DOI:10.19344/j.cnki.issn1671-5276.2019.04.030

基于区域自治的多 RGV 分布式动态路径规划算法

王泽坤,罗福源,袁启虎
(南京航空航天大学 机电学院,江苏 南京 210016)

摘　要:针对医院轨道物流传输系统中医用多 RGV 调度系统路径规划问题,提出了一种基于区域自治的分布式动态路径规划算法。算法在通过区域划分构建区域自治地图模型的基础上,应用两阶段动态路径规划策略解决区域内路径规划问题,同时通过引入边缘节点概念,结合启发式策略,解决跨区域路径规划问题。仿真结果证明,该算法较传统集中式动态路径规划算法,可显著提高路径规划的效率。

关键词:医院轨道物流传输系统;动态路径规划;区域自治;分布式;启发式策略
中图分类号:TP391.9　　**文献标志码**:A　　**文章编号**:1671-5276(2019)04-0110-06

Distributed Dynamic Path Planning Algorithm for Multi-RGV Based on Regional Autonomy

WANG Zekun, LUO Fuyuan, YUAN Qihu
(College of Mechanic and Electronic Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

Abstract: To solve the routing problem for the multiple rail guided vehicle (RGVs) in medical rail logistic system, a distributed dynamic route planning algorithm based on regional autonomy is proposed. In our approach, the regional autonomy map model is built by region partition. Two-stage dynamic path planning algorithm is applied to solving the regional routing problem. Meanwhile, the heuristic strategy combined with the introduced concept of edge node is used to solve the interregional routing problem. The simulation results show that this algorithm can be used to improve the efficiency of the path planning.

Keywords: medical rail logistic system; dynamic path planning; regional autonomy; distribution; heuristic strategy

0　引言

医用 RGV(rail guided vehicle)作为轨道物流传输系统的运输载体,是一种运行在特定的医院物流轨道上,将物资从起始站输送至目的站的物流传输设备,在现代化医院物流领域应用前景巨大。目前医用轨道物流传输系统的发明和历史已接近 40 年,在国外的发展及应用已相对成熟。国际上有包括西门子、特立等多个品牌的医用轨道系统提供商。而在国内,由于应用时间相对较短,因此针对该类系统研究较少,大多数研究主要集中在系统的组成介绍、特性分析、使用场景、常见故障等方面,而针对医用多 RGV 调度系统尤其是路径规划问题的研究几乎为空白[1-3]。

在与多 RGV 路径规划问题相类似的多 AGV(automated guided vehicle)路径规划问题研究领域中,由于 AGV 应用相对更为广泛,国内外学者进行了许多相关研究。其中 C.W.Kim[4] 等针对 AGV 双向路径规划,在 1991 年提出了一种无冲突最短时间路径规划算法,该算法可在考虑系统内其他 AGV 状态的情况下,为当前 AGV 规划出一条运行时间最短且无碰撞的运行路径。但是该算法无法针对突发状况做出有效处理。Jung Hong Lee[5] 等人针对多 AGV 路径规划在 1998 年提出了一种两阶段控制策略,主要思想是先离线生成路径表,再进行在线路径规划,从而降低系统运算量。国内的刘国栋[6] 等人在此基础上提出了一种两阶段动态路径规划算法,为降低运算量,引入了阻塞最大门限划定搜索深度;王家溶[7] 等同样在两阶段控制策略的基础上,提出了一种当离线阶段生成的 k 条最短路径均不满足要求时,基于带约束的遗传算法进行重选的方法,并引入了速度调节的冲突调节模式;胡彬[8] 等人针对基于时间窗分布的动态路径规划方法作了详细描述,并在仿真实验和真车环境下作了实验测试;张峥炜[9] 等人将时间窗理论与 Dijkstra 算法相结合,实现了自动化码头多 AGV 无冲突且行驶时间最短的动态路径规划。但这些算法均存在计算时间随着节点数目增多、路径长度增大和 AGV 数量增加而指数爆炸,从而导致实时性较差的问题。

为解决这个问题,本文在传统集中式动态路径规划算法的基础上,通过区域划分及引入启发式策略,提出了一种基于区域自治的分布式动态路径规划算法。主要包括以下 3 个阶段。

