rao-blackwellized particle filters [J]. *IEEE transactions on Robotics*, 2007, 23(1): 34-46.
- Guo L, Yang Q, Yan W. Intelligent path planning for automated guided vehicles system based on topological map [C]//2012 IEEE conference on control, systems & industrial informatics. IEEE, 2012: 69-74.
- Hale D H, Youngblood G M, Dixit P N. Automatically-generated convex region decomposition for real-time spatial agent navigation in virtual worlds [J]. 2008.

- Han T, Almeida J S, da Silva S P P, et al. An effective approach to unmanned aerial vehicle navigation using visual topological map in outdoor and indoor environments [J]. *Computer Communications*, 2020, 150: 696-702.
- Hart P E, Nilsson N J, Raphael B. A formal basis for the heuristic determination of minimum cost paths [J]. *IEEE Transactions on Systems Science and Cybernetics*, 1968, 4(2): 100-107.
- Hayne R, Luo R, Berenson D. Considering avoidance and consistency in motion planning for human-robot manipulation in a shared workspace [C]//2016 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2016: 3948-3954.
- He Z, Hou J, Schwertfeger S. Furniture free mapping using 3d lidars [C]//2019 IEEE International Conference on Robotics and Biomimetics (ROBIO). IEEE, 2019.
- He Z, Sun H, Hou J, et al. Hierarchical topometric representation of 3d robotic maps [J]. *Autonomous Robots*, 2021: 1-17.
- Hert S, Schirra S. 2D convex hulls and extreme points [M/OL]//CGAL User and Reference Manual. CGAL Editorial Board, 2022. [https://doc.cgal.org/4.11/Convex\\_hull\\_2/index.html#Chapter\\_2D\\_Convex\\_Hulls\\_and\\_Extreme\\_Points](https://doc.cgal.org/4.11/Convex_hull_2/index.html#Chapter_2D_Convex_Hulls_and_Extreme_Points).
- Hoff K E, Culver T, Keyser J, et al. Fast computation of generalized voronoi diagrams using graphics hardware [J]. 2000: 375-376.
- Holleman C, Kavraki L E. A framework for using the workspace medial axis in prm planners [J]. 2000, 2: 1408-1413.
- Horn B K. Closed-form solution of absolute orientation using unit quaternions [J]. *Josa a*, 1987, 4 (4): 629-642.
- Hou J, Yuan Y, Schwertfeger S. Topological area graph generation and its application to path planning [J]. arXiv preprint arXiv:1811.05113, 2018.
- Hou J, Kuang H, Schwertfeger S. Fast 2d map matching based on area graphs [C/OL]//2019 IEEE International Conference on Robotics and Biomimetics (ROBIO). 2019a: 1723-1729. DOI: [10.1109/ROBIO49542.2019.8961732](https://doi.org/10.1109/ROBIO49542.2019.8961732).
- Hou J, Yuan Y, Schwertfeger S. Area graph: Generation of topological maps using the voronoi diagram [C]//2019 19th International Conference on Advanced Robotics (ICAR). IEEE, 2019b: 509-515.
- Hou J, Yuan Y, He Z, et al. Matching maps based on the area graph [J]. *Intelligent Service Robotics*, 2022: 1-26.
- Howard A. Dataset ap\_hill\_07b [M/OL]//Radish: Robotics Research Datasets. DSpaceMIT, 2020. <https://dspace.mit.edu/handle/1721.1/62267>.
- Howard A, Parker L E, Sukhatme G S. Experiments with a large heterogeneous mobile robot team:

- Exploration, mapping, deployment and detection [J/OL]. The International Journal of Robotics Research, 2006, 25(5-6): 431-447. DOI: [10.1177/0278364906065378](https://doi.org/10.1177/0278364906065378).
- Hu M K. Visual pattern recognition by moment invariants [J/OL]. IRE Transactions on Information Theory, 1962, 8(2): 179-187. DOI: [10.1109/TIT.1962.1057692](https://doi.org/10.1109/TIT.1962.1057692).
- Huang W H, Beevers K R. Topological map merging [J]. The International Journal of Robotics Research, 2005, 24(8): 601-613.
- Huang Z, Leng J. Analysis of hu's moment invariants on image scaling and rotation [C/OL]// 2010 2nd International Conference on Computer Engineering and Technology: volume 7. 2010: V7-476-V7-480. DOI: [10.1109/ICCET.2010.5485542](https://doi.org/10.1109/ICCET.2010.5485542).
- Hughes N, Chang Y, Carlone L. Hydra: a real-time spatial perception system for 3d scene graph construction and optimization [J]. 2022.
- Kai M W, Stachniss C, Burgard W. Coordinated multi-robot exploration using a segmentation of the environment [C]//Ieee/rsj International Conference on Intelligent Robots and Systems. 2008: 1160-1165.
- Kakuma D, Tsuichihara S, Ricardez G A G, et al. Alignment of occupancy grid and floor maps using graph matching [C]//2017 IEEE 11th international conference on semantic computing (ICSC). IEEE, 2017: 57-60.
- Karavelas M. 2D voronoi diagram adaptor [M/OL]. On-line Resources, 2022. [https://doc.cgal.org/4.11/Voronoi\\_diagram\\_2/index.html](https://doc.cgal.org/4.11/Voronoi_diagram_2/index.html).
- Kavraki L E, Svestka P, Latombe J, et al. Probabilistic roadmaps for path planning in high-dimensional configuration spaces [J]. international conference on robotics and automation, 1996, 12(4): 566-580.
- Kuffner J J, Lavelle S M. Rrt-connect: An efficient approach to single-query path planning [C]// IEEE International Conference on Robotics and Automation, 2000. Proceedings. ICRA. 2002: 995 - 1001.
- Kuhn H W. The hungarian method for the assignment problem [J]. Naval research logistics quarterly, 1955, 2(1-2): 83-97.
- Latombe J C. Robot motion planning [J]. Communications on Pure & Applied Mathematics, 1991, 48(9): 1173-1186.
- Lebedeva O, Kripak M, Gozbenko V. Increasing effectiveness of the transportation network by using the automation of a voronoi diagram [J]. Transportation Research Procedia, 2018, 36: 427-433.
- Lowe D G. Distinctive image features from scale-invariant keypoints [J]. International journal of computer vision, 2004, 60(2): 91-110.
- Lu D. global\_planner - ros wiki [M/OL]. ROS Wiki, 2022. [http://wiki.ros.org/global\\_planner](http://wiki.ros.org/global_planner).
- Lucas B D, Kanade T. An iterative image registration technique with an application to stereo vision

- [C]//IJCAI' 81: Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1981: 674-679.
- Ma J, Qiu W, Zhao J, et al. Robust l2e estimation of transformation for non-rigid registration [J/OL]. IEEE Transactions on Signal Processing, 2015, 63: 1-1. DOI: [10.1109/TSP.2014.2388434](https://doi.org/10.1109/TSP.2014.2388434).
- Martin S R, Wright S E, Sheppard J W. Offline and online evolutionary bi-directional rrt algorithms for efficient re-planning in dynamic environments [C]//IEEE International Conference on Automation Science and Engineering. 2007: 1131-1136.
- Mielle M, Magnusson M, Lilienthal A J. A method to segment maps from different modalities using free space layout - maoris : Map of ripples segmentation [C]//IEEE International Conference on Robotics and Automation (ICRA). IEEE Press, 2017.
- Moravec H, Elfes A. High resolution maps from wide angle sonar [C/OL]//Proceedings. 1985 IEEE International Conference on Robotics and Automation: volume 2. 1985: 116-121. DOI: [10.1109/ROBOT.1985.1087316](https://doi.org/10.1109/ROBOT.1985.1087316).
- Mozos I M, Triebel R, Jensfelt P, et al. Supervised semantic labeling of places using information extracted from sensor data [J]. Robotics and Autonomous Systems, 2007, 55(5): 391-402.
- Mozos O, Stachniss C, Rottmann A, et al. Using adaboost for place labeling and topological map building [J/OL]. International Journal of Robotic Research - IJRR, 2005, 28: 453-472. DOI: [10.1007/978-3-540-48113-3\\_39](https://doi.org/10.1007/978-3-540-48113-3_39).
- Muller D E, Preparata F P. Finding the intersection of two convex polyhedra [J]. Theoretical Computer Science, 1978, 7(2): 217-236.
- Munkres J. Algorithms for the assignment and transportation problems [J]. Journal of the society for industrial and applied mathematics, 1957, 5(1): 32-38.
- Oleynikova H, Taylor Z, Siegwart R, et al. Sparse 3d topological graphs for micro-aerial vehicle planning [C/OL]//2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2018: 1-9. DOI: [10.1109/IROS.2018.8594152](https://doi.org/10.1109/IROS.2018.8594152).
- Otsu N. A threshold selection method from gray-level histograms [J]. IEEE transactions on systems, man, and cybernetics, 1979, 9(1): 62-66.
- Pehlivanoglu Y V. A new vibrational genetic algorithm enhanced with a voronoi diagram for path planning of autonomous uav [J]. Aerospace Science and Technology, 2012, 16(1): 47-55.
- Reem D. The geometric stability of voronoi diagrams with respect to small changes of the sites [C]//Proceedings of the twenty-seventh annual symposium on Computational geometry. 2011: 254-263.
- Rohnert H. Moving discs between polygons. [J]. 1988, 317: 502-515.
- Rosinol A, Abate M, Chang Y, et al. Kimera: an open-source library for real-time metric-semantic

