**本文内容简介：**

正文

# 结构内容

## 概括和目的，添加ing related work

For most situations, computing a collision-free path from one point to another point for single robot is not a difficult task. But if refering to multi-robot system, things will be much complex. In many prior works, priority configuration is a significant parameter, which plays a crucial role to the planning result.

In order to obtain better priority configuration in a multi-robot path planning process, in this paper we propose a prioritized approach, whose key idea can be described as:

One priority configuration would be regarded as a discrete state, which is obtained by a *priority planner* iteratively, and *path planner* conducts path searching in real map under each configuration. That is, if we have n robots in the planning task, we would have n! sets of priority configuration, therefore n! priority planners, and n path planners within each priority planner. Every time after path planner comes out with a valid result, we estimate it and translate evaluation to priority planner as feedback. The next priority planner would try out configurations in an order based on the value of the feedback. Via this iterative procedure, the planning task not only ensures that each robot goes to its destination in real map (if a valid plan exists) but also utilizes feedbacks for priority configuration such that an efficient plan can be found with the minimum time of searching.

为了在多机器人路径规划过程中，获得更加优秀的优先级配置，XXX方法将task and motion planning的思想引入其中：

一种机器人优先级序列作为一种action，对于每一个action，需要在代价空间中规划一条无碰撞、代价小的路径。即，从s到g。

因此，我们可以将多机器人路径规划问题抽象为两个层面：

一个层面是上层的action planning，也就是对优先级序列的优化。因为在同一环境、同样的机器人任务条件下，可以证明，必然存在一种最有的优先级序列，使得所有机器人的规划总代价取得最小。这样的代价函数可以是所有机器人的规划路径的总长度，也可以是最后一个机器人完成任务的时间，也就是说，代价函数的选取依靠具体的任务环境来刻画。

对应与每一种action（优先级配置），需要为每一个机器人进行motion planning以获得一条对应于该机器人起点、终点位置的，与环境、其他机器人不相互碰撞的安全、高效路径。这样的方法很多，我们可以选择适应于某具体环境而更加高效的方法。

本片问题关心的地方在于，如何获得一种最优的优先级序列。

## 术语定义

This paper defines the prioritized path planning task for multi-robots as a tuple , where

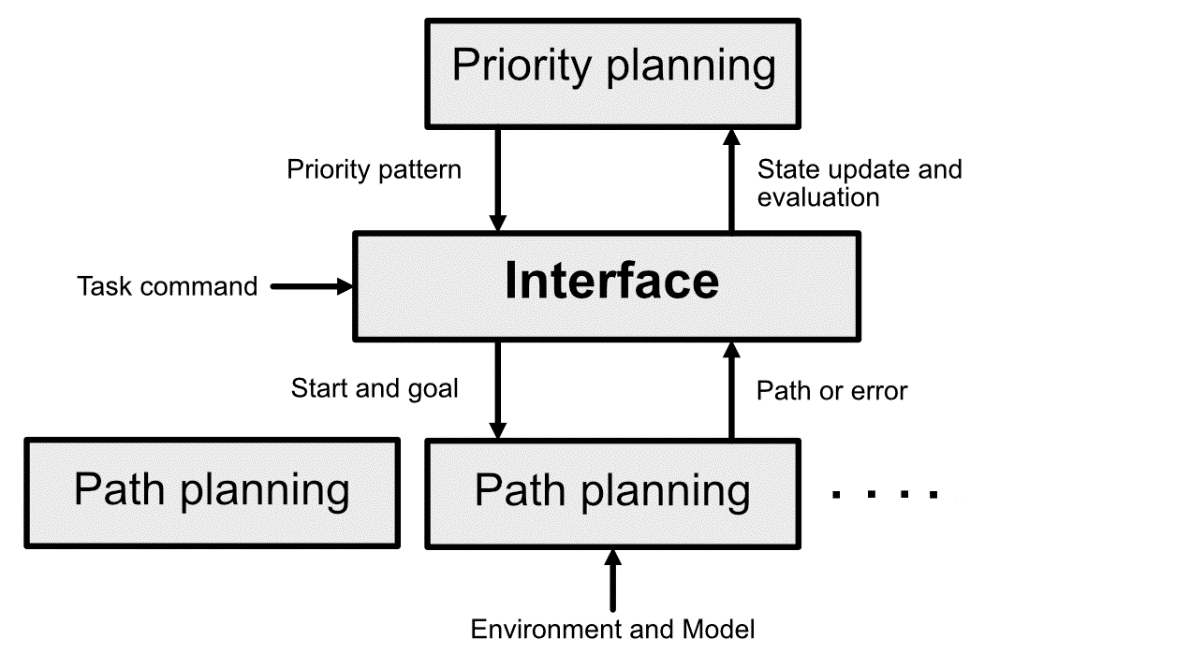
*  denotes the system of multiple robots, which consist of from a set of robots as , where  is the number of robots in system.
*  denotes the set of priority patterns. A priority pattern  is a specific robot sequence, which can be represented as , where . Generally for  robots system, there is  priority patterns.
*  denotes the state space, in which a state  of all robots’ positions usually is expressed as a set . Here  is the  robot’s position coordinates in the map.
*  and  denote the start state and the target state of the path planning process. Apparently  or  is a set state in , for this reason they can also be represented as a position set of all robots in system.
*  denotes a set of feasible path planning algorithms for the specific environment. Not a few path planning algorithms have been developed in past related works, for example A\*, RRT, D\* algorithms.
*  denotes the type of feedback from planned path, which is designed for communication between path planner and priority planner. It will be discussed in detail in the following article.
*  denotes the result paths for all robots in system, which is a set . A  is a collision-free path from start position to target position as .