初始化阶段:首先针对全局地图进行区域划分,并引入区域内边缘节点的概念,从而生成带区域 ID 属性的区域自治地图模型,并将该模型存储至各区域控制器中。

离线规划阶段:各区域控制器生成本区域内各节点间

基金项目:江苏省优秀青年基金项目(BK20160084);中央高校基本科研业务费项目(NS2016056);江苏省科技支撑计划项目(BE2014112)

作者简介:王泽坤(1992—),男,山东菏泽人,硕士研究生,研究方向为数控技术及其应用。

的所有可行路径备选集合,并计算出各区域内边缘节点到其他区域各节点间的理想最短行驶时间。

在线规划阶段:在各个区域内部,如有新路径规划任务生成,则各区域控制器基于先到先服务原则及任务优先级,依次为区域内所有 RGV 自当前位置到目标终点重新分配最优路径。其中,对于终点在本区域内的路径规划任务,该路径为完整路径;对于终点在本区域外的路径规划任务,该路径为到下一区域边缘节点的部分路径。

1　问题描述

1.1　RGV 运行地图描述

在医用轨道物流传输系统中,RGV 沿着医院内架设的特定专用轨道行驶,并通过连接两条不同轨道的转轨器实现轨道转换,从而完成在物流工作站间的物资搬运任务。轨道及连接轨道的转轨器以及物流工作站的整体布局便构成了 RGV 的运行地图。本文选择基于图论的方式对 RGV 的运行地图进行建模,将转轨器及物流工作站统一视为资源节点,用图 $G=\{V,E\}$ 表示整个系统的网络拓扑结构。其中 $V=\{v_1,v_2,\cdots,v_n\}$ 代表资源节点(顶点)的集合。而 $E=\{e_{ij}=(v_i,v_j)\mid(v_i,v_j\in V)\wedge(\omega(e_{ij})>0),e_{ij}\in E\}$ 为节点间连接轨道(边)的有权重集合。$\omega(e_{ij})$ 是边 e_{ij} 的权重,由公式 $\omega(e_{ij})=L(e_{ij})/sp$ 确定,sp 代表车辆在该轨道边上的平均行驶速度,$L(e_{ij})$ 代表边的长度。

1.2　区域划分及区域自治地图模型构建

合理的区域划分要能确保区域间无交集且区域并集等于原环境。为探讨区域间路径规划问题,特别引入边缘节点及连接边的概念,由于本文所研究的医院轨道物流传输系统节点是按楼层分布的,在每个楼层内部节点分布众多,而在楼层间则只有若干个连接的轨道边。因此可依据这个特点将原轨道物流传输系统划分为多个自治区域(autonoumous ystem, AS)。这里针对划分后的自治区域做出如下定义:

定义自治区域 AS_k,内节点集合 $E_{IN}^{AS_k}$ 及内连接边集合 $V_{IN}^{AS_k}$ 满足:

$$E_{IN}^{AS_k}=\{e_{uv}=(u,v)\mid(u\in AS_k)\wedge(v\in AS_k),e_{uv}\in E_{IN}^{AS_k}\}\quad(1)$$

$$V_{IN}^{AS_k}=\{v\mid\forall e_i\in adj(v),\text{s.t},e_i\in AS_k,v\in V_{IN}^{AS_k}\}\quad(2)$$

定义自治区域 AS_k 外连接边集合 $E_{OUT}^{AS_k}$ 及边缘节点集合 $V_{OUT}^{AS_k}$ 满足:

$$E_{OUT}^{AS_k}=\{e_{uv}=(u,v)\mid(u\in AS_k)\wedge(v\notin AS_k),e_{uv}\in E_{OUT}^{AS_k}\}\quad(3)$$

$$V_{OUT}^{AS_k}=\{v\mid\exists e_i\in adj(v),\text{s.t},e_i\notin AS_k,v\in V_{IN}^{AS_k}\}\quad(4)$$

式中 $adj(v)$ 表示节点 v 的邻接边集合,邻接边是指与节点 v 存在直接联系的边。为简化算法流程作如下约定:

1) 两节点间由两条方向相反的单向道连接。

2) 对于与自治区域 AS_k 存在直接连通关系的下一区域的任意边缘节点,在当前区域的边缘节点集合中存在唯一边缘节点与之对应。

则自治区域 AS_k 的图为:

$$G_k=\{V_k,E_k\}=\{V_{IN}^{AS_k}\cup V_{OUT}^{AS_k},E_{IN}^{AS_k}\cup E_{OUT}^{AS_k}\}$$

同时所有自治区域图的并集可复原全局地图。如图 1 所示为利用点和线抽象出的简易 RGV 区域自治地图模型。圆圈内数字代表节点号,有向箭头上数字代表相应边的权重。据图分析可知,原 RGV 运行地图被划分成了两个自治区域 AS1 及 AS2,且 AS1 的边缘节点为 4 和 6,AS2 的边缘节点为 7 和 9。

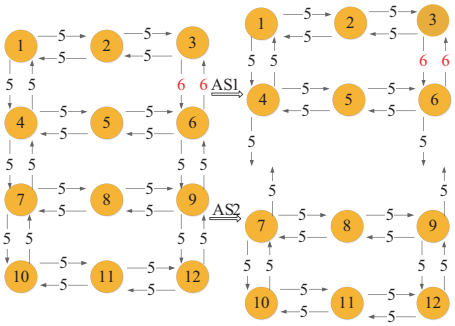


图 1　多 RGV 运行区域自治地图模型

1.3　多 RGV 路径规划任务描述

类比多 AGV 动态路径规划定义[10],多 RGV 动态路径规划问题可定义为基于系统实时运行状态,为系统内多个 RGV 从各自起点到终点寻找无碰撞最优行驶路径,从而可将医用多 RGV 动态路径规划问题归纳为以下 3 个方面的子问题:1) 确定从任务起点到任务终点之间是否存在连通的轨道;2) 规划出的轨道必须是可行的,即无阻塞、无冲突、无死锁的;3) 规划出的轨道必须是最优的,或者是部分最优的,从而使得整个系统的 RGV 总体运行时间达到最优。

为方便之后讨论,定义多 RGV 系统内一个动态路径规划任务形式为:

$$M_R=(src,dst,t_{\text{start}},V,ID)\quad(5)$$

式中:src 为 RGV 任务起始点;dst 为任务终点;t_{start} 为任务起始时间;V 为执行 RGV 的编号;ID 为路径规划任务编号。由于在 RGV 所建立的区域地图模型中,RGV 的起始位置可能在调度节点上,也可能在连接轨道边上,因此针对 src 及 t_{start} 做出如下规定:

若 RGV 位于连接边上,则 src 为当前单向连接边的终点,t_{start} 满足:

$$t_{\text{start}}=t_{\text{cur}}+\omega_{\text{left}}\quad(6)$$

式中:t_{cur} 代表当前时间,ω_{left} 代表 RGV 自当前位置到下一节点的剩余权值。

若 RGV 位于节点上,则 src 为当前节点,t_{start} 满足:

$$t_{\text{start}}=t_{\text{cur}}\quad(7)$$

2　算法描述

2.1　路径备选集及边缘节点理想最短行驶时间求解算法

1) 基于 DFS 算法的备选路径集求解算法

由于经过区域划分后的各自治区域地图内部节点数

目及连接边数目相对较少,因此考虑算法的易实现性,采用深度优先搜索(DFS)算法生成自治区域内各节点 a 至其他任意节点 b 间可行路径备选集合 P_a^b,则针对子区域 AS_k 主要步骤为:

步骤 1:设定起始点 $V0$,$V0$ 属于 AS_k,采用 DFS 算法,求得 $V0$ 到区域 AS_k 内任意节点的所有备选路径集合,并按照路径长度升序排列。

步骤 2:遍历区域 AS_k 内所有节点为 $V0$,并存储在 AS_k 的路径库。

为存储区域内各点到其他节点的备选路径集合,首先需要定义备选路径集合存储数据结构。借助 C++标准模板 STL(standard template library)中的动态数组容器 vector 及关联容器 map,以 vecor<int>存储单条路径,以 vector<vector<int>>存储路径集合,以 map<int, vector<vector<vector>>> 将该路径集合关联到由 int 确定的节点号上。定义节点结构体 VEXDATA,则 map<int, vector<vector<vector>>> PreparePath 为其中的备选路径集合成员变量。

