哈爾濱工業大學

毕业设计（论文）的外文文献翻译

原始资料的题目/来源：**Autonomous Robotic Exploration Based Autonomous Robotic Exploration Autonomous Robotic Exploration/IEEE Access**

翻译后的中文题目：基于边界点优化和多步路径规划的机器人自主探索

专　　业　机器人工程

学　　生　贾彦鹏

学　　号　**2191300311**

班　　号　**1913502**

指导教师　黄博

翻译日期　**2023.04.18**

外文文献的中文翻译

摘要：机器人对未知环境的自主探索是机器人智能化的关键技术。为了提高搜索效率，本文提出了一种基于边界点优化和多步路径规划的搜索策略。在边界点优化部分，提出了一种随机边界点优化(RFPO)算法，选择评价值最高的边界点作为目标边界点。综合考虑信息增益、导航成本和机器人定位精度，来确定边界点的评价函数。在路径规划部分，我们提出了一个多步探索策略。我们没有直接规划从机器人当前位置到目标边界点的全局路径，而是设置了局部探索路径步长。当机器人的运动距离达到局部探索路径步长时，我们重新选择当前最优边界点进行路径规划，以减少机器人走一些重复路径的可能性。最后，通过相关实验验证了该策略的有效性。

关键词：自主探索；随机边界点优化算法；边界点评价函数；多步路径规划

I.引言

机器人自主探索[1,2]是机器人领域的一个重要研究课题。主要目标是让机器人在有限时间且无需人工干预的情况下，获得最完整、最准确的环境地图。许多现有的地图探索策略都是基于边界[3-7,9]，边界定义为未知空间与已知空间的分界线。基于边界的探索策略的思想是引导机器人到未知区域完成探索任务，因此自主探索任务一般分为三个步骤:生成边界点、选择评价值最高的边界点、规划前往所选边界点的路径。

边界点的生成以基于边界的探索策略为前提。在现有的研究中，一些边界点生成算法是基于数字图像处理的边缘检测和区域提取技术[3,4]。为了提取边界边缘，必须对整个地图进行处理，随着地图的扩展，处理它将消耗越来越多的计算资源。这引起了对边界边缘有效检测的研究。Keidar 和 Kaminka[5]提出了一种只处理新的激光读取数据的边界检测算法。在 Senarathne 等人[6]的研究中，他提出了一种通过跟踪网格单元的中间变化来生成边界点的方法，并且只考虑更新的网格单元来进行最后的边界生成操作。Umari 和 Mukhopadhyay[8]将快速探索随机树(rapid-exploration Random Tree, RRT)算法应用于边界点生成。由于 RRT 算法的随机性，生成的边界点分布不均匀。

目标边界点的选择是有效探索的关键。以边界为基础的战略是由 Yamauchi[9]首先提出的。所使用的探索策略是识别当前地图中的所有边界区域，然后驱动机器人前往最近的边界点。这种方法对于探索任务有两个缺点。首先，它平等对待所有边界。其次，它仅限于一个信息来源:寻找新边界区域。因此提出了多种不同的边界点选择算法。Simmons 等[10]和 Moorehead 等[11]提出了结合信息增益和探索成本来选择目标边界点的边界点选择函数。Carlone 和 Lyons[12]使用混合整数线性规划(MILP)模型来获得自主探测的最优边界点。Mei 等[13]提出了一种基于方向信息选择机器人下一个探索目标边界点的算法。拉古纳大学和波恩大学[14]的团队提出了一种新的探索策略，通过考虑之前看到的环境来利用背景知识，从而做出更好的探索决策。Gautam 等人[15]使用 K-means

算法对边界点进行聚类，并使用轮盘赌方法将这些边界点分配给机器人。

在路径规划部分，Bircher 等人[16]和 Ellips 和 Hossein[17]提出了一种基于 RRT 算法生成机器人搜索路径的算法。Lauri 和 Ritala[18]将该问题描述为带有信息论目标函数的部分可观察马尔可夫决策过程(POMDP)，并应用带开环近似的正向模拟算法求解。通过对局部环境信息采样，进行正演模拟，计算出每条路径的不同信息增益，以确定探索路径的选择。Stachniss 等人[19]提出了一种算法，该算法使用高效的 Rao-Blackwellized 粒子滤波器来表示关于地图和姿态的后验。它权衡了执行动作的成本和预期的信息增益，并考虑了机器人沿着路径运动可能收集的传感器测量值。有时，机器人会回到它们之前去过的地方，以减少不确定性。在 Elhoseny 等人[20]的研究中，他提出了一种基于遗传算法(GA)的路径规划方法，以便在动态和复杂的有障碍物的环境中工作。在 Senarathne 等人[21]的研究中，他增加了传统的基于边界的探测策略，包括一个概率决策步骤，以决定在计划路径上进一步移动到下一个传感位置是否可取。如果该运动不理想，则取消该运动，并选择新的传感位置作为下一个传感任务。

在本文中，我们提出了一种基于边界点优化和多步路径规划的策略。为了得到最优边界点，我们提出了一种随机边界点优化算法。该算法对 RRT 算法生成的随机边界点进行优化。将该算法与边界点评价函数相结合，可以得到当前最优边界点。其中一个主要问题是如何对边界点进行评价。在本文中，我们考虑了三个因素：边界点的信息增益、导航成本和机器人定位精度。考虑到这些因素，我们定义了边界点评价函数来评价所有边界点。在机器人到最优边界点的路径规划过程中，提出了多步探索策略。根据地图大小，我们定义了一个局部探索路径步长。当机器人的运动距离达到局部探索路径步长时，重新计算当前最优边界点并重新选择进行探索。这可能会防止机器人选择一些重复的路径。

本文的组织结构如下。第二节介绍了最优边界点提取方法，包括边界点生成、边界点评价函数、RFPO 算法三部分。第三节介绍了多步探索策略。在第四节中，我们进行了相关的实验来验证我们策略的有效性。最后，在本文第五节中进行了总结。