## 引入TMP思想并适应化改造-method

To provide a collision-free, efficient path for all robot from its own start position  to target position , a new approach will be discussed in this paper, which conducts searching both in priority set  and state space  based on the idea Task and motion planning (TMP).

In other words, a whole searching construct can be divided into two main layers, namely upper planner - priority planner and lower planner - path planner. To this two-layer searching construct, the key point to achieve it is to realize the communication between priority planner and path planner. However, problem is that priority planning is discrete, while the path planning, unfortunately, is continuous. On one hand, priority planner hard considers the underlying continuous, physical constraints while searching in the discrete priority set , for example, obstacles. On the other hand, the path planner also has difficulty using the upper-layer’s information to guide its lower-layer path searching, although that information theoretically can optimize the search result.

Our approach is to design an interface layer between the upper priority planner and lower path planner, to enable communication between priority planning and path planning in the searching process. Basically, this interface layer extracts a task defined feedback, and periodically translates it to the upper priority planner, which would guide the search to the end iteratively. In this way the path planning construct can obtain not only a optimal path set but also an optimized priority pattern.



tmp算法的思想在于设计一种上层action规划与下层action规划的应用接口层，使得上下层规划之间能够进行有效的信息交互。smap算法作为tmp算法中的一种，其获得过icra的优秀论文。其核心思路在于设置一种motion planning所生成的feedback，将feedback通过中间层返回给离散的action层，将action曾进行迭代以逼近更优秀的解。

我们需要的工作是设置一种上下层交互的feedback形式，并设计一种路径搜索的迭代形式算法框架，完成多机器人之间的信息交互规则。



//TODO是否需要设置距离近的才设为不可通过。

1. 每一步的搜索算法 Searching\_path()

在每一步碰撞检测，简单引述一下具有时间信息的a\*算法。

**Path planning**

Function **** searches a collision-free path for robot  with a limited length. Hence usually it may not reach the robot’s goal position but be a part of the whole path. In this way our searching is discrete and the whole path would be obtained as an iterative process.

Many path planning methods for single robot path planning were developed in related works, therefore our approach can take advantages of these researches and employ them as a plug in our algorithm. However, the related position with other prior robots should be considered, more specifically other prior robots should be regard as a dynamic obstacle for other lower prior robots.

In this paper a method A\*-T that is similar with A\* algorithm but considering conflict with other robots is used. The main modification of A\*-T is:

* It has a limited length  of result path.
* It regards adjacent map point  that is contained in prior robot’s path in one iteration as inviable.

These modifications allow A\*-T can be conducted iteratively and after every iteration return a basic evaluation about the short, planned path. Pseudocode is given in Algo. 2.

This method was tested in a very narrow map, in which just two passageway robots can pass through. Two robots were placed at the both ends of the passageway respectively, in addition, one has start and goal position as the other’s goal and start position on the contrary. The result of this test can be seen in fig. x and fig. x as three-dimensional paths. The height of every points in path(green and red) is time cost from relevant start point, which here simply taken as distance.

|  |  |
| --- | --- |
|  |  |
| TODO 这里将上面的图换成二维做标注和展示 | TODO 这里将上面的图换成二维做标注和展示 |



**result evaluation**

The evaluation of one iteration is designed to represent, from which state it will be more possible to reach the goal position and it better can work efficiently in practice. Concretely, in our approach  is obtained as

,

Where  denotes the set that contain all planned paths for robots, and  denotes the last point of short path in one iteration. Apparently, this evaluation describes the sum of all robots’ remaining distances to their goals. The function would be invoked every time after iterative search to provide feedback(evaluation). Robots may find a set of paths to their targets more quickly if choosing to expand from a priority state with a minimal evaluation.

**priority planning**

In our approach priority planning is a tree search. Throughout the planning process, a tree like data structure  is maintained to form the search. The initial state  of robots is given as root of tree , and the goal state  is the final state in state space that is needed to reach via tree’s grow.