2) 基于 Dijkstra 算法的边缘节点理想最短行驶时间求解算法

考虑到全局地图节点数目较多,且实用中的 RGV 运行地图为稀疏图,为提高算法效率,选择 Dijkstra 算法用于求解理想最短行驶时间,设 $path_{a,b}$ 为节点 a 至节点 b 的最短路径,并定义自节点 a 至节点 b 的理想最短行驶时间为 $Th_{a\,\mathrm{image}}^{b}$,则有

$$T_{a\,\mathrm{image}}^{b}=T_{\mathrm{edge}}+T_{\mathrm{vertex}} \tag{8}$$

式中:T_{edge} 代表 $path_{a,b}$ 中所有边的的权值和,T_{vertex} 代表 $path_{a,b}$ 中所有节点总的消耗时间。在实际 RGV 运行中,各转轨器节点消耗时间通常为常值 C_{cost},因此该值的计算形式为:

$$T_{vertex}=n\times C_{\mathrm{cost}} \tag{9}$$

式中 n 代表路径 $path_{a,b}$ 中转轨器节点数目。

对于 AS_k,其边缘节点到全局地图所有节点的理想最短行驶时间的求解流程如下:

步骤 1:设定起始边缘节点 $V0$,$V0$ 属于式(4)所确定的 AS_k 的边缘节点集合,采用经典 Dijkstra 算法,求得 $V0$ 到全局地图内任意其他节点的最短路径序列。并按照式(8)和式(9)计算对应的 T_{image}。

步骤 2:遍历 AS_k 内所有边缘节点为 $V0$,求得所有对应的 T_{image}。

2.2　基于节点时间窗的区域内 RGV 动态路径规划算法

在多 AGV 系统中,由于在边的连接处时间耗费相对较小,人们常采用基于边排布的时间窗排布算法确定动态路径。这种方法通常需要规定单条边上同时只能存在一台车。但实际应用中,由于有的边长度较长,若同时只有一台车占用,将会降低系统内车辆运行效率。为改进这种情况,可以通过在实际节点之间引入虚拟辅助节点解决。但这样又会增大地图密度,降低算法效率。

针对多 RGV 运行过程中将会在转轨器处出现时间占用的现象,本文提出了基于节点时间窗排布的方法。

离线阶段:对于 AS_k,G_k 为其对应图,首先通过前文所提及的备选路径集求解算法求 G_k 内各节点到其他节点的备选路径集合,同时 AS_k 控制器内保存了各节点已排布好的时间窗序列。定义节点上单个时间窗模型为:

$$T_W^{\,i}=(tin_i,tout_i) \tag{10}$$

式中:tin_i 代表节点 v_i 被 RGV 进入该节点起始时间;$tout_i$ 代表 RGV 离开该节点的时间。

实际应用中采用链表 list<TimeWindow>存储该节点上已排布的所有时间窗序列。Timewindow 为单个时间窗模型存储结构。

则对于单条路径序列 path 中的各个节点 v_i,时间窗在各节点初始化原则定义如下:

若节点 v_i 为起始节点,则 RGV 进入节点时间 tin_i 等于任务开始时间 t_{start},若节点 v_i 不是起始节点,则进入节点时间 tin_i 等于 RGV 离开第 $i-1$ 个节点的时间加边 $e_{i-1,i}$ 的权值。RGV 离开节点时间 $tout_i$ 始终等于 tin_i 加上 RGV 在该节点预计停留时间,即

$$tin_i=t_{\mathrm{start}},i=0 \tag{11}$$

$$tin_i=tout_{i-1}+\omega(e_{i-1,i}),i\geqslant 1 \tag{12}$$

$$tout_i=tin_i+t_{\mathrm{exp}},i=0 \tag{13}$$

$$tout_i=tin_i+t_{\mathrm{exp}},i>=1 \tag{14}$$

由于各节点时间窗在初始化后可能与当前节点时间窗链表中的某个时间窗存在重叠,因此需要定义两时间窗无重叠判定规则为需要满足:

$$tin_i\geqslant tout_{i,\mathrm{old}}^{k}\vee tout_i\geqslant tin_{i,\mathrm{old}}^{k} \tag{15}$$