II.最优边界点提取

A.边界点的生成

本文采用 SLAM 算法 GMapping[25,26]构建二维占用网格图。在占用网格图中，网格单元有三种状态:空闲、占用和未知。占用网格图中的边界点定义为划分自由网格和未知网格的分界线。我们使用 RRT[27]算法在地图中生成边界点。该算法的优点是生成树构建简单，可以快速遍历空间中未探索的区域，特别适合包含障碍物的系统。此外，对于封闭环境，RRT 算法具有完整性，可以保证机器人在自主探索过程中探索所有区域并构建完整的地图。

首先，对快速搜索随机树进行初始化。一旦树完成初始化，就插入机器人的当前

位置$p_{current}$作为树的根节点。我们用p_i表示边界点，其中i是下标，以区分不同的边界点。新的点p_{rand}在地图上随机生成。找出当前树中已经存在的且与新生成点最近的节点$p_{nearest}$。将这两点连接起来得到一条直线，沿着这条直线从点$p_{nearest}$移动η的长度到点p_{rand}得到一个新的点p_{new}。η表示树的生长速度。当η设得很大时，树会长得很快，但它长不到一些小角落。反之，当η值设置较小时，虽能到达拐角，但生成边界点的速度会降低。为了保持这两个因素之间的平衡，我们根据地图大小来设置η值。如果在$p_{nearest}$和p_{new}之间的直线上没有障碍，则将相应的点和边添加到树中。否则，生成的点将被丢弃。同时，如果新生成的点位于未知区域，则认为该点为边界点。它会被标记在地图上，然后我们停止这棵树的生长。将当前机器人的位置作为新的根节点，我们构建一个新的快速探索随机树来生成边界点。边界点生成示例如图 1 所示。

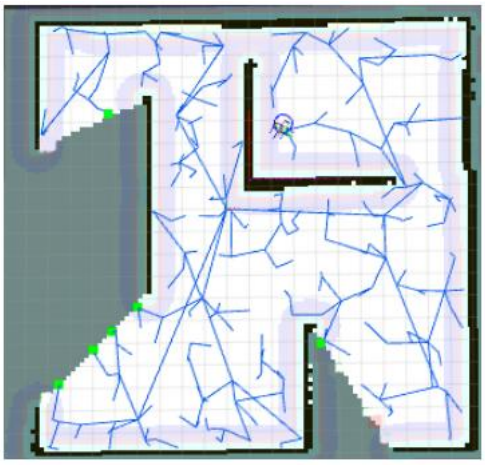

图 1 生成边界点。白色区域代表自由空间，灰色区域代表未知区域，黑色区域代表障碍，蓝色线代表生成树，绿色点代表边界点

B.边界点评价函数

边界点评价函数是边界点选择的依据。我们从边界点的信息增益、导航成本和机器人定位精度三个方面对边界点进行评估。在这些因素中，信息增益被定义为对于一个给定边界点预期被探索的未知区域面积。目前信息增益的计算方法可分为两类:一类是直接测量目标边界点可见区域内未检测到的空间大小[22]，另一类是基于信息熵法，该方法最早由 Bourgault[23,24]提出。本文采用第一种方法计算信息增益。以边界点为圆心，以激光雷达探测距离为半径形成圆。边界点检测圈如图 2 所示。通过计算圆圈中未知单元格的数量来量化信息增益。导航成本定义为机器人到达边界点的预期距离。计算机器人当前位置到目标边界点的欧氏距离来表示。此外，准确的地图还依赖于机

器人对自身姿态的准确估计。在目标边界点的探测范围内，如果能检测到更多的直线特征或其他特征(如断点、拐角、折线)，机器人就能更准确地定位自己。为了与信息增益的计算单位保持一致，我们用边界点检测圈内障碍物的面积来表示。通过计算圆圈中被占用的单元数来量化。基于以上因素，我们假设机器人在时间t内到达目标边界点p_i时，信息增益为I_t，导航代价为C_t，障碍物面积为F_t，边界点评价函数定义如下所示：

$$E(p_i(t)) = \frac{\alpha(I_t - I_{t-1}) + \gamma(F_t - F_{t-1})}{\beta(C_t - C_{t-1})} \tag{1}$$

其中，α、β、γ分别为信息增益、导航成本和障碍物面积的权重。这些权重用于调整不同因素的重要性。可根据不同的任务和环境进行设置。如果探索任务要求尽快完成探索，则增加α值，降低γ值;如果探索任务更注重地图的准确性，则减少α值，增加γ值。β通常取 1 表示单位导航成本下的增益值。利用边界点评价函数对所有边界点进行评价。选取值最大的点作为目标边界点。

C.随机边界点优化算法

由于边界点的生成部分始终运行在整个探测过程中，因此随着探测任务的执行，我们将得到许多边界点。然而，由于 RRT 算法的随机性，这些边界点的分布是不均匀的。因此，我们需要对生成的边界点进行优化。借鉴 GSO[29]算法的思想，提出 RFPO 算法。GSO 算法是一种新型的仿生群体智能优化算法。它模拟了高亮度萤火虫会吸引低亮度萤火虫向其移动的自然现象，使所有萤火虫集中在一个更好的位置，从而实现问题的优化。

在 RFPO 算法中，我们将每个边界点视为一只萤火虫，并将边界点评价函数$E(p_i(t))$的值作为其绝对亮度值$L_i(t)$：

$$L_i(t) = E(p_i(t)) \tag{2}$$

如果萤火虫i的绝对亮度值大于萤火虫j的绝对亮度值，则萤火虫j会被萤火虫i吸引并向其移动。这种吸引力的大小由萤火虫i对萤火虫j的相对亮度值决定，萤火虫i在萤火虫j所在位置的亮度强度定义为萤火虫i对萤火虫j的相对亮度，相对亮度值越大，吸引力越大。因此，为了对边界点i相对于边界点j的吸引力进行建模，首先必须对边界点i对边界点j的相对亮度进行建模。考虑到萤火虫的相对亮度随着距离的增加而减