- localization and mapping [C]//2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2020: 1689-1696.
- Rusu R B, Cousins S. 3D is here: Point Cloud Library (PCL) [C]//IEEE International Conference on Robotics and Automation (ICRA). Shanghai, China: IEEE, 2011a.
- Rusu R B, Cousins S. 3d is here: Point cloud library (pcl) [C]//2011 IEEE international conference on robotics and automation. IEEE, 2011b: 1-4.
- Saeedi S, Paull L, Trentini M, et al. Efficient map merging using a probabilistic generalized voronoi diagram [C]//2012 IEEE/RSJ international conference on intelligent robots and systems. IEEE, 2012: 4419-4424.
- Saeedi S, Paull L, Trentini M, et al. Group mapping: A topological approach to map merging for multiple robots [J]. IEEE Robotics & Automation Magazine, 2014, 21(2): 60-72.
- Saeedi S, Paull L, Trentini M, et al. Map merging for multiple robots using hough peak matching [J]. Robotics and Autonomous Systems, 2014, 62(10): 1408-1424.
- Santos L, Santos F N, Magalhães S, et al. Path planning approach with the extraction of topological maps from occupancy grid maps in steep slope vineyards [C]//2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC). IEEE, 2019: 1-7.
- Schwertfeger S, Birk A. Evaluation of map quality by matching and scoring high-level, topological map structures [C]//IEEE International Conference on Robotics and Automation (ICRA). IEEE Press, 2013.
- Schwertfeger S, Birk A. Map evaluation using matched topology graphs [J/OL]. Autonomous Robots, 2015: 1–27. DOI: [10.1007/s10514-015-9493-5](https://doi.org/10.1007/s10514-015-9493-5).
- Schwertfeger S, Birk A. Map evaluation using matched topology graphs [J]. Autonomous Robots, 2016, 40(5): 761-787.
- Schwertfeger S, Yu T. Matching paths in topological maps [C]//9th Symposium on Intelligent Autonomous Vehicles (IAV), IFAC. IFAC, 2016b.
- Setalaphruk V, Ueno A, Kume I, et al. Robot navigation in corridor environments using a sketch floor map [C]//IEEE International Symposium on Computational Intelligence in Robotics and Automation, 2003. Proceedings. 2008: 552-557 vol.2.
- Setalaphruk V, Ueno A, Kume I, et al. Robot navigation in corridor environments using a sketch floor map [C]//Proceedings 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation. Computational Intelligence in Robotics and Automation for the New Millennium (Cat. No. 03EX694): volume 2. IEEE, 2003: 552-557.
- Shahbandi S G, Magnusson M. 2d map alignment with region decomposition [J]. Autonomous Robots, 2019, 43(5): 1117-1136.