同时定义节点时间窗重叠后的更新原则为:

$$tin_i=tout_{i,\mathrm{old}}^{k} \tag{16}$$

$$tout_i=tout_{i,\mathrm{old}}^{k}+t_{\mathrm{exp}} \tag{17}$$

式中 $tout_{i,\mathrm{old}}^{k}$ 属于节点 v_i 的第 k 个时间窗。

将式(5)所确定动态路径进行扩展,可定义自治区域 AS_k 动态路径规划任务输入形式如下:

$$M_{\mathrm{Route}}=(src,dst,t_{\mathrm{start}},P_{src}^{dst},V,ID) \tag{18}$$

式中 P_{src}^{dst} 为自初始节点 src 至终点 dst 的所有备选路径集合,且按路径长度升序排列。则对该任务有动态阶段算法步骤如下:

步骤 1:设定初始路径 P_0,P_0 为备选路径集合 P_{src}^{dst} 中的第一条最短路径,设定最短完成时间 t_{MIN},t_{MIN} 初值定为无穷大。

步骤 2:设定初始节点 V_0,V_0 为 P_0 中第一个节点。按照式(11)和式(13)初始化时间窗 $T_W^{V0}=(tin_{v_0},\ tout_{v0})$。

步骤 3:按照式(12)和式(14)顺序遍历 P_0 中所有节点为 V_0,将当前 T_W^{V0} 插入 V_0 所维护的升序排列的时间窗占用链表。然后根据式(15)判别是否重叠。若无重叠,则继续执行;若有重叠,则按式(16)及式(17)进行更新,并继续插入 V_0 所维护时间链表,直到找到无重叠插入点。

步骤 4:若在步骤 3 中所找到当前路径 P_0 中各时间窗均为无重合插入,则转至步骤 6。

若存在重叠后更新插入,则记录此时最末节点时间窗 T_W^{V0} 中的 $tout_{V_0}$ 为 t_{MIN},并以路径集合 P_{src}^{dst} 中下一路径作为 P_0,转至步骤 2。

步骤 5:找到对应最小 t_{MIN} 的路径号。

步骤 6:输出该路径 P_0,算法结束。

这种算法最终可为区域内 RGV 路径规划任务规划出一条无冲突的理论最优路径。

2.3　引入启发式策略的多 RGV 跨区域动态路径规划算法

传统集中式的基于时间窗的两阶段动态路径规划算法的主要瓶颈就在于当系统内节点数目或者执行任务的 AGV 增大一倍,则计算量指数爆炸,无法满足实时性的要求。为此本文借助启发式策略中的手段——目标分析法的思想,提出相应的解决方案。具体实施方法是在区域划分的基础上,将跨区域路径规划任务分解为若干个单区域动态路径规划任务来实现。具体方法如图 1 所示,若区域 AS1 内节点 1 处 RGV 要运行至区域 AS2 内节点 11 处,则首先由区域控制器 1 动态规划自节点 1 到区域 AS2 边缘节点的路径,该边缘节点可能为 7 或者 9,随后在 RGV 运行至该边界节点时,再由区域控制器 2 动态规划自该边缘节点至节点 11 的路径。

经过引入这种策略,若将地图划分为 C 个区域,则原有 N 个节点 K 个并行 RGV 的动态路径规划求解规模可分解为 C 个 N/C 个节点、K/C 个并行 RGV 动态路径规划的子问题,同时每个子问题又都由单独的区域控制器执行,则对于每一个区域控制器来说,其求解规模至少缩小了 C^2 倍,从而可以有效解决传统两阶段动态路径规划算法应对大规模地图多 RGV 运行时的实时性问题。同时,由新 RGV 加入新区域所引发的所有 RGV 任务路径重规划又会解决由于 RGV 运行速率不一致导致的实际运行中产生的路径非最优问题,从而进一步提高其实用性。这种算法的关键在于如何确定从当前所处区域到下一区域的边缘节点。