小，定义边界点i对边界点j的相对亮度值为：

$$L_{ij}(t)=L_i(t)e^{-r_{ij}^2} \tag{3}$$

其中r_{ij}为边界点i到边界点j的距离。假设萤火虫i对萤火虫j的吸引力与萤火虫i对萤火虫j的相对亮度成正比，则边界点i对边界点j的吸引力可由(4)计算得到：

$$A_{ij}(t)=\rho L_{ij}(t) \tag{4}$$

其中ρ是吸引系数，对于所有边界点，不同的ρ值具有相同的影响，所以我们可以取$\rho=1$。

对于每个边界点，都有一个感知半径r。它的值应根据感知传感器的范围来设置。在这个范围内，每一个边界点都会找到绝对亮度值大于自己的其他边界点，形成自己的邻域集$N_i(t)$：

$$N_i(t)=\{r_{ij}\leqslant r;L_i(t)<L_j(t)\} \tag{5}$$

确定邻域集后，计算边界点移动到其邻域集中其他边界点的概率如下:

$$P_{ij}(t)=\frac{A_{ij}(t)}{\Sigma_{k\in N_i(t)}A_{ik}(t)} \tag{6}$$

为了在邻域集中选择一个要移动的目标点，我们使用了轮盘赌的方法。轮盘赌[30]是基于上述邻域集计算的概率值。一旦确定了目标边界点p_i，边界点p_i的运动由式(7)计算：

$$p_i(t+1)=p_i(t)+s\left(\frac{p_j(t)-p_i(t)}{\|p_j(t)-p_i(t)\|}\right) \tag{7}$$

其中$p_i(t)$、$p_j(t)$表示边界点的当前位置，$p_i(t+1)$为移动后边界点i的位置。s为运动步长，可根据传感器测量范围确定其值。设置迭代变量m，当优化迭代结束时，得到一个或多个局部收敛点。

边界点的生成和机器人的运动同时进行。当机器人的运动距离s_{move}移动小于确定的局部探索路径步长s_{fixed}时，RRT 算法生成边界点。当机器人的运动距离s_{move}达到确定的局部探索路径步长s_{fixed}时，我们删除不再处于边界的边界点和机器人无法到达的边界点。然后对所有边界点进行优化，从优化结果中选取评价值最高的边界点作为目

标边界点。优化效果见第四节图 5。算法 1 描述了提取最优边界点的整个过程：

Algorithm 1 *Optimal Frontier Points Extraction*

Input: map
Output: goalpoint

tree ← InitialzeTree (η)
tree ← InitialzeNode($P_{current}$,∅)
InitializeGSO(ρ,r,m,s)
while True **do**
  while $s_{move} < s_{fixed}$
    P_{rand} ← Random
    $P_{nearest}$ ← Nearest(tree, P_{rand})
    P_{new} ← Move($P_{nearest}$, P_{rand},η)
      if ObstacleFree($P_{nearest}$,P_{new}) **then**
        tree ← InsertNode(P_{new},($P_{nearest}$, P_{new}))
        if UnknowRegion(P_{new}) **then**
          PublishPoint(P_{new})
          tree ← Insert ($P_{current}$,∅)
        end if
      end if
  end while
  Clear Valid Points
  if Nember(p) = 0 **then**
    break
  end if
  set t=1
  for t ≤ m **do**
    for i = 1 to N **do**
      $L_i(t) = E(p_i(t))$
    end for
    for i = 1 to N **do**
      $N_i(t) = \{r_{ij} \leq r; L_i(t) < L_j(t)\}$
      for each $j \in N_i(t)$ **do**
        $L_{ij}(t) = L_i(t)e^{-r_{ij}^2}$
        $A_{ij}(t) = \rho L_{ij}(t)$
        $P_{ij}(t) = A_{ij}(t) / \Sigma A_{ik}(t)$
      end for
      $j = Roulette(P_{ik}(t), k \in N_i(t))$
      $p_i(t+1) = p_i(t) + s((p_j(t)-p_i(t))/|p_j(t)-p_i(t)|)$
    end for
    t = t + 1
  end for
  return goalpoint = Max(E(p))
end while

III.多步探索策略

路径规划对自主探索的效率也至关重要。在本文中，我们提出了一个多步探索策略。根据上述步骤获得当前最优目标边界点后，我们不直接规划从机器人当前位置到最优目标边界点的全局路径。相反，我们定义了一个用于多步路径规划的局部探索路径步长s_{fixed}。我们定义s_{fixed}的原因是：在机器人运动的过程中，会产生一些新的边界点，一些旧的边界点会失效，而新生成的边界点可能会优于当前的最优目标边界点。这可能会导致机器人选择一些重复的路径。因此，我们定义了一个局部探索路径步长。每次当机器人的运动距离达到步长时，我们都会清除无效点，并对所有剩余的边界点进行重新优化和重新选择，以避免这种情况的发生。在每个局部探索路径步长内，我们使用动态的方法[32]。采用动态窗口法进行机器人避障局部路径规划。