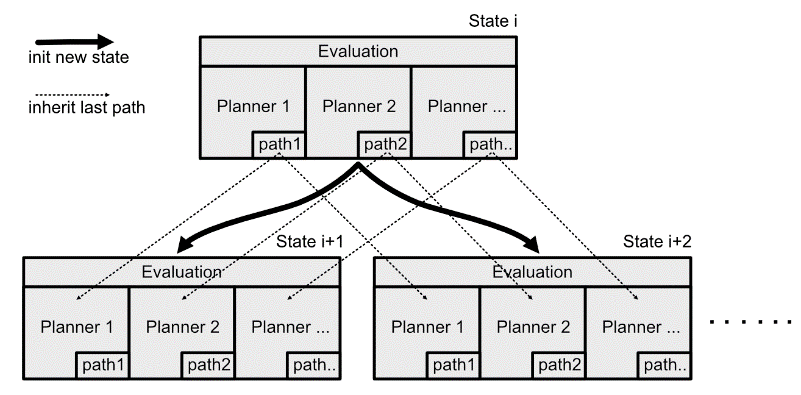
The priority planner explores among the state space. Function  selects a vertex , the next vertex to expand. Function  generates new vertices  and add them to tree . ( here denotes a priority configuration. )

Ever vertex  has a member variable , expressing how good a vertex is as described hereinbefore. During involving function , a vertex  is selected if it has a minimal  along all . Or using a more probabilistic method a vertex  is then selected with probability proportional to its evaluation,



In this way vertex  associated with high evaluation have a greater chance of being selected.

Every vertex  is a path planner that has been described hereinbefore. From the perspective of breadth grow, in one iteration, the number of vertices is determined by the variety of priority configurations. Generally, for  robots system, there are  priority configurations, that is, in every iteration,  would be inserted as children to . Then from the perspective of depth grow, new vertex  would inherit its parent vertex’s state. would take the current position of  as its starting point and generate path accordingly. Function  is represented in pic.x to make it easier to understand.



Till now, our searching tree  can grow from state space  until a new vertex  that satisfies the goal state , before an integrated path is generated by connect the segmental path from the last vertex  to the root vertex . However the most important thing is that, thanks to the evaluation feedbacks from every vertex, which is in fact plays as a heuristic information, we no longer need to search the whole state space .

因为本身是一个np问题，所以这样的运算代价是可以接受的。

~~考虑继承关系需要考虑继承的深度~~

~~这涉及到了局部极值的问题，需要调节继承的深度。~~

~~同时，迭代的离散化还有预见性问题。尤其对于非凸的地图环境中，没有预见性的迭代可能会导致陷于局部极小。~~

在下一次的迭代中，先遍历树，寻找具有最好value（feedback）值的节点，基于该节点继续往深度方向搜索。基于这样的方式，形成一种利用贪心信息的迭代搜索，这样，我们可以不用遍历整个树空间，进而减小我们的搜索量。

|  |  |
| --- | --- |
| 树的生长流程图可以仔细画一个：    这里可以突出展示一下，因为贪心信息的引入，而可以不用去把所有的树节点搜索完，这棵树可以称为葫芦状，先长大，然后会收敛。  收敛性需要证明吗？ | 上下层交互的机器人群体移动的图可以画一个： |

## 实验与仿真

证明算法的可行性，两个机器人的优先级对规划成功与否的影响；

多个机器人规划的路径的优秀程度，与其他算法的对比

## 结论与展望

在总结与展望的时候，可以对地图环境的非凸性做一些论述。这样，可以为未来的研究方向做一些介绍。

# 图表、示意图等内容

1. 两三个机器人移动的过程手算可以来一个，然后用两三种不同颜色的线条表达机器人的移动过程，将图中的地图点进行赋值变化。这样一个简短的移动过程，可以用几张图来表达。
2. 预测多少步与走多少步的搜索长度的解耦，这样可以获得更好的预见性。这与地图的非凸性是有关的，可以论述一下为什么要这样做，也可以画一张图举一个例子。
3. 可以简述一下机器人优先级对路径规划结果的影响，就用之前看到的某篇论文的描述吧，注意引用。

可以说明，在最糟糕的情况下，机器人的优先级不仅会影响到路径规划的效果，甚至会决定路径规划成功与否。

1. 最后在仿真的时候，仓库的地图可以来一个。

# 其他未用的材料

发生第二次碰撞的情况，需要讲一下吗？

我们会希望这样的过程能够迭代起来，应为对于通常的方法来说，要获得一群机器人的最有优先级序列是一个np问题。xxx提出了模拟退化法来求得具体机器人群体的最优优先级，即在优先级序列空间中进行ramdom sample，从所有的sample中获得一个最好评价值的优先级序列。