由于本文所作规则对于与当前区域存在直接连通关系的下一区域的任意边缘节点,在当前区域的边缘节点集合中存在唯一边缘节点与之对应,因而可通过确定本区域边缘节点进而间接确定下一区域边缘节点。对于实际应用中不满足这项规则的地图,可通过进一步完善数据结构及评价函数予以解决。

本文通过构建启发式评价函数来寻找本区域边缘节点,具体公式如下:

$$f(n)=T_{\text{image}}(n)+T_{\text{cur}}(n) \tag{19}$$

式中 $f(n)$ 代表本区域自初始节点经本区域第 n 个边缘节点到目标节点的启发式函数,它由两部分构成,第一部分 $T_{\text{image}}(n)$ 表示自该边缘节点到目标节点的理想最短行驶时间 T_{image},第二部分 $T_{\text{cur}}(n)$ 代表自初始节点到该边缘节点按照系统内所保存上一次节点时间窗链表基于时间窗排布出的估计最短行驶时间,即预计到达边缘节点所需花费时间。

离线阶段:

对于图 G_k 找到区域 AS_k 内各节点到其余节点的所有可行路径,组成路径集合 P。对于图 G,应用前文描述的最短行驶时间算法找到各区域边缘节点到全图其余所有节点的理想最短行驶时间 T_{image}。

动态阶段:

步骤 0:若系统出现新任务,则启动系统内动态路径规划,并基于当前任务执行状态更新原有系统内任务作为新的路径规划任务。

步骤 1:按照任务开始时间先后顺序对区域控制器所维护的待规划任务列表进行降序排序。

步骤 2:设定排好序的待规划任务列表第一个任务为初始路径规划任务 M_0。

步骤 3:顺序遍历待规划任务列表中所有待规划任务为 M_0。

步骤 4:判断 M_0 终点 dst 是否属于本区域,若属于本区域,则转至步骤 6,否则转至步骤 5。

步骤 5:根据式(19)从本区域边缘节点集合中选择到终点 dst 理想最短行驶时间最小的边缘节点作为 dst,其余值不变,从而重新定义 M_0,并转至步骤 4。

步骤 6:执行 2.2 节中的基于节点时间窗的区域内 RGV 动态路径规划算法,确定该任务路径。

步骤 7:完成所有路径规划任务,算法结束。

3　仿真实例验证

3.1　实验思路

由于本文所提出的分布式两阶段动态算法为一种分布式算法,在规划全局路径任务时,需要多个控制器协同合作,无法借助传统的仿真实验平台进行测试。为对本算法进行性能测试和比较,本文选择通过设计相应测试示例程序模拟多 RGV 运行流程和算法执行流程实现。主要思路是通过设计如表 1 所示的 RGV 数据结构 RGVDATA 以及如表 2 所示的 RGV 位置数据结构 RGV_POS,并借助 Visual C++环境提供的定时器 Timer 模拟系统运行时钟,在定时器处理函数 OnTimer 中遍历 RGVDATA 数组中单个 RGV 的位置信息与 RGV 的状态信息,如处于运行状态或停止状态,根据其数据信息进行相应更新,同时将本算法与集中式两阶段路径规划算法以及静态路径规划算法实验结果进行比较,最后得出相应结论。

表 1　RGVDATA 数据结构说明表

数据名称	数据类型	数据含义
ID	Int 整型	RGV 编号
State	RGV_STATE 枚举值	RGV 状态
Pos	RGV_POS 自定义结构体	RGV 位置信息
TaskID	Int 整型	RGV 执行任务编号

表 2　RGV_POS

数据名称	数据存储类型	数据含义
areaID	Int	表示 RGV 当前所处区域
RunID	Int	代表当前节点号或 当前所处轨道下一节点号
posID	Int	代表离开当前节点或下一节点所需时间