在动态窗口法中，为了根据给定的速度模拟出机器人相应的运动轨迹，需要知道机器人的运动模型。本文采用具有线速度和角速度的差动驱动机器人。设t时刻机器人位姿为$[x_t,y_t,\theta_t]^T$，给定速度对为$[v_t,\omega_t]^T$，窗口时间为Δt。那么计算模型如下：

$$x_{t+1}=x_t-\frac{v_t}{\omega_t}\sin\theta_t+\frac{v_t}{\omega_t}\sin(\theta_t+\omega_t\Delta t) \tag{8}$$

$$y_{t+1}=y_t-\frac{v_t}{\omega_t}\cos\theta_t+\frac{v_t}{\omega_t}\cos(\theta_t+\omega_t\Delta t) \tag{9}$$

$$\theta_{t+1}=\theta_t+\omega_t\Delta t \tag{10}$$

通过上述机器人运动模型，我们可以根据不同的速度计算出相应的运动轨迹。我们根据机器人自身的速度限制和环境约束，在一定范围内选择多组不同的速度，模拟机器人的运动轨迹。然后重新定义轨迹评估函数，对生成的多个轨迹进行评估，并选择得分最高的轨迹由机器人执行。考虑自主探索和地图构建过程中新的信息增益ΔI，新的障碍物观测面积ΔF，角偏差$\Delta\theta$，模拟姿态与目标姿态之间的距离Δl，模拟姿态与障碍物之间的最近距离Δd，我们定义了以下评价函数：

$$R(v,\omega)=\varepsilon\Delta I+\varphi\Delta F-\phi\Delta\theta-\mu\Delta l+\sigma\Delta d \tag{11}$$

其中ε、φ、ϕ、μ、σ是上述因素的权重。为了消除式(11)中不同计算的影响，每个部分必须先标准化。例如，我们将模拟姿态与障碍物之间的最近距离归一化，如下所示：

$$normal\_d(i)=\frac{\Delta d(i)}{\Sigma_{i=1}^{n}\Delta d(i)} \tag{12}$$

其中i表示要评估的轨迹，n是我们通过抽样生成的所有轨迹。

评价函数的物理意义是：在机器人的局部探索和导航过程中，要求机器人实时避开障碍物，以最快的速度到达目标边界点，并沿着运动轨迹探索更多未知区域。对于机器人来说，直线、断点、角点、折线等特征可以帮助它更准确地定位。因此，沿着机器人的轨迹可以观察到的特征越多，机器人就会越准确地定位自己，地图也就越准确。如下图 3 所示，黄色点为目标边界点。我们在不同的速度集上模拟了许多轨迹(只有一些轨迹在图 3 中显示)。根据我们定义的轨迹评价函数，我们选取得分最高的轨迹，如下图 3 中用红色标记。可以看出，机器人执行红色轨迹可以快速到达目标边界点。同时，轨迹与墙体有一定的安全距离，墙体边界可以帮助机器人更准确地定位自身。

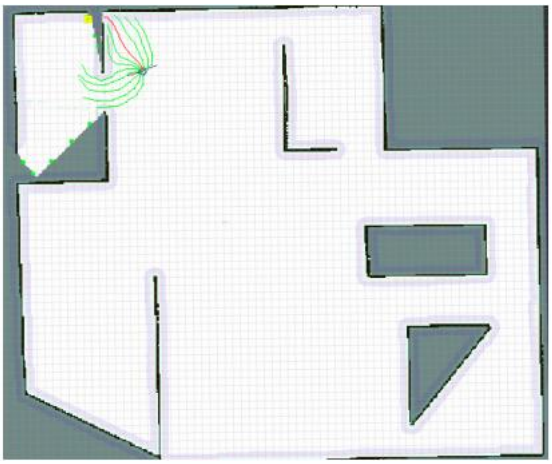

图 3 最佳探索轨迹的选择

IV.实验与结果

A.实验设置

通过仿真地图和真实地图对所提出策略的性能进行了实验验证。并与其他策略进行了比较。所有用于比较的策略都是在运行 Ubuntu 14.04 的 Intel core i7 3.60GHz 处理器和 8GB RAM 的计算机上使用 ROS 库在 c++中开发为 ROS 组件[33]。在我们的实验中，ROS 中的 gmapping 包是用来生成地图和定位机器人的。ROS 导航栈用于控制和指导机器人朝着探索目标前进。此外，系统参数如表 1 所示。

对于仿真环境，我们使用 Gazebo 模拟器[34]构建一个封闭空间，其中包含一些房间和障碍物，如下图 4 (a)所示。考虑到地图尺寸变化的影响，使用了不同的地图尺寸(20*20m, 40*40m, 60*60m)。机器人的半径为 0.2m，激光传感器的范围设置为 10m。图 4 (c)为在 20*20m 的模拟环境中建立的二维占用网格图。

在真实环境中，我们利用挡板构建了一个 10m * 10m 的空间，如下图 4 (b)所示。实验中使用的移动机器人平台为 EAIBOT Dashgo-D1。它配备了一个 Hokuyo UST-10LX 2D 激光传感器(10 米的检测范围和 270° 视野)。图 4 (d)为在真实环境中构建的二维占用网格图。

表 1 参数值

参数	取值	参数	取值
α	1	β	1
γ	0.5	ε	0.1
ϕ	0.1	Φ	0.2
μ	0.3	σ	0.3
ρ	1	r	5m
m	60	s	5m
$\eta_{(10*10)}$	1m	$\eta_{(20*20)}$	2m
$\eta_{(40*40)}$	4m	$\eta_{(60*60)}$	6m
$S_{fixed(10*10)}$	1m	$S_{fixed(20*20)}$	2m
$S_{fixed(40*40)}$	4m	$S_{fixed(60*60)}$	6m