- The CGAL Project. CGAL user and reference manual [M/OL]. 5.5 ed. CGAL Editorial Board, 2022. <https://doc.cgal.org/5.5/Manual/packages.html>.
- Thrun S. Robotic mapping: A survey [M]. Morgan Kaufmann, 2002.
- Thrun S. Learning metric-topological maps for indoor mobile robot navigation [M]. Elsevier Science Publishers Ltd., 1998.
- Thrun S, Burgard W, Fox D. Probabilistic robotics. 2005 [J]. Massachusetts Institute of Technology, USA, 2005.
- Toll W V, Cook A F, Geraerts R. Navigation meshes for realistic multi-layered environments [C]// Ieee/rsj International Conference on Intelligent Robots and Systems. 2011: 3526-3532.
- Triebel R, Pfaff P, Burgard W. Multi-level surface maps for outdoor terrain mapping and loop closing [C]//2006 IEEE/RSJ international conference on intelligent robots and systems. IEEE, 2006: 2276-2282.
- Van Toll W, Triesscheijn R, Kallmann M, et al. A comparative study of navigation meshes [J]. 2016.
- Van Toll W G, Cook A F, Geraerts R. A navigation mesh for dynamic environments [J]. Computer Animation & Virtual Worlds, 2012, 23(6): 535-546.
- Wallgrün J O. Hierarchical voronoi-based route graph representations for planning, spatial reasoning, and communication [Z]. 2004.
- Wallgrün J O. Voronoi graph matching for robot localization and mapping [M]//Transactions on computational science IX. Springer, 2010: 76-108.
- Wan S, Zhao Y, Wang T, et al. Multi-dimensional data indexing and range query processing via voronoi diagram for internet of things [J]. Future Generation Computer Systems, 2019, 91: 382-391.
- Wang C, Ma H, Chen W, et al. Efficient autonomous exploration with incrementally built topological map in 3-d environments [J]. IEEE Transactions on Instrumentation and Measurement, 2020, 69 (12): 9853-9865.
- Yang K. Anytime synchronized-biased-greedy rapidly-exploring random tree path planning in two dimensional complex environments [J]. International Journal of Control Automation & Systems, 2011, 9(4): 750.
- Yuan Y, Schwertfeger S. Incrementally building topology graphs via distance maps [C]//2019 IEEE International Conference on Real-time Computing and Robotics (RCAR). IEEE, 2019.
- Yuan Y, Borrmann D, Hou J, et al. Self-supervised point set local descriptors for point cloud registration [J]. Sensors, 2021, 21(2): 486.
- Yue Y, Zhao C, Wen M, et al. Collaborative semantic perception and relative localization based on map matching [C]//2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020: 6188-6193.

Zheng T, Duan Z, Wang J, et al. Research on distance transform and neural network lidar information sampling classification-based semantic segmentation of 2d indoor room maps [J/OL]. *Sensors*, 2021, 21(4). <https://www.mdpi.com/1424-8220/21/4/1365>. DOI: 10.3390/s21041365.

Zivkovic Z, Bakker B, Krose B. Hierarchical map building and planning based on graph partitioning [C]//IEEE International Conference on Robotics and Automation. 2006: 803-809.



## 致 谢



## 作者简介及攻读学位期间发表的学术论文与研究成果

### 作者简介

#### 基本情况:

侯佳维, 女, 1994 年生, 广东省汕头市人。

#### 教育经历

2016.09-2022.08	中国科学院大学上海科技大学移动机器人实验室	博士研究生
2012.09-2016.06	西南交通大学信息学院自动化专业	本科
2018.08-2019.02	瑞典厄勒布鲁大学 AASS 实验室	交流访问

#### 已发表 (或正式接受) 的学术论文:

1. **Hou, J.**, Yuan, Y. and Schwertfeger, S., 2019, December. "Area graph: Generation of topological maps using the voronoi diagram." In 2019 19th International Conference on Advanced Robotics (ICAR) (pp. 509-515). IEEE.

2. **Hou, J.**, Kuang, H. and Schwertfeger, S., 2019, December. "Fast 2D map matching based on area graphs." In 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO) (pp. 1723-1729). IEEE.

3. He, Z., **Hou, J.** and Schwertfeger, S., 2019, December. "Furniture free mapping using 3D LiDARs." In 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO) (pp. 583-589). IEEE.

4. **Hou, J.**, Yuan, Y., He, Z. and Schwertfeger, S., 2022. "Matching maps based on the Area Graph." Intelligent Service Robotics, 15(1), pp.69-94.

5. He, Z., Sun, H., **Hou, J.**, Ha, Y. and Schwertfeger, S., 2021. "Hierarchical topometric representation of 3D robotic maps." Autonomous Robots, 45(5), pp.755-771.

6. Cai, J., **Hou, J.**, Lu, Y., Chen, H., Kneip, L. and Schwertfeger, S., 2020, November. "Improving CNN-based planar object detection with geometric prior knowledge." In 2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR) (pp. 387-393). IEEE.

7. Yuan, Y., Borrmann, D., **Hou, J.**, Ma, Y., Nüchter, A. and Schwertfeger, S., 2021. "Self-supervised point set local descriptors for point cloud registration." *Sensors*, 21(2), p.486.

8. Zhi, X., **Hou, J.**, Lu, Y., Kneip, L. and Schwertfeger, S., 2022. "Multical: Spatiotemporal Calibration for Multiple IMUs, Cameras and LiDARs." 已被 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE. 正式接受。

#### 参加的研究项目及获奖情况：

- 2020.09-2021.11 自动棉花采摘机器人（校企合作）  
上海科技大学创新创业大赛教师组二等奖
- 2021.03-2022.08 多机器人协作平台（CMU 合作项目）
- 2022 年 中国科学院大学三好学生