3.2　实验平台及主要程序流程

本测试实验选用 Windows7 为系统平台,测试开发工具为 Visual 2010,测试开发语言为 C++。

实验主要分为两个主要程序流程:一个是如图 2 所示的主程序流程,主要实现遍历系统内待规划路径任务数组,从而触发路径规划程序,针对本文分布式路径规划特点,设计了区域控制器数据结构 VAREA,主要存储本区域内地图信息及 RGV 信息以及本区域内执行任务数组及待路径规划任务数组;另一个是如图 3 所示的定时器 Timer 触发函数流程,该函数每隔 1 s 触发一次,用来更新系统实时状态。系统仿真时钟计数器定为 *TCLK*,初始为 0,每次 Timer 函数触发时自动加 1,此处设计为当 *TCLK* 超过 100 时即停止,同时为便于分析,设置各节点处停留时间为 4。

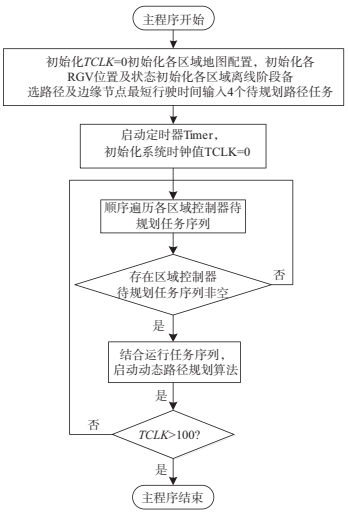


图 2　主程序流程图

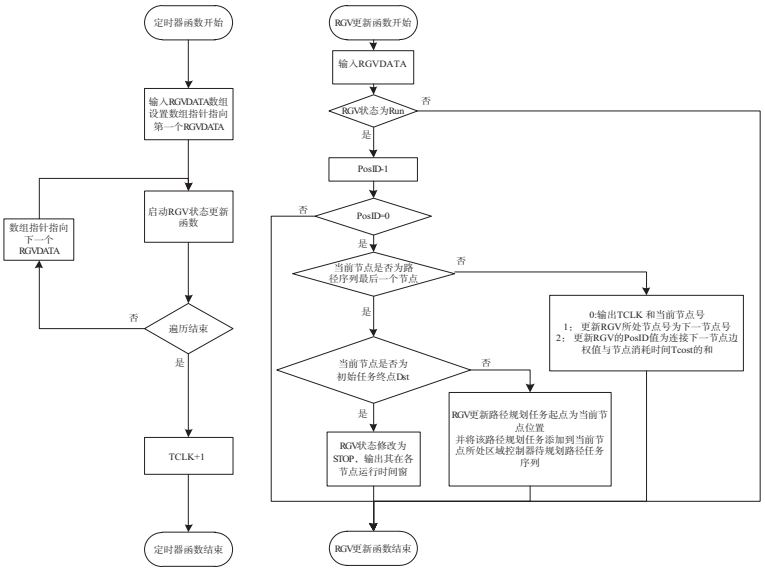


图 3　定时器触发流程

3.3　比较实验设计

为比较本文提出的分布式动态路径规划算法与传统的集中式动态路径规划算法以及静态路径规划算法的区别,本文选择在图 1 所确定的区域自治地图下通过设计特定的路径规划任务来进行比较实验,所选方案为选择同一区域内 2 个路径规划任务进行路径规划测试和运行情况比较。具体任务设计形式如表 3 所示。

表 3　待规划路径任务表

任务编号	src	dst	tin0	tout0	执行 RGV
1	1	8	8	12	1
2	2	8	0	4	2

3.4　比较实验结果及分析

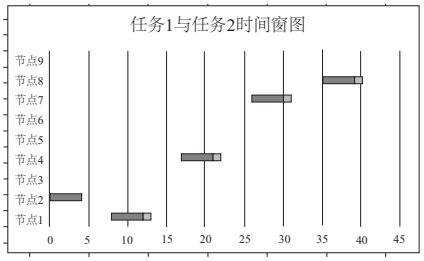
将这两次路径规划任务分别在集中式两阶段路径规划算法、分布式两阶段路径算法、Dijkstra 最短路径规划算法下进行路径规划,最后可得表 4 所示实际规划路径序列。