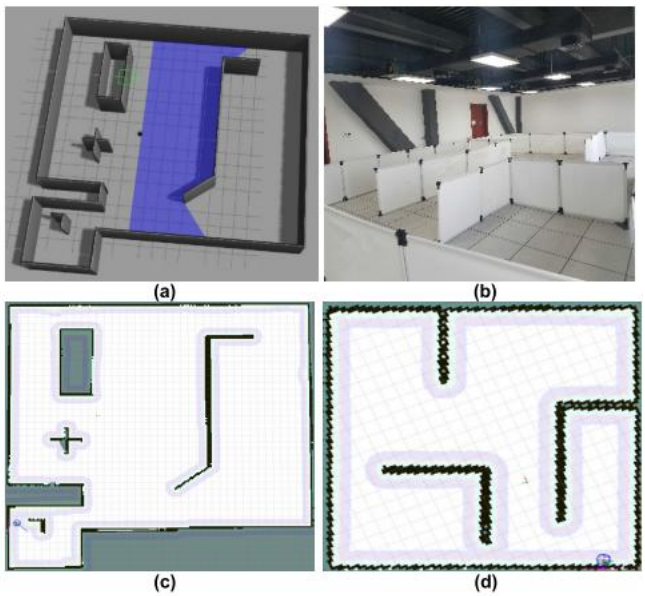

图 4 实验环境 (a)模拟环境 (b)真实环境

(c)为模拟环境生成的占用网格地图 (d)为真实环境生成占用网格地图

B.边界点优化结果

在进行自主机器人探索之前，先让机器人转一圈，得到一幅局部地图。在局部地图中，我们使用 RRT 算法生成边界点，如下图 5 (a)所示。由于 RRT 算法的随机性，边界点的位置都是随机的。可以看出，有的边界有很多边界点，有的边界只有很少的边界点。此外，在地图的各个边界上，边界点的分布也不均匀。图 5 (b)是用我们提出的算法生成的边界点。如图 5 (b)所示，优化后边界点数量大大减少，各边界上的边界点分布基本均匀。黄色点是根据我们定义的边界点评价函数计算出的当前情况下的最优边界点。

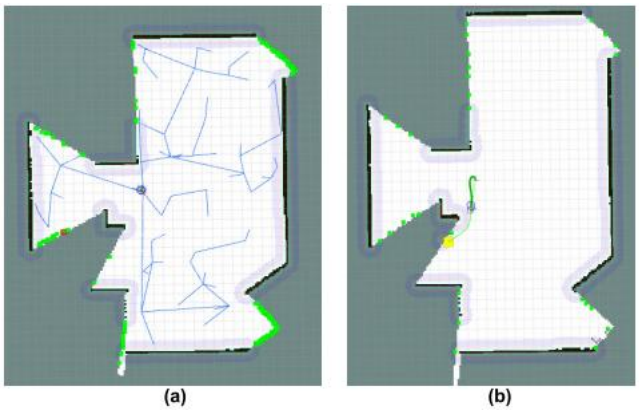

图 5 (a)优化前生成的边界点 (b)优化后生成的边界点

C.多步探索策略的结果

在本小节中，我们将比较不同路径规划策略的效果。如图 6 (a)和图 6 (b)所示，在传统的全局路径规划策略下，机器人直接规划从当前位置到目标边界点的路径，并搜索下一个目标边界点，直到到达前一个目标边界点。这种策略会导致一个问题：在探索过程中会产生新的边界点，这些边界点的位置可能比当前的目标边界点更好。机器人在到达前一个目标边界点后选择新的边界点进行探索，可能会导致机器人探索路径的重复，使得探索距离增加。然而，在多步路径规划策略中，每当机器人的运动距离达到确定的局部探索路径步长时，我们都会重新计算并重新选择最优边界点，以防止这种情况的发生。从下面的图 6 (c)可以看出，在机器人到达图 6 (a)中的目标边界点之前，当前的最优边界点已经发生了变化。因此，机器人已经规划了一条新的路径。结果是图 6 (c)的路径长度明显短于图 6 (a)和图 6 (b)。

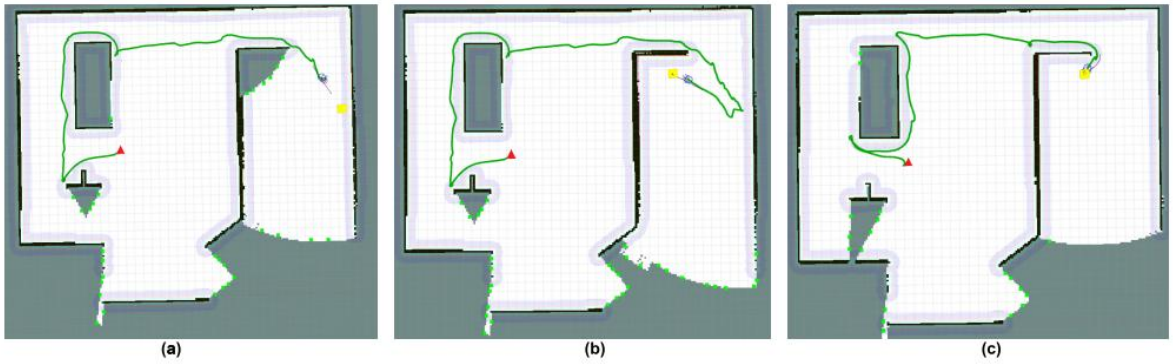

图 6 多步探索策略的结果，红色三角形表示起点，黄色点表示目标边界点。

(a)移动到目标边界点 (b)移动到下一个目标边界点 (c)移动到多步路径规划的目标边界。

D.与其他策略的比较

我们总共进行了 200 组实验，将我们提出的策略与其他四种策略进行比较。策略 1 的思想是随机选择边界点进行探索，我们称之为 RANDOM。策略 2 的思想是选择离机器人最近的边界点，我们用 NEAREST 来表示它。策略 3 采用了贪婪[35]算法的思想，因此本文将其记为 GREEDY。策略 4 是由 Umari 和 Mukhopadhyay[8]提出的，我们用 UMARI 来描述。我们在本文中提出的策略称为 RFPO。为了比较不同地图尺寸

对探索策略的影响，我们使用了 4 张不同尺寸的地图进行实验(一张真实地图和 3 张不同尺寸的模拟地图)。每张地图都要进行 50 组探索作业。这 50 次探索分为 5 组，每组代表一种探索策略。在 40 * 40 m 的模拟地图中，对于每种策略，我们从 10 次探索运行中选择一个实验结果来显示机器人的探索轨迹。结果如图 7 所示。

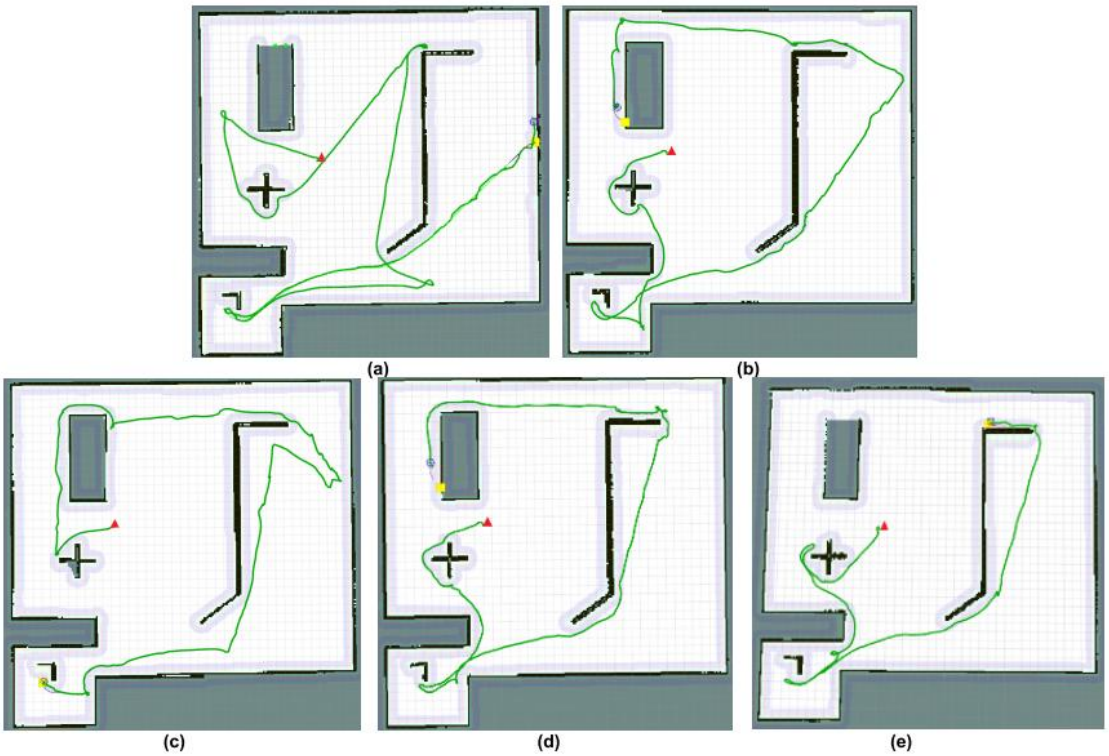

图 7 40*40 m 仿真图中各策略的探测轨迹 (a) RANDOM 策略的探测轨迹 (b) NEAREST 战略探索轨迹 (c) GREEDY 策略的探索轨迹 (d) UMARI 战略探索轨迹 (e) RFPO 战略的探索轨迹

我们利用 200 组实验数据对探测时间和探测距离进行统计，并绘制了以下图像。图 8 为四种不同地图中不同探索策略探索结束时的探索时间，图 9 为四种不同地图中不同探索策略探索结束时的探索距离。为了减少随机树的生长速度对实验结果的影响，我们根据不同的地图大小设置相应的生长速度。从图中我们可以看出，地图的尺寸越大，不同策略之间探索效率的差异就越明显。在 60*60m 的模拟地图上，我们提出的策略与其他四种策略相比，平均探索时间分别减少了 26.71%、7.36%、5.56%、1.62%，平均探索距离分别减少了 31.22%、15.56%、14.61%、8.43%。对于 RANDOM 策略来说，由于每次的目标边界点都是随机选择的，机器人会走很多重复的路线，所以探测时间和探测距离都会增加。而 NEAREST 策略和 GREEDY 策略会导致搜索变成局部最优问题，影响搜索的效率。UMARI 的策略直接规划了机器人从当前位置到探测目标点的路径，这可能会导致图 6 中的问题。在我们的探索策略中，我们对生成的随机边界点进行优化，在优化结果中选择评价值最大的点作为目标边界点。在路径规划部分，采用多步探索策略，降低机器人探索路径重复的概率。实验结果表明，无论是与探测时间相比，还是与探测距离相比，我们提出的探测策略的效果都优于其他