绘制出这 3 种算法的时间窗图如图 4 所示,其中图 4a)为 Dijkstra 算法所规划路径下运行时间窗,图 4b)为分布式动态算法和集中式动态算法实际运行中所产生时间窗,由于两者数据相同,因而一起讨论。图中深色部分为任务 1 对应时间窗,浅色部分为任务 2 所对应时间窗。则据图 4a)可知 Dijkstra 算法所规划路径在中途节点 1 点处开始存在时间窗重叠,重叠时间为 3 s,由于在 RGV 实际运行中重叠节点处为转轨器站点,不可被重复占用,因此任务 2 静态算法下抵达 8 节点实际时间应为 39 s。据图 4b)可知两种动态路径规划算法均可得到除终点外无重叠时间窗最优路径,则动态规划算法下任务 2 抵达 8 节点处时间为 37 s。此外集中式规划计算时间为 0.012 s,分布式规划计算时间仅为 0.007 s。可见,分布式动态路径规划算法可在用时远短于集中式动态路径规划算法的情况下,

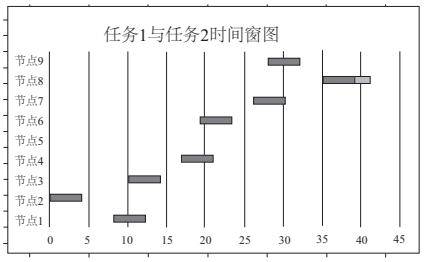
得到最优或次优路径。

表 4　3 种算法规划路径序列

任务编号	分布式动态算法	集中式动态算法	Dijkstra 静态算法
1	1 478	1 478	1 478
2	23 698	23 698	1 478



a) Dijkstra算法时间窗分布图



b) 分布式与集中式算法时间窗分布图

图 4　3 种算法最终路径时间窗分布比较图

4　结语

本文针对医院轨道物流传输系统中多 RGV 路径规划问题,提出了一种基于区域自治的分布式动态路径规划算法,该算法旨在解决传统集中式两阶段动态路径规划算法应对大规模地图实时性较差的问题。首先通过区域划分,构建了引入边缘节点的区域自治地图模型,随后在该模型的基础上,结合时间窗理论和两阶段控制策略,解决区域内多 RGV 路径规划问题。对于跨区域路径规划问题,通过引入启发式策略寻找边缘节点,最终实现多 RGV 分布式动态路径规划问题的解决。通过设计相应示例程序完成了小型地图仿真实验,实验验证了该算法相对于集中式动态路径规划算法在实时性上的优越性。

参考文献:

[1] 沈崇德. 医院物流传输系统浅析[J]. 中国医院,2009, 13(3):74-76.

[2] 赵红梅,李远达. 医院物流自动化输送系统分析[J]. 中国医院, 2015, 19(6):76-78.

[3] 赵昆. 快速轨道医流智能监控与调度系统关键技术的研究[D]. 南京:南京航空航天大学, 2016.

[4] Chang W Kim,Jose MA Tanchoco. Conflict-free shortest-time bidirectional AGV routeing[J]. The International Journal of Production Research, 1991, 29(12):2377-2391.

[5] Lee J H, Lee B H. Real time traffic control scheme of multiple AGV systems for collision Free minimum time motion: a routing table approach[J]. IEEE Transactions on Systems, Man And Cybertnetics, Part A: Systems and Humans, 1998, 28(3) : 347-358.

[6] 刘国栋, 曲道奎, 张雷. 多 AGV 调度系统中的两阶段动态路径规划[J]. 机器人, 2005(3):210-214.

[7] 王佳溶, 楼佩煌, 王晓勇. 基于改进的两阶段控制策略的 AGV 路径优化调度研究[J]. 机械科学与技术, 2008(9): 1211-1216.

[8] 胡彬, 王冰, 王春香,等. 一种基于时间窗的自动导引车动态路径规划方法[J]. 上海交通大学学报, 2012,46(6):967-971.

[9] 张峥炜, 陈波, 陈卫东. 时间窗约束下的 AGV 动态路径规划[J]. 微型电脑应用, 2016, 32(11):46-49.

[10] Ling Qiu, WenJing Hu, SheYing Huang, et al. Scheduling and routing algorithms for AGVs: A survey[J]. International Journal of Production Research, 2002, 40(3):745-760.

收稿日期:2018-02-26