策略，证明了我们策略的有效性。

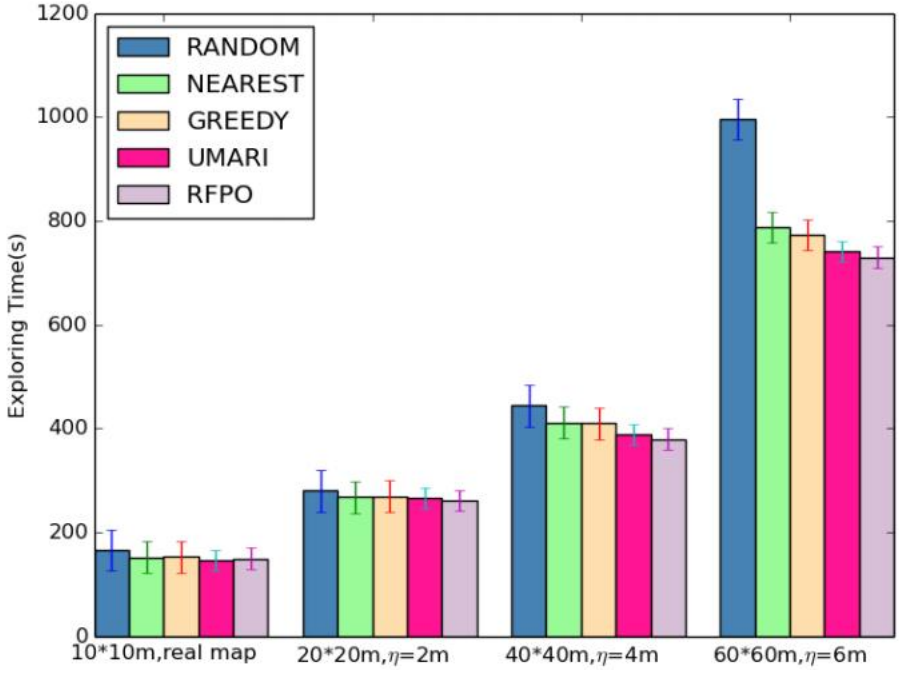

图 8 四种不同地图中不同探索策略探索结束时的探索时间

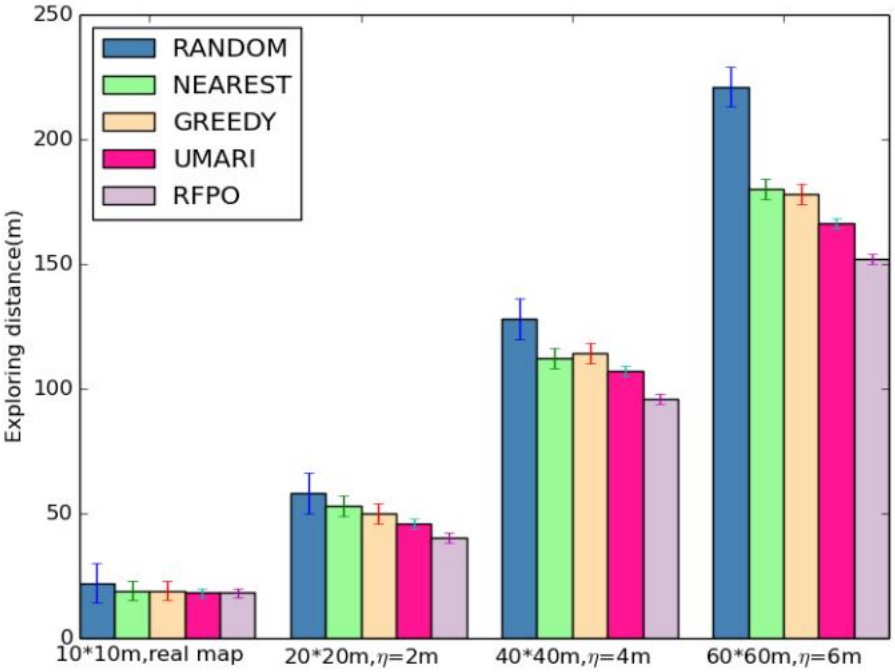

图 9 四种不同地图中不同探索策略探索结束时的探索距离

V.结论

本文提出了一种基于边界点优化和多步路径规划的机器人自主探索策略。该策略可以驱动机器人探索未知环境，并在无需人工干预的情况下高效地构建相应的二维占用栅格地图。在这个探索策略中，我们使用 RRT 算法来生成边界点，并提出 RFPO

算法来优化这些边界点。我们定义了边界点评价函数，选取当前最优边界点进行探索。在路径规划部分，我们设置一个局部探索路径步长，当机器人的运动距离达到局部探索路径步长时，重新选择目标边界点进行探索，以减少机器人走一些重复路径的可能性。我们分别在仿真环境和真实环境中进行了相关实验。实验结果验证了所提策略的有效性。由于我们目前只使用里程计数据结合激光传感器数据来构建二维占用网格地图，地图中包含的信息相对较少。接下来，我们打算将视觉传感器数据融合到自主探索中。视觉传感器数据的优点是可以获得更多的环境信息。这些数据可以被融合在一起，为以后的导航任务和其他相关工作构建具有更丰富信息的地图。此外，如何高效协调多个机器人[36]的探索也是我们准备做的一项重要任务。

参考文献

[1] V. Indelman, L. Carlone, and F. Dellaert, ‘‘Planning in the continuous domain: A generalized belief space approach for autonomous navigation in unknown environments,’’ Int. J. Robot. Res., vol. 34, no. 7, pp. 849–882, 2015.

[2] D. P. Ström, F. Nenci, and C. Stachniss, ‘‘Predictive exploration considering previously mapped environments,’’ in Proc. IEEE Int. Conf. Robot. Autom., May 2015, pp. 2761–2766.

[3] P. G. C. N. Senarathne and D. Wang, ‘‘Incremental algorithms for Safe and Reachable Frontier Detection for robot exploration,’’ Robot. Auton. Syst., vol. 72, pp. 189–206, Oct. 2015.

[4] M. Keidar and G. A. Kaminka, ‘‘Efficient frontier detection for robot exploration,’’ Int. J. Robot. Res., vol. 33, no. 2, pp. 215–236, 2014.

[5] M. Keidar and G. A. Kaminka, ‘‘Robot exploration with fast frontier detection: Theory and experiments,’’ in Proc. Int. Conf. Auton. Agents Multiagent Syst., 2013, pp. 113–120.

[6] P. G. C. N. Senarathne, D. Wang, Z. Wang, and Q. Chen, ‘‘Efficient frontier detection and management for robot exploration,’’ in Proc. IEEE Int. Conf. Cyber Technol. Automat., Control Intell. Syst., May 2013, pp. 114–119.

[7] A. Gautam, B. Jha, G. Kumar, J. K. Murthy, S. A. Ram, and S. Mohan, ‘‘FAST: Synchronous frontier allocation for scalable online multi-robot terrain coverage,’’ J. Intell. Robotic Syst., vol. 87, no. 3, pp. 545–564, 2017.

[8] H. Umari and S. Mukhopadhyay, ‘‘Autonomous robotic exploration based on multiple rapidly-exploring randomized trees,’’ in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Sep. 2017, pp. 1396–1402.

[9] B. Yamauchi, ‘‘A frontier-based approach for autonomous exploration,’’ in Proc. IEEE Int. Symp. Comput. Intell. Robot. Automat., (Cira), Jul. 1997, pp. 146–151.

[10] R. G. Simmons et al, ‘‘Coordination for multi-robot exploration and mapping,’’ in Proc. 17th Nat. Conf. Artif. Intell. 12th Conf. Innov. Appl. Artif. Intell., 2000, pp. 852–858.
[11] S. J. Moorehead, R. Simmons, and W. L. Whittaker, ‘‘Autonomous exploration using multiple sources of information,’’ in Proc. IEEE Int. Conf. Robot. Automat., vol. 3, May 2001, pp. 3098–3103.
[12] L. Carlone and D. Lyons, ‘‘Uncertainty-constrained robot exploration: A mixed-integer linear programming approach,’’ in Proc. IEEE Int. Conf. Robot. Autom., May/Jun. 2014, pp. 1140–1147.
[13] Y. Mei, Y.-H. Lu, C. S. G. Lee, and Y. C. Hu, ‘‘Energy-efficient mobile robot exploration,’’ in Proc. IEEE Int. Conf. Robot. Autom., May 2006, pp. 505–511.
[14] D. P. Ström, I. Bogoslavskyi, and C. Stachniss, ‘‘Robust exploration and homing for autonomous robots,’’ Robot. Auton. Syst., vol. 90, pp. 125–135, Apr. 2017.
[15] A. Gautam, J. K. Murthy, G. Kumar, S. P. A. Ram, B. Jha, and S. Mohan, ‘‘Cluster, allocate, cover: An efficient approach for multi-robot coverage,’’ in Proc. IEEE Int. Conf. Syst., Oct. 2016, pp. 197–203.
[16] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, ‘‘Receding horizon ‘next-best-view’ planner for 3D exploration,’’ in Proc. IEEE Int. Conf. Robot. Autom., May 2016, pp. 1462–1468.
[17] E. Masehian and H. Kakahaji, ‘‘NRR: A nonholonomic random replanner for navigation of car-like robots in unknown environments,’’ Robotica, vol. 32, no. 7, pp. 1101–1123, 2014.
[18] M. Lauri and R. Ritala, ‘‘Planning for robotic exploration based on forward simulation,’’ Robot. Auto. Syst., vol. 83, pp. 15–31, Sep. 2016.
[19] C. Stachniss, G. Grisetti, and W. Burgard, ‘‘Information gain-based exploration using rao-blackwellized particle filters,’’ Robot., Sci. Syst., vol. 2, pp. 65–72, Jun. 2005.
[20] M. Elhoseny, A. Shehab, and X. Yuan, ‘‘Optimizing robot path in dynamic environments using genetic algorithm and Bezier curve,’’ J. Intell. Fuzzy Syst., vol. 33, no. 4, pp. 2305–2316, 2017.
[21] P. G. C. N. Senarathne, ‘‘Frontier based exploration with task cancellation,’’ in Proc. IEEE Int. Symp. Saf., Secur., Rescue Robot., Oct. 2014, pp. 1–6.
[22] H. H. González-Baños and J. C. Latombe, ‘‘Navigation strategies for exploring indoor environments,’’ Int. J. Robot. Res., vol. 21, no. 10, pp. 829–848, 2001.
[23] A. A. Makarenko, S. B. Williams, F. Bourgault, and H. F. Durrant-Whyte, ‘‘An experiment in integrated exploration,’’ in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., vol. 1, Sep./Oct. 2002, pp. 534–539.

[24] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, ‘‘Information based adaptive robotic exploration,’’ in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., vol. 1, Sep./Oct. 2002, pp. 540–545 [25] G. Grisettiyz, C. Stachniss, and W. Burgard, ‘‘Improving grid-based SLAM with rao-blackwellized particle filters by adaptive proposals and selective resampling,’’ in Proc. IEEE Int. Conf. Robot. Autom., Apr. 2005, pp. 2432–2437.

[26] Grisetti, C. Stachniss, and W. Burgard, ‘‘Improved techniques for grid mapping with rao-blackwellized particle filters,’’ IEEE Trans. Robot., vol. 23, no. 1, pp. 34–46, Feb. 2007.

[27] S. M. LaValle, ‘‘Rapidly-exploring random trees: A new tool for path planning,’’ Dept. Comput. Sci., vol. 98, no. 11, Oct. 1998.

[28] K. N. Krishnanand and D. Ghose, ‘‘Detection of multiple source locations using a glowworm metaphor with applications to collective robotics,’’ in Proc. IEEE Swarm Intell. Symp. (SIS), Jun. 2005, pp. 84–91.

[29] K. N. Krishnanand and D. Ghose, ‘‘Multimodal Function Optimization using a Glowworm Metaphor with Applications to Collective Robotics,’’ in Proc. Indian Int. Conf. Artif. Intell., Dec. 2005, pp. 328–346.

[30] Internet. Roulette. Accessed: Mar. 8, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Roulette [31] D. Fox, W. Burgard, and S. Thrun, ‘‘The dynamic window approach to collision avoidance,’’ IEEE Robot. Autom. Mag., vol. 4, no. 1, pp. 23–33, Mar. 1997.

[32] M. Seder and I. Petrovic, ‘‘Dynamic window based approach to mobile robot motion control in the presence of moving obstacles,’’ in Proc. IEEE Int. Conf. Robot. Autom., Apr. 2007, pp. 1986–1991.

[33] M. Quigley et al, ‘‘ROS: An open-source robot operating system,’’ in Proc. ICRA Workshop Open Source Softw., 2009, pp. 1–6.

[34] Internet. Gazebo simulator. Accessed: Mar. 8, 2019. [Online]. Available: http://gazebosim.org/ [35] F. Amigoni and A. Gallo, ‘‘A multi-objective exploration strategy for mobile robots,’’ in Proc. IEEE Int. Conf. Robot. Automat., Apr. 2005, pp. 3850–3855.

[36] A. Q. Li, R. Cipolleschi, M. Giusto, and F. Amigoni, ‘‘A semantically-informed multirobot system for exploration of relevant areas in search and rescue settings,’’ Auton. Robots, vol. 40, no. 4, pp. 581–597, 2016.

外文文献的原稿